Shahid Beheshti University
Data Science (2023-2024)

Department of Computer Science
Shahid Beheshti University
Asal Rahbari

Abstract

Sentiment analysis on 515K hotel reviews

Recommendation System

Regression Model

Clustering

Introduction

Sentiment analysis

Sentiment analysis, a component of Natural Language Processing (NLP), involves discerning emotions from raw texts. Commonly applied to social media posts and customer reviews, it aims to automatically determine whether users express positivity or negativity and the reasons behind it. Our focus is on analyzing hotel reviews, where each review comprises textual feedback on the customer's hotel experience and an overall rating. The objective is to predict whether a given textual review represents a positive or negative sentiment. The overall ratings, ranging from 2.5/10 to 10/10, are categorized into:

- "bad reviews" (overall ratings < 7)
- "good reviews" (overall ratings >= 7).

The challenge lies in accurately predicting this categorization using only the unprocessed textual data from the reviews.

Here are some of the challenges:

Subjectivity: Sentiment analysis is subjective and can be influenced by the context in which the text is written. For example, a review that says “The hotel was not bad” could be interpreted as either positive or negative, depending on the context.

Data quality: The quality of the data can also be a challenge. Reviews may contain spelling errors, grammatical mistakes, or other issues that can affect the accuracy of the sentiment analysis.

Related work/Background

The main purpose of our project is applying sentiment analysis on some hotel reviews, so here I explain some of the approaches and methods that are usually used to make this happen.

Generally, there are several methods for sentiment analysis, but among all, three of them are frequently used:

- **Rule-based Methods:** This approach is based on the idea of generating a set of rules and then use those manually constructed rules to identify the sentiment.
  ***How are these rules set?** First, we should create a list of words and phrases then, assign a sentiment score to each of them. Clearly, this sentiment score can be positive, negative or even neutral.
  ***How are these words chosen?** This is pretty much up to the data or text you're working with. For example, in our case we're trying to find the sentiment for hotel reviews, based on the context of our project we can understand which words should be included in our list. Words like room, bad, clean, good, fresh, dirty, nice, and so on are all related to our subject.
- **Machine learning-based Methods:** In this method we try to train a machine learning model on our dataset which consists of texts labeled by their sentiment. In this case, our model learns the relation and association between labels (sentiment scores) and the patterns in the text, like the words it contains, the order of those words, etc. After training the model it surely would be able to predict the sentiment of a given text with a certain amount of accuracy.
  Some algorithms used are: Naive Bayes, Support Vector Machines, Convolutional Neural Networks, …
- **Hybrid Methods:** As you may have already guessed, this one is the combination of the previous method we talked about. Using this method, we're aiming for taking advantage of both methods mentioned earlier. We generate the features using rule-based method and then train our machine learning model on those features.

What are some libraries used for sentiment analysis?

1) **NLTK:** The Natural Language Tool Kit is one of widely used libraries for sentiment analysis. It includes a variety of tools for sentiment analysis like a pre-defined sentiment classifier, sentiment lexicon and so on.

2) **Scikit-Learn:** This library is one of the most popular ones among all, which includes many of the machine learning algorithms that can be used for sentiment analysis like Naïve Bayes, Decision Tree, …

3) **TextBlob:** This library of python provides us with a simple API for common natural language processing tasks, which of course includes sentiment analysis. It has a pre-trained sentiment model that classifies a text intro one of the groups positive, negative or neutral.

4) **Vader:** It's a rule-based sentiment analysis tool that is specifically designed for social media text. It uses a sentiment lexicon and a set of pre-defined rules to assign a sentiment score to a text.

Proposed method

Here are some methods that we used in Text Cleaning:

Lowercasing: All text in the reviews was converted to lowercase. This step ensures consistency and prevents the model from treating the same word differently due to case variations.

Punctuation Removal: Special characters and punctuation marks (e.g., commas, periods, exclamation marks) were eliminated. These symbols do not carry sentiment information and can be safely removed. This process helps break down the text into meaningful units and removes unnecessary symbols.

Tokenization: The reviews were split into individual words (tokens). Tokenization is essential for further analysis, such as feature extraction and modeling.

Stop Word Removal: Common words like "the," "and," and other stop words were likely removed. These words do not add much value to sentiment analysis and can be safely excluded.

Lemmatization or Stemming: Words were reduced to their base form. For example, "running" and "ran" would both be transformed to "run." This step helps reduce dimensionality and improve model performance.

Removing Words with Numbers: Filter out words that contain numeric digits. This step helps remove numbers and numeric expressions from the text, as they may not contribute to sentiment analysis.

Removing Empty Tokens: Any empty tokens resulting from previous steps are removed. This ensures that the processed text contains meaningful content.

Part-of-Speech (POS) Tagging: Perform POS tagging, assigning a grammatical category (e.g., noun, verb, adjective) to each word.

Removing Single-Letter Words: Words with only one letter are filtered out. Such words are often not informative for sentiment analysis.

Joining Tokens Back into a Cleaned Text: Finally, the cleaned tokens are joined back together to form a coherent and processed text.

Here are some methods that we used in Feature Engineering:

Sentiment Analysis Columns:

Calculate sentiment scores for each review. These scores represent the positivity, negativity, neutrality, and overall sentiment of the text.

Number of Characters and Words Columns:

Create two new features:

- Number of Characters (nb_chars): This feature captures the total character count in each review.
- Number of Words (nb_words): This feature counts the words in each review after splitting by spaces.

These features provide insights into the length and complexity of the reviews.

Doc2Vec Vector Columns:

Generate document vectors using the Doc2Vec technique.

These vectors represent the semantic content of each review, allowing for similarity comparisons and clustering.

Enhancing the representation of the reviews for downstream tasks, by adding these vector columns.

TF-IDF (Term Frequency-Inverse Document Frequency) Columns:

Create features based on word importance using TF-IDF.

These features capture the relevance of specific words within each review relative to the entire dataset.

Enabling models to consider the significance of individual words in sentiment analysis or other tasks, by incorporating these columns.

In summary enhancing the dataset by introducing sentiment-related features, text length metrics, document vectors, and word importance scores.

Here are some methods that we used in Modeling:

Classification Method Used:

Random Forest Classifier for sentiment analysis.

Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It’s robust, handles non-linear relationships well, and can handle both numerical and categorical features.

Classification Results:

The model predicts whether a review is a bad review (negative sentiment) or not.

Sentiment Classes Predicted:

Positive: Reviews that are not bad (i.e., not classified as negative).

Negative: Reviews classified as bad.

Receiver Operating Characteristic (ROC) Curve:

The ROC curve visually represents the model’s discrimination skills.

It plots the true positive rate (TPR) against the false positive rate (FPR) at different decision thresholds.

The area under the ROC curve (AUC) quantifies the model’s ability to distinguish between classes.

An AUC of 0.86 indicates good performance.

Recall curve:

The Precision-Recall curve visually illustrates how well a model performs in terms of precision and recall. It plots precision (y-axis) against recall (x-axis) for different decision thresholds. The area under the curve (AP) is calculated as 0.61, indicating moderate performance.

Precision represents the proportion of correctly predicted positive instances among all predicted positive instances.

Recall (also known as sensitivity) represents the proportion of correctly predicted positive instances among all actual positive instances. This curve assesses the trade-off between precision and recall, crucial for imbalanced classification tasks.

Here are some methods that we used in Clustring:

Clustering Method Used:

I applied the K_Means clustering algorithm to the geographical data points (latitude and longitude) in the dataset. K_Means is an unsupervised machine learning technique that aims to group similar data points into clusters based on their feature similarity. The algorithm iteratively assigns data points to the nearest cluster center (centroid) and updates the centroids until convergence. The number of clusters (k) is specified beforehand (in this case, 6 clusters).

Summary of Results:

After cleaning the dataset by removing any null values, K_Means clustered the data points into six distinct clusters. The clusters are visualized on a map of Western Europe, where each point is colored according to its assigned cluster ID. The resulting clusters represent regions with concentrations of data points. The color bar indicates the cluster IDs, allowing us to identify different geographical areas.

Result

Data Context

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

Data Content

The dataset has 515738 rows and 17 columns. 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'
- **Review Total Negative Word Counts:** 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'
- **Review Total Positive Word Counts:** Total number of words in the positive review.
- **Reviewer Score:** Score the reviewer has given to the hotel, based on his/her experience
- **Total Number of Reviews Reviewer Has Given:** Number of Reviews the reviewers has given in the past.
- **Total Number of Reviews:** Total number of valid reviews the hotel has.
- **Tags:** Tags reviewer gave the hotel.
- **Days since review:** Duration between the review date and scrape date.
- **Additional Number of 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:** longitude of the hotel

Data processing

We start by calculating the unique values for each variable, to provide insights into the diversity and distinctiveness of the data. The results show the number of unique entries for each attribute, shedding light on the range and individuality of information within the dataset. This information can be crucial for understanding the scope and variety of the dataset, offering a foundation for subsequent statistical analysis.

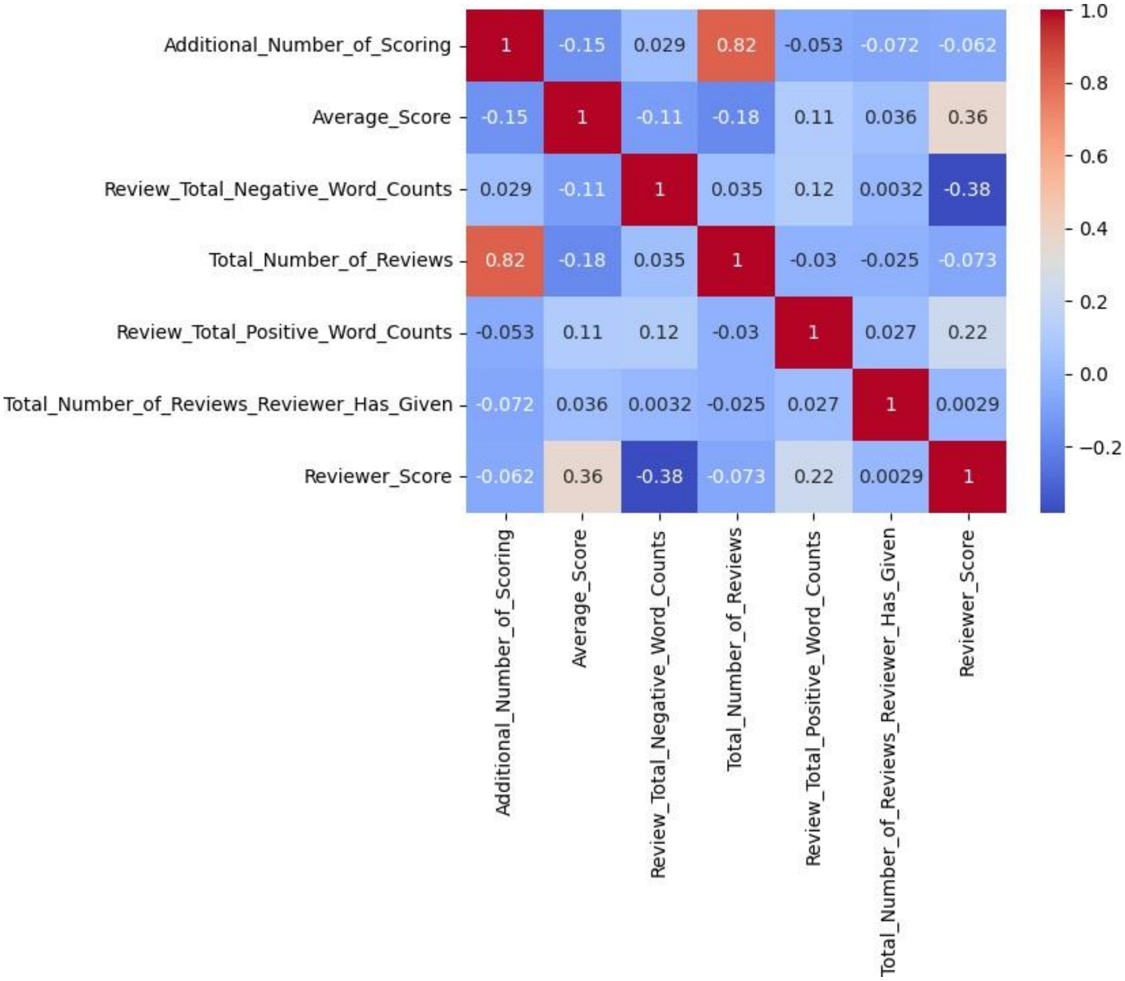
- **Null values**
  In the data processing phase, a comprehensive analysis of missing values was conducted. The process involved identifying columns with missing data, calculating the number of missing entries in each relevant column, and determining the corresponding percentage of missing values relative to the total dataset size. The results were then presented in a clear tabular format, showcasing the number and percentage of missing values for each applicable column. In this specific instance, the output table reveals that two columns, labeled 'lat' and 'lng,' exhibit 3268 missing entries each, constituting 0.63% of the entire dataset for both columns. Currently, the focus does not include addressing missing values in these two columns, because they won’t be utilized in the implementation of statistical methods and dataset manipulation.

Visualization

In this part we used different kinds of plots to visualize various aspects of our data.

- **Correlation between numerical columns**
  Correlation is a measure of the strength and direction of the relationship between two variables. Correlation coefficients range from -1 to 1, where -1 indicates a perfect negative correlation, 0 indicates no correlation, and 1 indicates a perfect positive correlation.
  Positive correlation tells us that those variables increase or decrease together while, the negative one tells us vice versa.

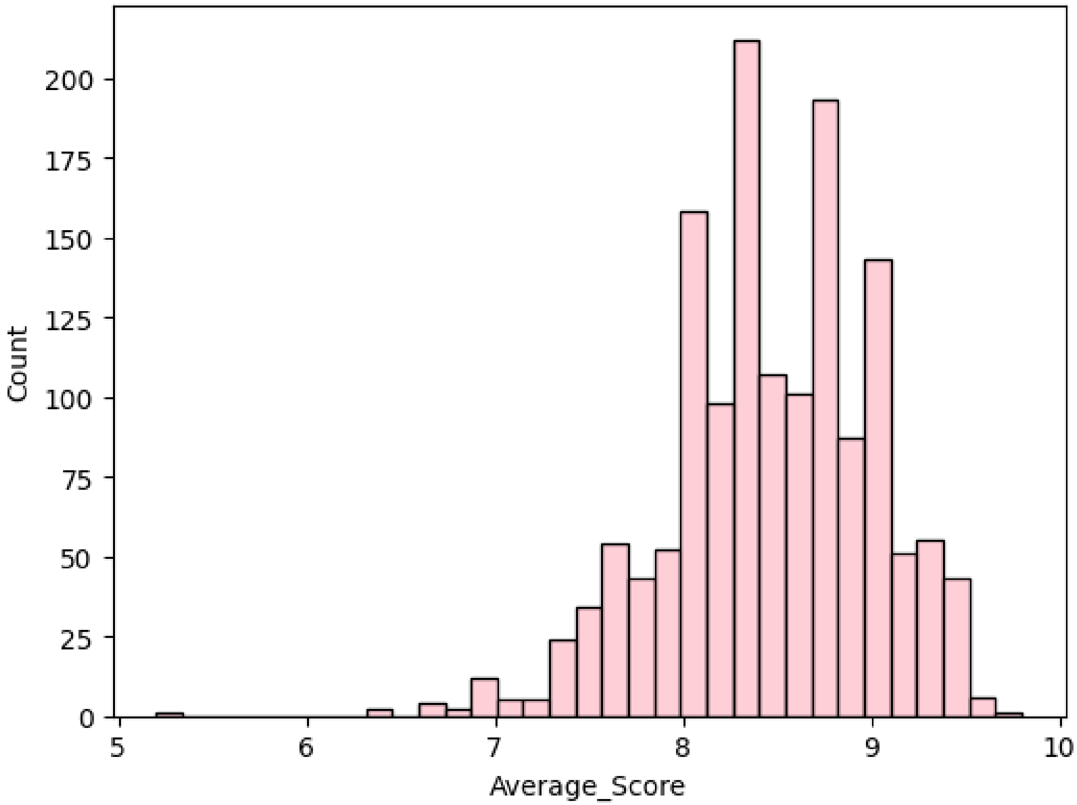
Here we wanted to see the correlation of our numerical features:



As you can see some pairs are poorly correlated, some highly. Some have positive correlation coefficient, some have negative.

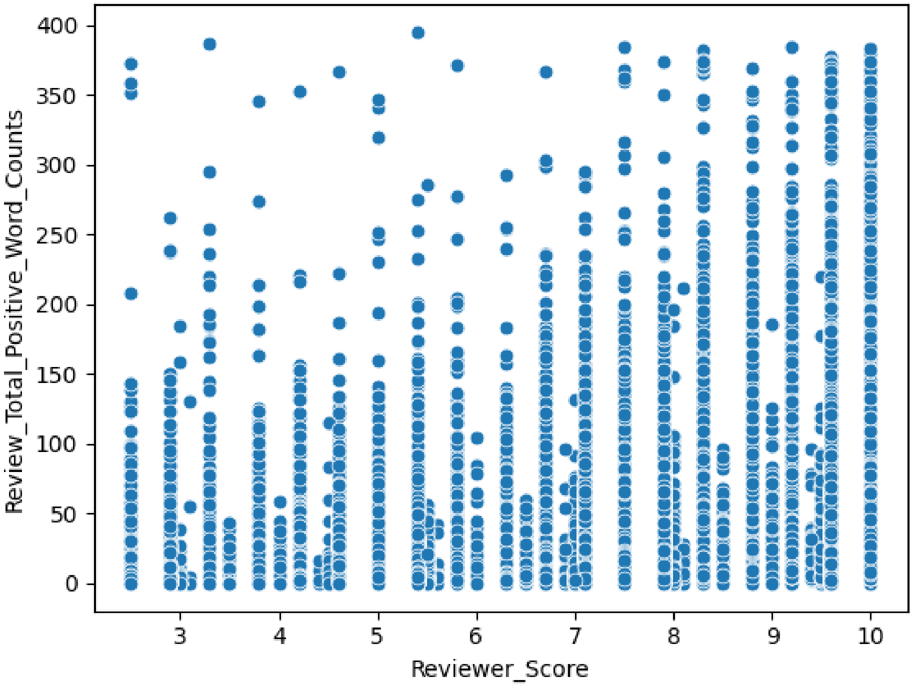
- **Average Score Distribution**
  The distribution of data can be shown by the shape of it when all of the possible values are plotted on a frequency graph. By observing the distribution of data we can gain some valuable knowledge about our data. For example, if our data has a normal distribution we know that it’s unimodal and the mean and median are the same.

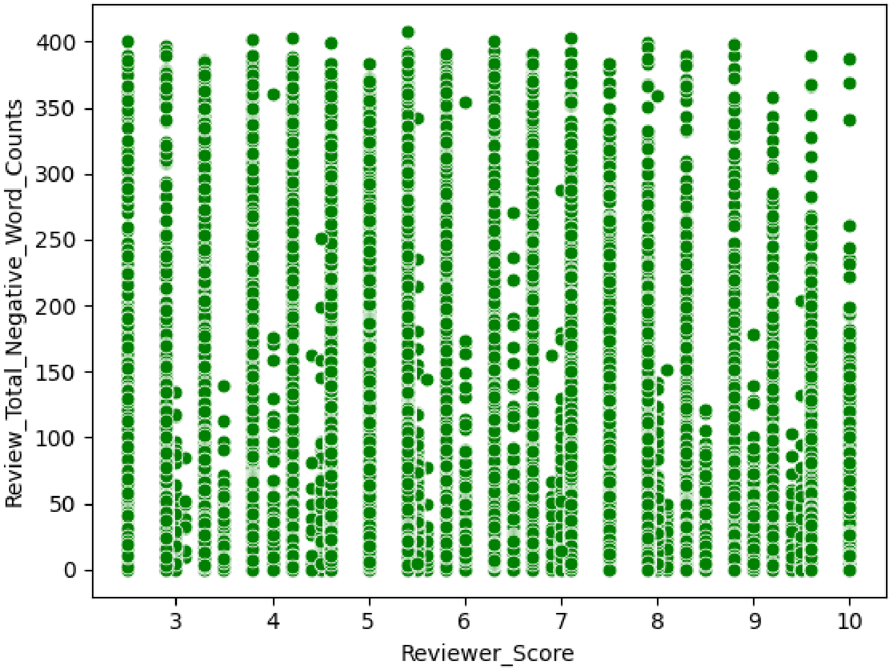
  In our dataset we have a very important feature called average score, which shows the overall score of a hotel. It can be useful for our further tasks to know the distribution of this feature.



According to this plot we can assume that most of the hotels across the globe are acting fairly good and have reasonable scores.

- **Negative/Positive words effect on the reviewers score**
  Using Scatter plot we’re trying to have a better presentation of how the number of positive or negative words in a review affects the score that the reviewer gives to hotel based on their experience.

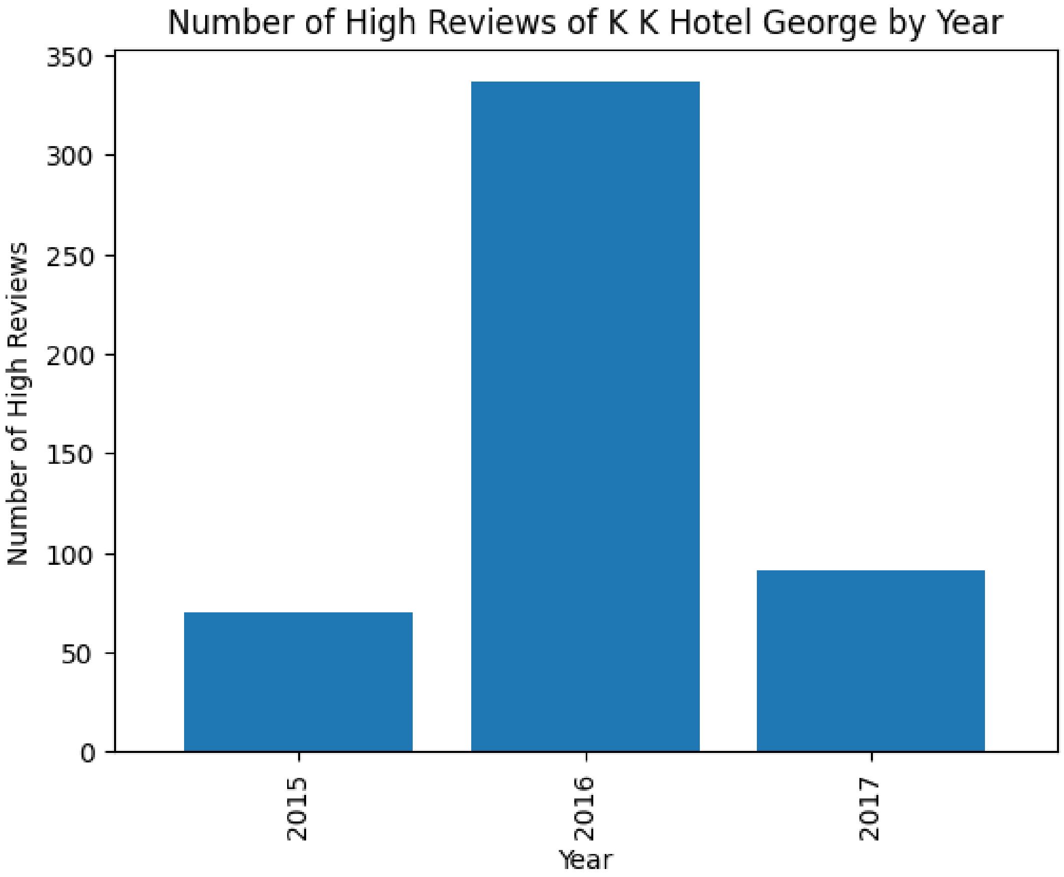




- **Review Date(Year) Distribution**

  Since we have a feature called review date, it’s a good idea to also give it a glance. Here, we decided to check the number of high review scores that each hotel received through the years that have been recorded and stored in our dataset.

  After some Analysis we figured out that there are only three distinct years included in this dataset, which are 2015, 2016 and 2017. Then, we plotted a bar chart for each of the hotels in which, each bar represents a year and the height of the bar is the frequency of the received high scores in the specific year.



- **Correlation between reviewer's nationality and scores**
  The process involves grouping the data by nationality and calculating the average scores assigned by reviewers from each nationality. The results are then visualized using a bar plot, where each bar represents the average review score associated with a specific nationality. This graphical representation offers insights into how review scores vary across different nationalities, providing a visual summary of the average sentiment expressed by reviewers from each nationality. The x-axis denotes the reviewer nationality, the y-axis represents the average review score, and each bar corresponds to a specific nationality, showcasing the distribution of scores across different groups.

Sentiment analysis

In the sentiment analysis (part of the Natural Language Processing (NLP)) section, the focus is on extracting emotional tones from raw text data. This process aims to understand whether users express positive or negative sentiments in their reviews. To start, the overall rating scores of the reviews are examined, ranging from 2.5 to 10.0. The objective is to classify reviews into two distinct categories: those with overall ratings less than 7 are labeled as bad, while those with ratings equal to or greater than 7 are considered good. The subsequent task involves predicting this categorization using only the raw textual data from the reviews.

To achieve this, each review is split into positive and negative components. Initially, these components are grouped together, streamlining the analysis to focus solely on the raw text data. The sentiment label, indicating whether a review is classified as bad (1) or good (0), is assigned based on the predefined rating threshold which is 7 here. The resulting Data Frame provides a glimpse into the processed data, showing the positive and negative textual information along with their corresponding sentiment labels.

- **Sampling data**
  To speed up computational processes, a data sampling step is implemented. This involves selecting a subset of data from the original dataset. By getting a sample, computational efficiency is improved without compromising the overall integrity of the analysis. The sampling is performed with a specified fraction of the dataset, ensuring randomness and reproducibility through a defined random state.

- **Cleaning data**
  In the sentiment analysis section, data cleaning is an essential step to refine and prepare the text data for subsequent analysis. Firstly, instances where users provide feedback without specifying negative or positive comments are identified and removed, streamlining the textual content. Following this, a series of operations are applied to further clean the text data. These operations encompass tasks such as removing punctuation, stopwords, and performing lemmatization. Punctuation removal aids in simplifying the text, while stopwords, common words with minimal semantic value, are excluded to focus on more meaningful content. Lemmatization is employed to reduce words to their base or root form, ensuring consistency and improving the accuracy of subsequent analyses.

To accomplish these tasks, natural language processing (NLP) tools such as the NLTK library are utilized. Specifically, part-of-speech tagging is employed to determine the grammatical category of each word, allowing for context aware lemmatization. The result is a refined set of text data, cleaned and preprocessed to enhance the effectiveness of sentiment analysis algorithms on the given dataset.

- **Feature engineering**
  In the feature engineering section of the sentiment analysis process, additional attributes are introduced to enhance the understanding of the textual data. Initially, sentiment analysis features are incorporated using the Vader sentiment analysis tool from the NLTK module. Vader leverages a lexicon of words to assess positive and negative sentiments, considering sentence context to compute neutrality, positivity, negativity, and an overall sentiment score for each text. In addition to sentiment features, custom metrics are introduced to capture specific aspects of the textual content. Two key metrics are calculated for each text: the number of characters (nb_chars) and the number of words (nb_words). These metrics offer insights into the length and complexity of the reviews, contributing valuable information to the overall feature set. The sentiment analysis features and the newly introduced metrics contribute to a more comprehensive representation of the textual data. The result is an augmented dataset with additional dimensions that capture both the sentiment nuances and structural characteristics of the reviews.
  Next the goal is to extract meaningful numerical representations from the textual data. This process involves leveraging advanced techniques such as Word2Vec and TF-IDF to create enriched feature sets that capture the semantic relationships and importance of words. We use the Gensim module to implement Word2Vec, a technique that transforms each word in the text into a numerical vector representation. By employing shallow neural networks and considering the contexts in which words appear, Word2Vec creates vectors that exhibit similarity for semantically related words.
  These vectors serve as numerical representations for the words in the reviews. Subsequently, the Doc2Vec model is employed to transform entire reviews into numerical vectors. This process involves training the model on the text data, and once trained, applying it to the reviews to obtain their

respective representation vectors. The resulting vectors capture the semantic content of the reviews in a numerical format.

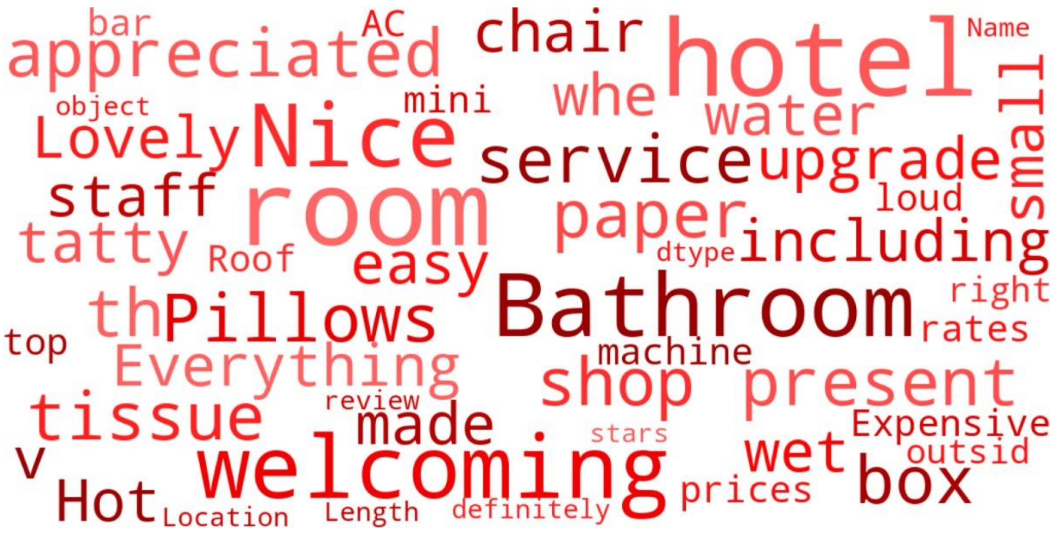
To further enhance the feature set, TF-IDF values are incorporated. TF-IDF (Term Frequency - Inverse Document Frequency) assigns values to words based on their importance in the text. It considers both the frequency of a word in a document and its rarity across all documents, providing a deep understanding of word importance. The TF-IDF values are calculated for every word and document in the dataset, enriching the feature set with information on the relative significance of words in each review.

As a result of these feature engineering steps, the dataset is augmented with numerical vectors derived from Word2Vec and Doc2Vec, as well as TF-IDF values for individual words. The resulting dataset, now with an expanded shape of (51574, 2637), indicates a substantial augmentation in terms of both the number of instances and features.

- **Exploratory data analysis**

  In the exploratory data analysis section, the focus is on gaining insights into the distribution of the target variable, "is_bad_review" and visually exploring the most frequent words in the reviews. The first part of the analysis involves examining the distribution of reviews classified as "bad" or "not bad" based on the sentiment analysis. The output reveals the proportion of negative reviews (1) compared to positive reviews (0) in the dataset.

  The results indicate a highly imbalanced dataset, because a few percentage of our reviews are considered as negative ones. Next, a Word Cloud is generated to offer a visual representation of the most prevalent words in the reviews. This graphical representation helps to identify key terms that frequently appear in the dataset, offering a glimpse into the themes and topics discussed in the reviews. The Word Cloud reveals that the majority of words are associated with hotels, such as "room" and "staff," while others delve into the customer experience with terms like "appreciated," "lovely," "loud," and "expensive."



In the exploratory data analysis section, various analyses are performed to explore deeper into the sentiment characteristics of the reviews. The process includes identifying highly positive and highly negative reviews, addressing potential issues in sentiment interpretation, and visualizing the distribution of sentiment scores among positive and negative reviews. Firstly, the focus is on extracting the highest positive sentiment reviews with more than 5 words. The selected reviews show instances where positive sentiments are expressed strongly, providing insights into the most positive aspects of the dataset. Recognizing potential challenges in sentiment analysis, the analysis acknowledges that the most negative reviews may contain errors. Vader, the sentiment analysis tool, can sometimes misinterpret words like 'no' or 'nothing' as negative when used to indicate the absence of problems with the hotel. This awareness underscores the need for a deep interpretation of sentiment results. Next, the lowest negative sentiment reviews with more than 5 words are identified, shows instances where negative sentiments are most pronounced.

At the end of this section, to visualize the sentiment distribution, a plot is generated showing the distribution of sentiment scores for positive and negative reviews. The graph reveals that Vader tends to categorize most good reviews as highly positive, while bad reviews typically exhibit lower compound sentiment scores. This visual representation underscores the significance of the sentiment features computed earlier in the modeling process, emphasizing the impact of sentiment characteristics on the overall analysis.

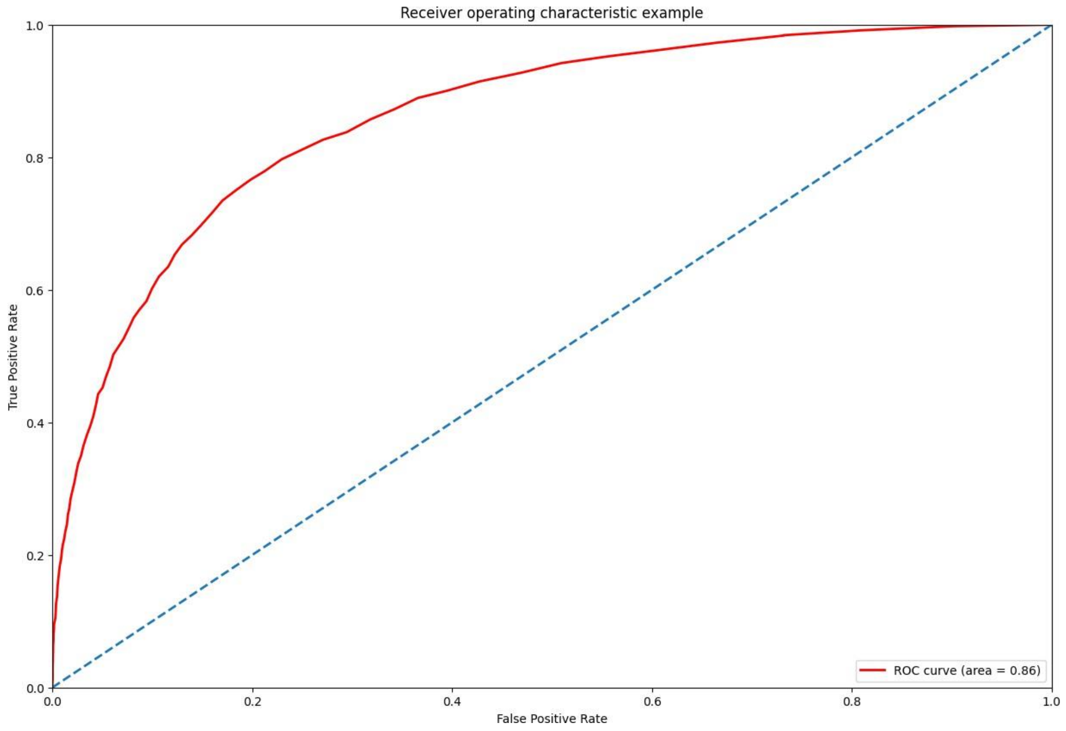
In summary, the exploratory data analysis section provides a deeper understanding of sentiment in the dataset, highlighting extreme positive and negative cases, addressing potential challenges, and visually depicting the sentiment distribution among positive and negative reviews.

- **Model**

  In the model section, a Random Forest (RF) classifier is employed for predicting reviewer scores based on the features extracted from the sentiment analysis and other relevant information. The classifier is trained on a subset of the dataset, and its feature importance are analyzed to understand the factors influencing the model's predictions. The computed feature importances reveal that sentiment analysis features, vector representations of the texts, and specific words play crucial roles in the model's performance. The importance of these features highlights their significant contribution to the predictive power of the Random Forest model.

ROC Curve

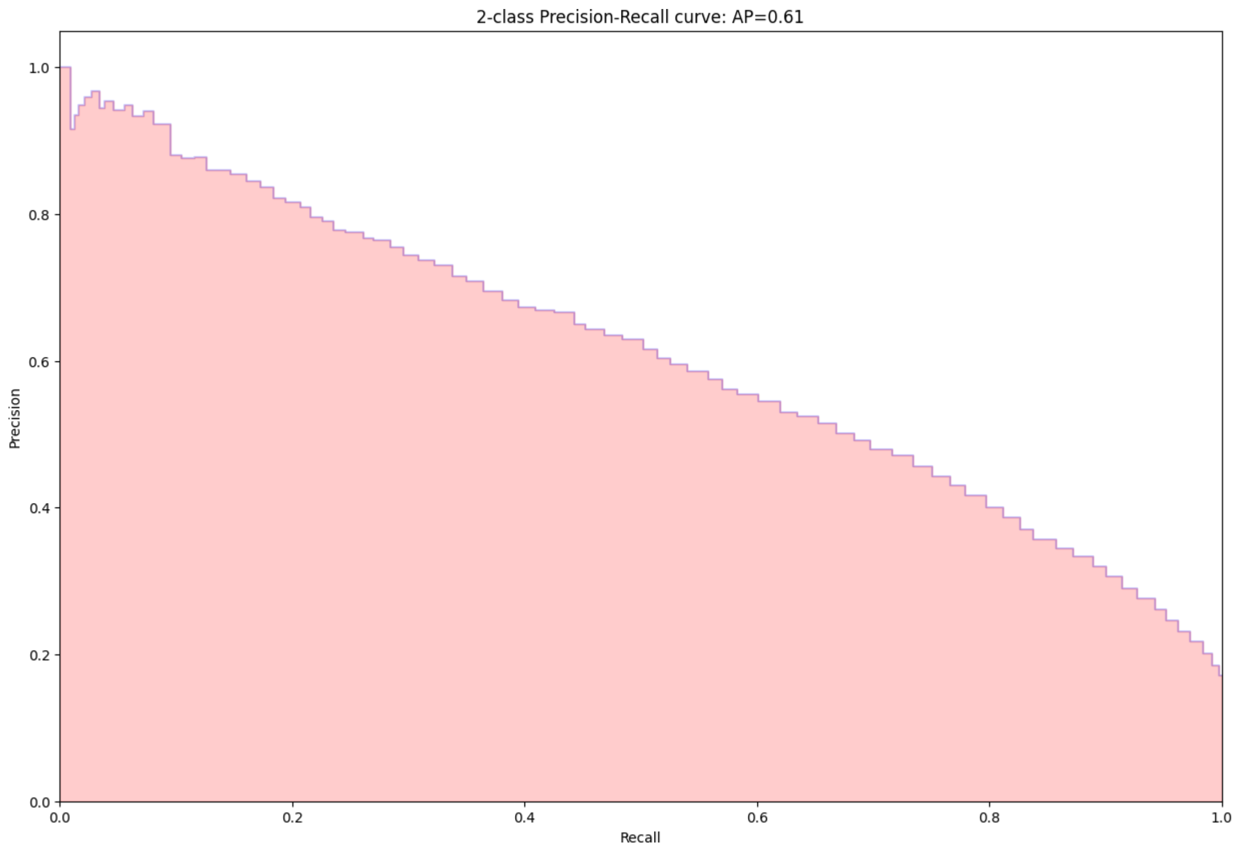
To assess the overall quality of the classifier, a Receiver Operating Characteristic (ROC) curve is generated. The ROC curve provides insights into the trade-off between true positive and false positive rates across different classification thresholds. The curve's area under the curve (AUC ROC) is calculated as a metric summarizing the classifier's performance. However, it is noted that the imbalanced nature of the dataset can impact the interpretation of ROC curves, especially in cases where the number of negative instances (good reviews) is significantly higher. In such situations, the Area Under the Curve Precision-Recall (AUC PR) or Average Precision (AP) is suggested as a more appropriate metric. This metric considers the precision-recall trade-off, providing a more accurate reflection of the classifier's performance, particularly in imbalanced datasets.



PR Curve

In the modeling section focusing on the reviewer's score, the precision recall curve is generated and analyzed using the Area Under the Curve Precision-Recall (AUC PR) metric, also known as Average Precision (AP). This evaluation metric is deemed more appropriate for imbalanced datasets where the negative class (good reviews) dominates.

The precision-recall curve visually illustrates the trade-off between precision and recall at different prediction thresholds. The curve depicts how precision decreases with an increase in recall, emphasizing the need to choose a prediction threshold that aligns with specific goals. Depending on the desired balance between precision and recall, adjusting the prediction threshold allows for customization of the model's behavior. If prioritizing high recall is crucial, a low prediction threshold should be selected to detect most positive observations, even at the cost of lower precision. Conversely, if high precision is desired with less emphasis on recall, a higher prediction threshold can be chosen.



In order to know if our model performs better than another classifier, we can simply use the AP metric. To assess the quality of our model, we can compare it to a simple decision baseline. For example, take a random classifier as a baseline that would predict half of the time 1 and half of the time 0 for the label.

Such a classifier would have a precision of 4.3%, which corresponds to the proportion of positive observations. For every recall value the precision would stay the same, and this would lead us to an AP of 0.043. The AP of our model is approximately near 0.40, which is more than 8 times higher than the AP of the random method. This means that our model has a good predictive power.

Recommender System (Location)

This part of our project is dedicated to developing a recommender system that gives users hotel recommendations whenever given a destination.

The main idea for this recommender system comes from our dataset containing user reviews for some hotels. Each of these users have rated the hotel from scale 0 to 10. An overall rating for each hotel can be calculated using the "mean" method for review scores.

One of the most important points about this recommender system is that it should only recommend hotels that actually exists in the users' destination. But how can we figure out if the location of a hotel and users' destination are the same?

One of the good ways for finding that out is to extract the country of hotels out of their address (which is provided by our dataset).

The next step is computing the average rating of the hotel by getting the mean of all ratings for each hotel. So, we group the rows of our data frame by the hotel names and then calculate the average of review scores for each hotel and store it a new column 'average_rating'.

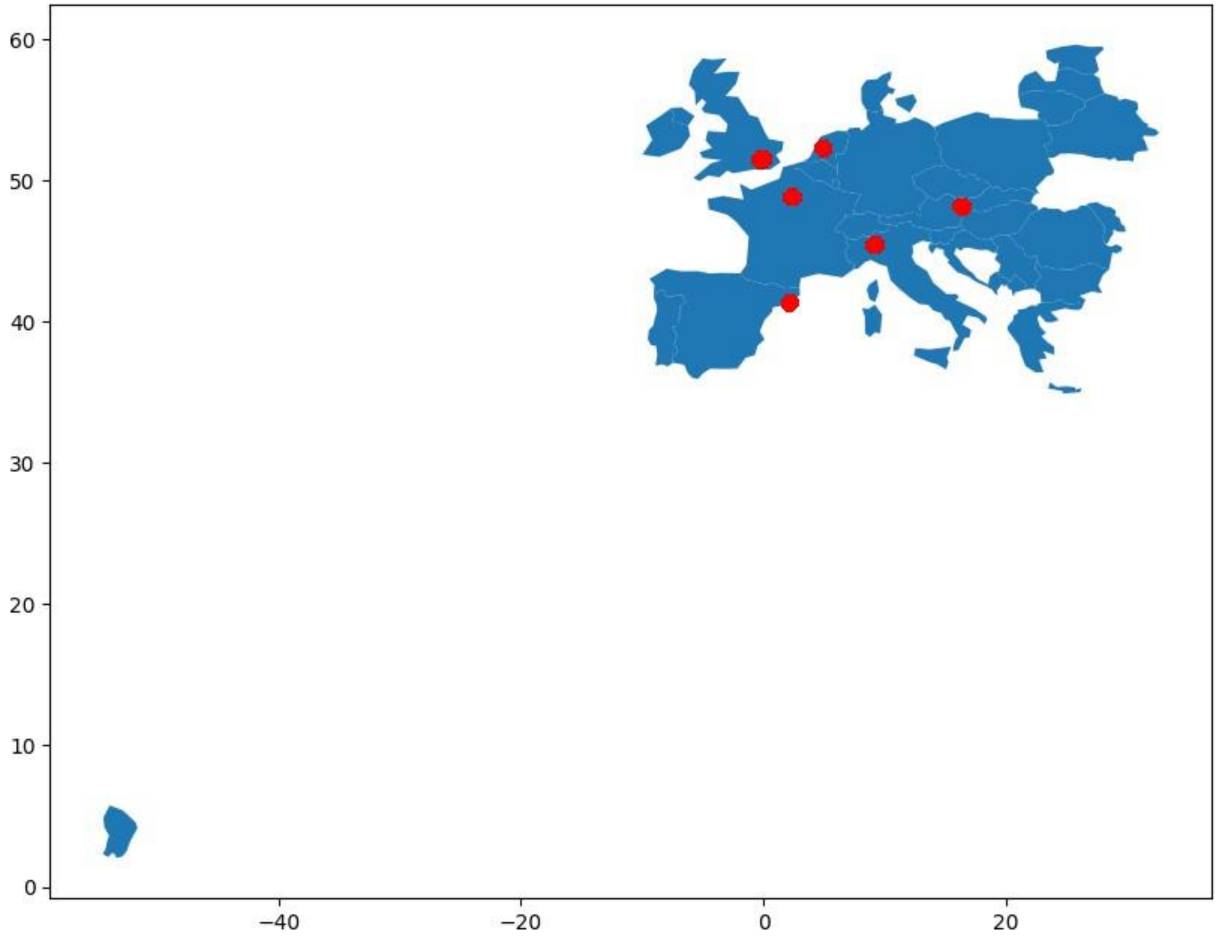
In the final step, we define two functions. One for getting input from user and one for getting the result (recommendations) based on the user's input.

We've provided you an example for the given input and corresponding output.

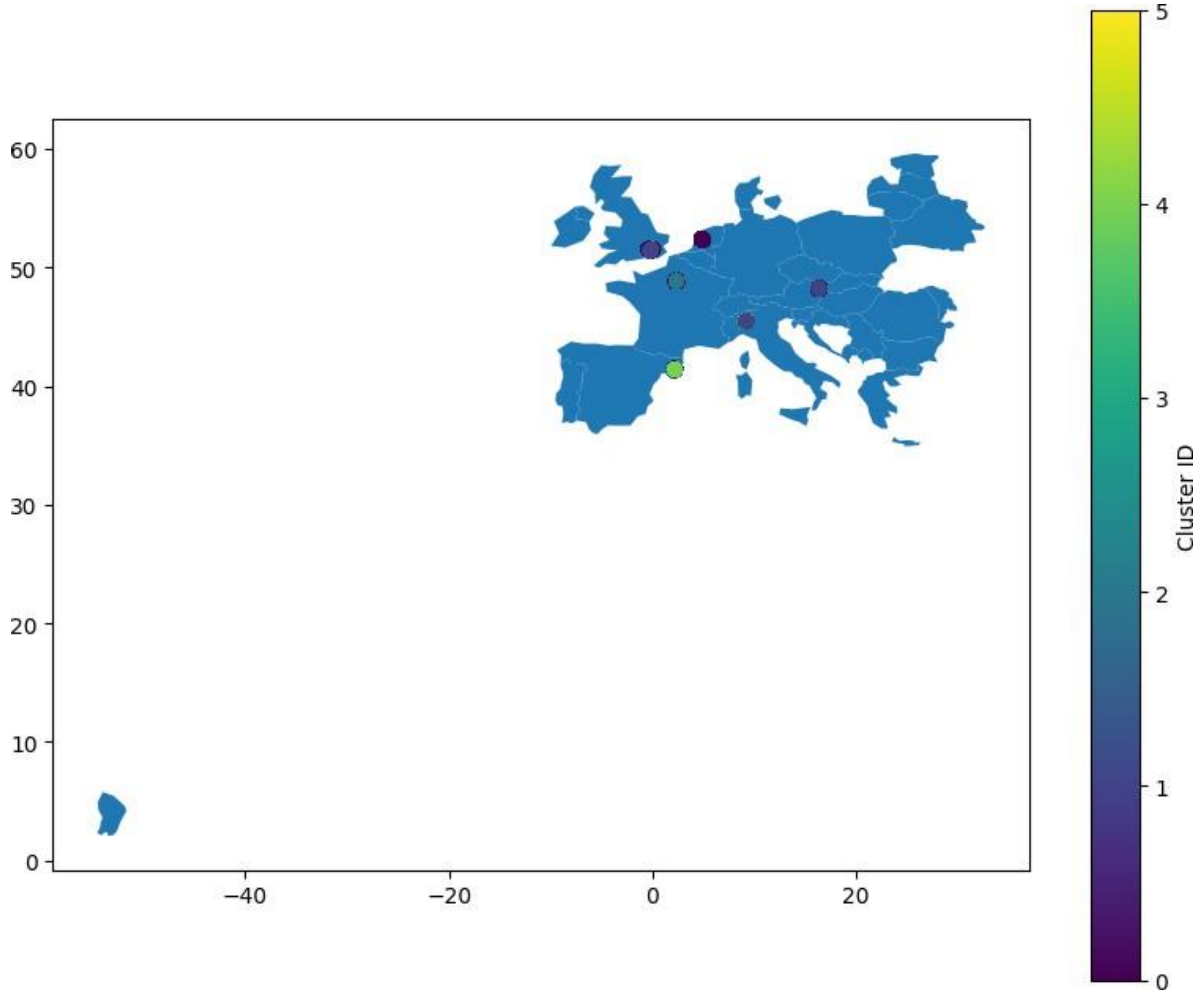
Clustering

In the clustering analysis section, the geographic locations of hotel reviews are utilized to group them into distinct clusters. The process begins by plotting the locations of the reviews on a map of Western Europe. The dataset is filtered to include only Western European countries, and the longitude and latitude coordinates of the hotel reviews are added to the map.

Observing the map, it becomes evident that the reviews are distributed across six different regions: **Amsterdam, NL. Barcelona, ESP. London, UK. Milan, ITLY. Paris, FR. Vienna, AUST.** To formalize these groupings, a K-means clustering analysis is performed with six clusters. The K-means algorithm groups the reviews based on their geographic proximity, and the resulting clusters are visualized on a map.



The plot illustrates the successful clustering of the 512,470 reviews into six distinct regions. Each cluster is represented by a unique color, and the close-up map showcases the geographic distribution of these clusters. The identified clusters correspond to the predetermined regions: Amsterdam, Barcelona, London, Milan, Paris, and Vienna.



The K-means clustering algorithm effectively organizes the hotel reviews into spatially coherent groups, providing valuable insights into the geographic distribution of sentiments in Western European cities. The distinct clusters on the map reflect the success of the clustering process and offer a visual representation of the regional patterns in the dataset.

Discussion

- **Why is using the mentioned methods in the Text cleaning useful?**
  Lowercasing: Applying this method on our text will actually prevent the model to distinguish words like “BaD” and “bad”, that are in fact the same word with different looks.
  Punctuation Removal: In the context of sentiment analysis we don’t really need to take punctuation marks into account so, it’s essential to remove them.
  Tokenization: Tokenizing our text’s words will help us to represent each word as a token. We can think about this part as a kind of encoding.
  Stop Word Removal: Removing all of the frequent or rare words will help us to enhance the performance of our analysis. The reason is that words like “the” or “that” that may appear so many times in texts don’t really have a sentimental value or worth for us and they’re all about the grammar of the language. So not only removing them would cause us no harm, but also it’s necessary to do so.
  Lemmatization or Stemming: This task again helps a potential model to identify words like “running” and “run” as the same thing. This is a critical subject in tokenizing text.
  Removing Words with Numbers: In our sentiment analysis numbers will only get in our way so, we just remove the words that contain numeric parts.

- **How is TF-IDF helping us in sentiment analysis?**
  TF-IDF is a technique used in natural language processing to weigh the importance of words in a document. TF-IDF can be used to identify the most important words in a document that contribute to its sentiment. By giving more weight to words that are rare in the corpus but frequent in a particular document, TF-IDF can help to identify the words that are most indicative of the sentiment of the document.

- **How does Random Forest Classifier act in sentiment analysis?**
  In the context of sentiment analysis Random Forest Classifiers can be trained on a dataset of text labeled with their sentiment (positive, negative, or neutral) to predict the sentiment of new, unlabeled text.