Hotel Recommendation System

INTRODUCTION

Project Agenda: Given a hotel I am interested in and the comments, how should I anticipate myvisit?

When planning for a trip, selecting a hotel is extremely challenging due to the number of hotels available and the amount of relevant information to compare. With so many factors hidden in the text of written reviews and hundreds or thousands of reviews per hotel, it is unrealistic to expect a user to be able to make a truly educated decision about their hotel choice. Since the specific amenities, services, and locations offered by a hotel are often important to the quality of a trip, it is highly important for the user to be able to access this information in a format which allows them to select the ideal hotel with minimal effort to improve the likelihood of a successful trip and minimize stress and planning fatigue. For hotels, customer reviews matter, as it brings customer loyalty, which will improve sales of hotels as well. For this reason, hotels should observe reviews given by customers.

Due to the value of customer reviews to both customer and hotel, customer reviews have become a great research topic in all industries including the hotel industry. Also, for researchers like us, I got a dataset where I can do different analyses on text data (in this case it is reviews) like sentiment analysis observing sentiments of the customers while writing reviews. Proposing a recommendation system which tourists can use to anticipate their visit and selecting the best hotel for them.

Data Description

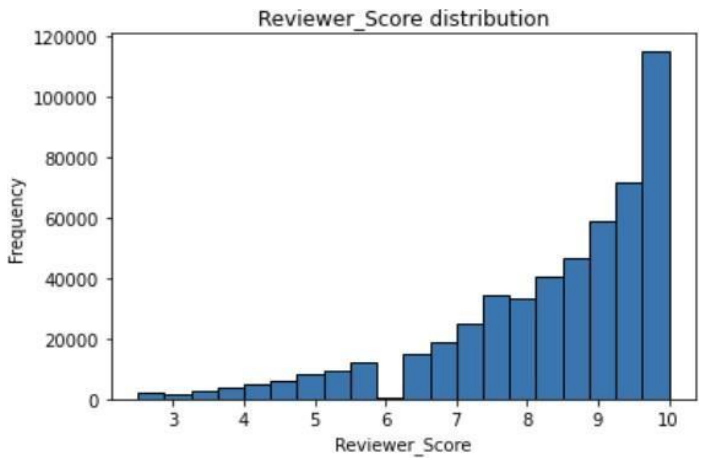
The data was scraped from Booking.com and it has customer reviews and scoring of 1,493 luxury hotels across Europe. I have 515,738 rows and 17 columns in the dataset. There are 3,268 missing values in the lat and lng columns, and all other data is present. The columns are as follows.

Column Name	Description
Hotel_Address	Address of hotel
Additional_Number_of_Scoring	There are also some guests who just scored on the service rather than a review. This number indicates how many valid scores without review there.

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.
Total_Number_of_Reviews	Total number of valid reviews the hotel has.
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.
Total_Number_of_Reviews_Reviewer_Has_Given	Number of Reviews the reviewers has given in the past.
Reviewer_Score	Score the reviewer has given to the hotel, based on his/her experience
Tags	Tags reviewer gave the hotel. Tags require more in depth analysis for a responsible evaluation. The Tags column contains data from the user's trip, including the following: 1. Trip type 2. Group type/size 3. Room type 4. Trip duration
days_since_review	Duration between the review date and scrape date.
lat	Latitude of the hotel
lng	Longitude of the hotel

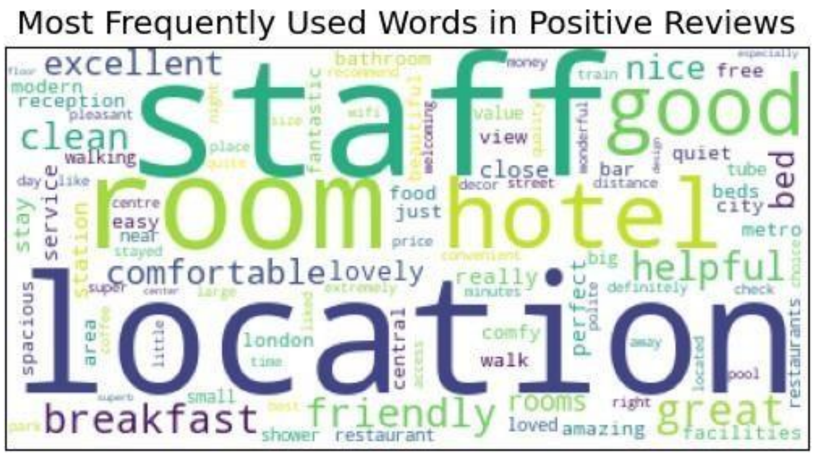
<u>EXPLORATORY DATA ANALYSIS</u>

<u>Review Score Distribution</u>



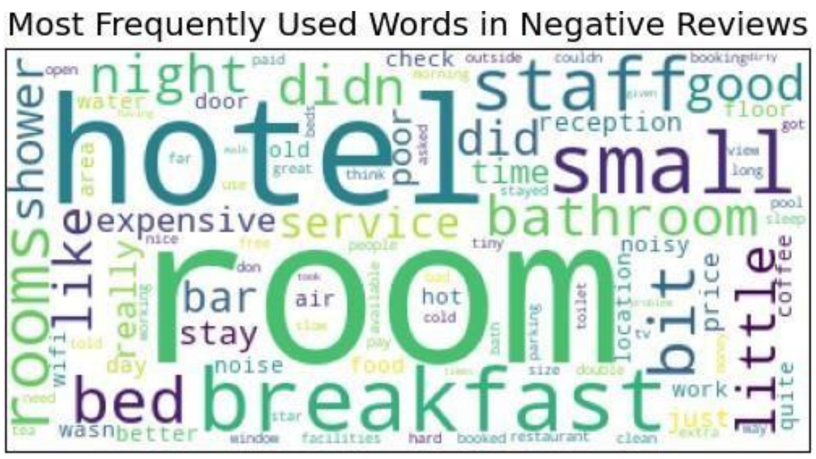
Reviewer score is left skewed. Most hotels are highly rated likely because the data only contains luxury hotels. In other words, most reviewers have a positive experience for these hotels. This distribution also serves as a reminder that there may be an imbalanced situation in sentiment analysis that the number of positive reviews may be much higher than the number of negative reviews.

<u>Word Cloud for Positive Reviews</u>



From the word cloud for positive reviews, I see that some of the most commonly used words in positive reviews are “staff”, “location”, “room”, “hotel”, and “good”. This mainly aligns with intuition on the words that we would expect to see in positive reviews about hotels.

<u>Word Cloud for Negative Reviews</u>

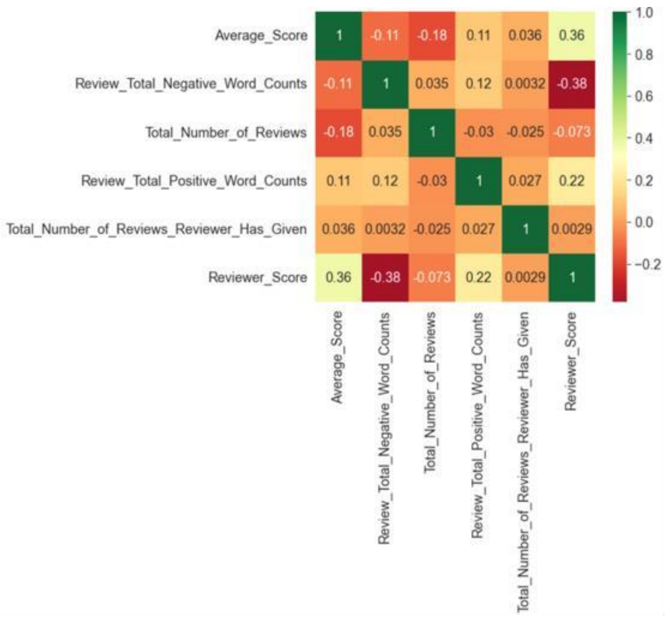


In the negative review word cloud, we see the most common negative words contain “room”, “hotel”, “small”, and “breakfast”, and “staff”. Notably, we see many of the same words appear frequently in both the positive and negative reviews, such as “room” and “staff”. This indicates the importance in understanding the review context, so we can better understand the impact of the word to the review.

CHOICE OF FEATURE, TARGET, AND FEATURE SELECTION

Justification

As our analysis revolves around the reviews of a hotel, it made intuitive sense for the target to be the review score variable. This is because, as the project agenda states, our goal is to quantify anticipated hotel visits. The review score allows us to determine how well the visit went.



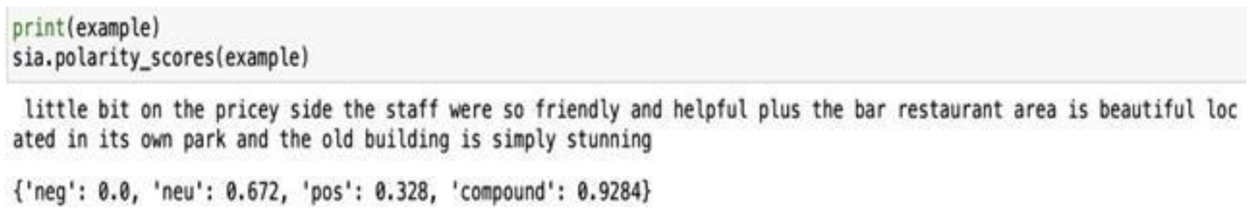
The above plot is a correlation matrix, indicating the association of columns within our data. Because of the lack of correlation between our target and other features, it was clear that the unprocessed data was inadequate in answering the project agenda . Thus, I decided to do feature engineering and create new features using sentiment analysis.

VADER Sentiment Analysis

Sentiment analysis is a computational study related to people's opinions, sentiments, emotions, and attitudes. Millions of comments generated by social media and review websites have provided large amounts of resources for sentiment analysis. However, it is hard for information in these contents to be extracted and used directly. Text data is a form of unstructured data. Unstructured data is difficult to understand using traditional ways since it is not organized in a defined manner. Compared to organized data stored in a formal database, text data includes more irregularities and ambiguity. In this case, sentiment analysis is an efficient way to interpret data and extract opinions from it.

VADER (Valence Aware Dictionary for Sentiment Reasoning) is a model used for text sentiment analysis that is sensitive to both polarity (positive/negative) and intensity (strength) of emotion. It is available in the NLTK package and can be applied directly to unlabelled text data. VADER sentiment analysis relies on a dictionary that maps lexical features to emotion intensities known as sentiment scores. The sentiment score of a text can be obtained by summing up the intensity of each word in the text. VADER not only talks about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

The compound score is a metric that calculates the sum of all the lexicon ratings which have been normalized between -1(most extreme negative) and +1 (most extreme positive).
positive sentiment: (compound score >= 0.05)
neutral sentiment: (compound score > -0.05) and (compound score < 0.05)
negative sentiment: (compound score <= -0.05)



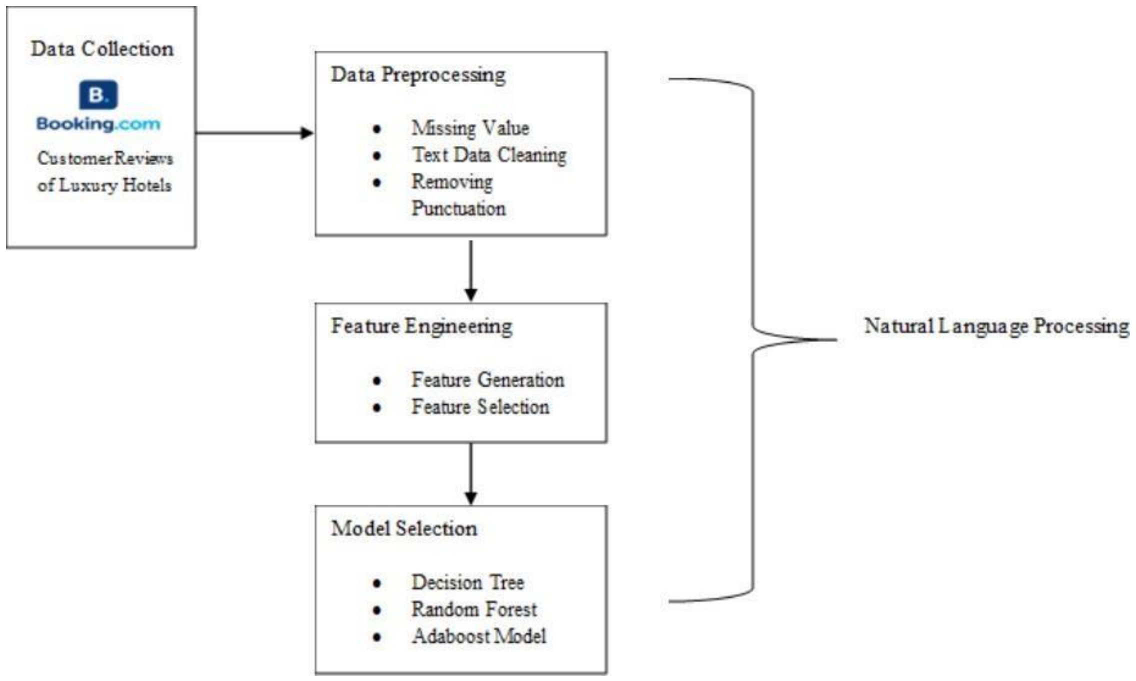
```
print(example)
sia.polarity_scores(example)
```

```
 little bit on the pricey side the staff were so friendly and helpful plus the bar restaurant area is beautiful loc
ated in its own park and the old building is simply stunning

{'neg': 0.0, 'neu': 0.672, 'pos': 0.328, 'compound': 0.9284}
```

Model Pipeline Architecture

Raw text data cannot be used for sentiment analysis directly since segments such as punctuations and numbers in textual data would damage the analysis result. In this case, the data needs to be pre-processed. I have used positive_reviews and negative_reviews and combined them as Total_reviews. Then, I used NLTK to deal with data cleaning and tagging, such as removing punctuation. I utilized Total_reviews to run sentiment analysis and then create polarity scores.



<u>Choice of Feature and Target</u>

Feature variable - polarity scores returned by running sentiment analysis on Total_reviews as features.
Target – Reviewer_Score (as explained above)

MODEL

Decision Tree Regressor

The Decision Tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete. In other words, it is not represented just by a discrete, known set of numbers or values.

Random Forest Regressor

The Random Forest algorithm combines ensemble learning methods with the decision tree framework to create multiple randomly drawn decision trees from the data, averaging the results to output a new result that often leads to strong predictions/classifications.

Adaboost Regressor

The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction.

Model Rational

I could have used other regression models, but since the size of the dataset is quite large, I preferred the decision tree model over other regression models. Also, since decision trees are prone to overfitting, I also used Random Forest and Adaboost to see if they perform better than Decision Tree.

Assumptions

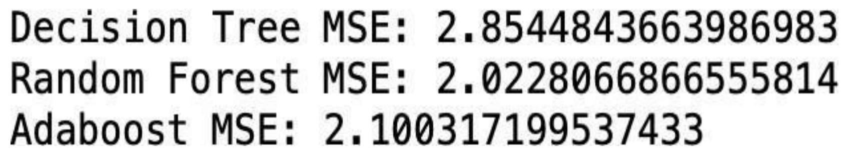
Since tree-based modeling is a non-parametric approach, it makes no assumptions of the training data or prediction residuals, e.g., no distributional, independence, or constant variance assumptions.

MODEL SELECTION

Metric used to compare models

Mean squared error (MSE) is the average of the summation of the squared difference between the actual output value and the predicted output value. Our goal is to reduce the MSE as much as possible.

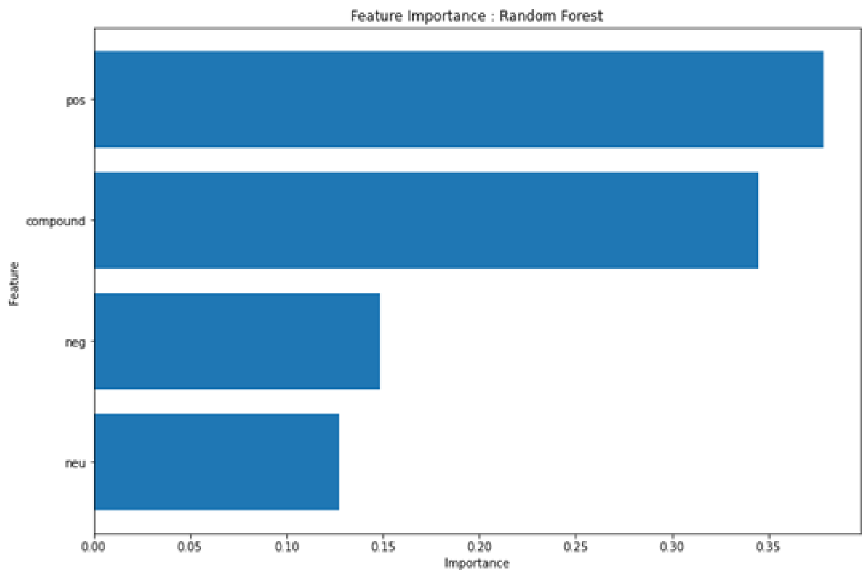
MSE values for all three models



```
Decision Tree MSE: 2.8544843663986983
Random Forest MSE: 2.0228066866555814
Adaboost MSE: 2.100317199537433
```

Comparing MSE values I saw, Random Forest is performing best among all three models. Apart from having low MSE, Random forests are generally more accurate than individual decision trees because they combine multiple trees and reduce overfitting, providing better predictive performance and robustness.

Using Random Forest, I also plotted Feature importance of all the feature variables and observed “pos” polarity score as most important feature, reason can be because since data is skewed and we mostly have more positive reviews than negative reviews.



Model Example

Example 1 -
Review: “Easy access to San Diego main areas. Nice to have good coffee in lobby & brown bag breakfast. Nice staff.”
Vader Sentiment Scores: {'neg': 0.0, 'neu': 0.528, 'pos': 0.472, 'compound': 0.8885}
Reviewer score for example 1 using random forest regressor: [8.77382355]

Example 2 -
Review: “room was dirty and i was afraid to walk, and bathroom tile floor could have been cleaner. Spots/general soiling on the chairs and the tile was dirty, along with hair and dirt in the corners.”
Vader Sentiment Score: {'neg': 0.206, 'neu': 0.752, 'pos': 0.043, 'compound': -0.7579}
Reviewer score for example 2 using random forest regressor: [5.50503333]

FACTOR ANALYSIS

Purpose of Factor Analysis

To better recommend hotels to future guests, it must first be understood what the guest is most interested in. While the original dataset provides a baseline score for each hotel, it does not include any context to why a hotel is scored highly, a necessary component to effectively match hotels with new travelers given their interests. The following factor analysis allows us to provide additional scoring categories within five key areas of a hotel beyond the baseline score, thus creating a stronger recommendation system where the user can select a hotel based upon the factors that they are most interested in as we will see later in the paper.

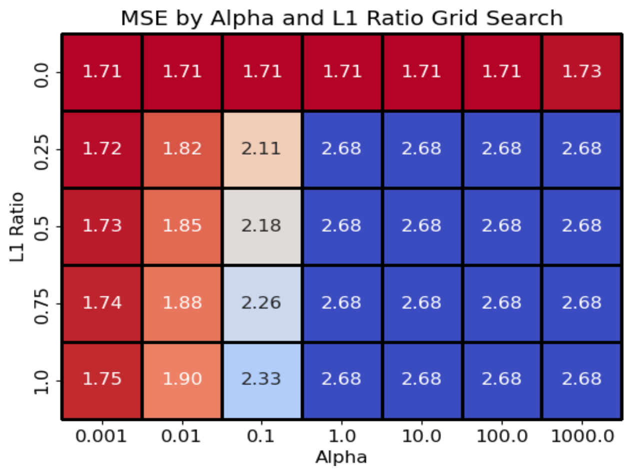
Data Preprocessing and Format

For this factor analysis, our model takes as input lemmatized nouns from positive and negative hotel reviews. By extracting nouns, we theoretically are left with all factors within the review. The lemmatization process helps promote consistency within the word structure of these factors. The data is then converted into binary bag-of-words format, where each factor within its given review context represents a feature of the model, and each value being a binary indication of whether the feature appeared in the review. To omit uncommon phrases, only factors appearing in at least 0.1% of reviews are included. With this input, we estimate the review score.

Model Selection

In building the model, elastic net, lasso, and ridge strategies were compared to find the best performance on out-of-sample data. These modeling algorithms were selected because they allow for easily interpretable feature values, perform feature selection via the L1 penalty in the case of lasso and elastic net, and handle multicollinearity via the L2 penalty in the case of ridge and elastic net. While they do introduce bias in the coefficient estimates, I elected to use these strategies over a multiple linear regression model due to the high dimensionality of our data.

To find the optimal parameters, a grid search was performed. This allowed numerous models to be compared throughout the exploration for the parameters that best performed on out-of-sample data. The search helped select an optimal value of alpha from 0.001 to 1,000 and l1 ratio between 0 and 1. When the l1 ratio was 0 or 1, a ridge model or lasso model was specified instead of elastic net, respectively. 3-fold cross validation was used to assess the mean squared error on out-of-sample data. The performance of each model in the grid search is visible with the following heatmap.

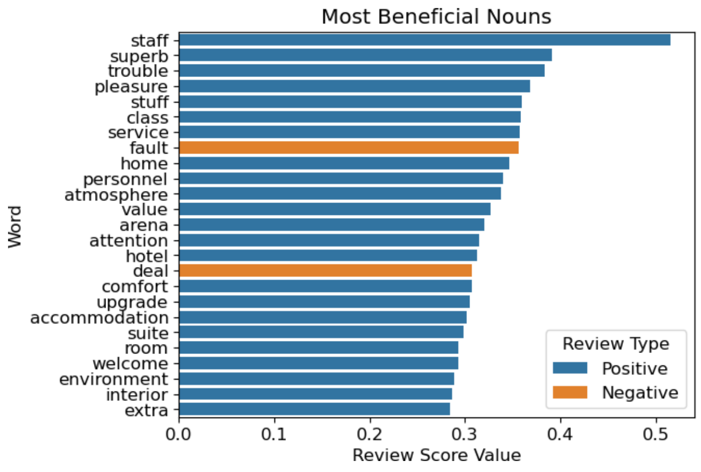


L1 Ratio \ Alpha	0.001	0.01	0.1	1.0	10.0	100.0	1000.0
0.0	1.71	1.71	1.71	1.71	1.71	1.71	1.73
0.25	1.72	1.82	2.11	2.68	2.68	2.68	2.68
0.5	1.73	1.85	2.18	2.68	2.68	2.68	2.68
0.75	1.74	1.88	2.26	2.68	2.68	2.68	2.68
1.0	1.75	1.90	2.33	2.68	2.68	2.68	2.68

Upon completion of the grid search, the chosen parameters were a ridge model with alpha of 100, obtaining an out-of-sample mean squared error of 1.71, a very slight improvement over other tested ridge models. In comparison, the best elastic net model used alpha of 0.001 and l1 ratio of 0.25 to achieve an out-of-sample mean squared error of 1.72, and the best lasso model used alpha of 0.001 to achieve an out-of-sample mean squared error of 1.75. The ridge model was then recompiled on the entirety of the dataset.

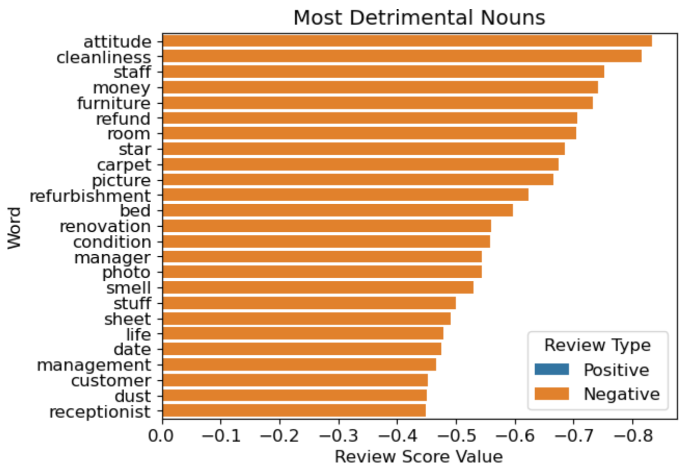
Results

With the created model, it is possible to see which hotel traits are most valuable in increasing and decreasing a reviewer's hotel score. More specifically, we can use the model coefficients to see, on average, how much a noun is expected to increase or decrease the reviewer's score, provided all other elements of the review are held constant. The chart presented below showcases the nouns that exhibit the most substantial expected increase in the score given the specified type of review.



From the plot, it becomes evident that "staff" is the most beneficial noun with a coefficient value of 0.52. Statistically, it can be expected that the reviewer score increases by 0.52 on average if "staff" is mentioned in the positive section of the review, provided that the rest of the review remains constant. This finding aligns with intuitive expectations, as a positive interaction with hotel staff generally translates to a higher review score.

In examining the opposite end of the spectrum, we can gain valuable insights into the least desirable traits of a hotel by exploring the most detrimental nouns. The plot provided below presents these nouns, shedding light on areas where improvement is critical for enhancing guest satisfaction.



At the forefront of detrimental nouns is "attitude" (-0.83). This underscores a clear message that hotel guests greatly disvalue a poor attitude from employees, emphasizing the crucial role of staff demeanor in the overall guest experience.

Use of Factor Analysis in Recommendation System

It is important to note the value that this factor analysis has toward our recommendation system, outlined in the upcoming section of our paper. Notably, the original data only provides an overall score for each hotel. However, many travelers weigh different factors of a hotel differently, thus the overall score may not always be representative in how a future guest can anticipate their visit. For example, a traveler on a business trip may care more about the comfort level of the hotel, so they can rest comfortably before an important meeting. Conversely, a family on vacation may have more interest in finding a hotel with a lively environment.

By using the most important traits as found in the factor analysis, we can assign each hotel additional scoring for service, comfort, cleanliness, environment, and value. These scores are all compiled based upon the frequency in which reviews positively and negatively mention the key words found in the factor analysis. Specifically, the service score corresponds to the mentions of hotel employees and their attitude. The comfort score is based upon the perception of valuable room amenities, such as bedding. The cleanliness score is compiled on the clean nature and condition of the hotel. The environment score is judged upon the nature and atmosphere of the hotel and surrounding area, including amenities outside of the hotel room. The value score is computed according to the monetary factors of the hotel.

With these additional scoring categories of service, comfort, cleanliness, environment, and value, we have provided resources in our recommendation system where a traveler can choose a hotel based upon the traits that matter most to them. For example, a user may decide to stay at a hotel with a lower overall score but a higher score in the value category because they are mainly looking to get the most out of their spendings. This analysis, used in combination with the upcoming recommendation system, helps answer the agenda as it allows a user to decide exactly what features and traits that they are most interested in, choosing a recommended stay based on their desired visiting experience.

<u>**RECOMMENDATION SYSTEM**</u>

<u>Design Approach</u>

Building a user-centric hotel recommendation system should focus on understanding and fulfilling the unique needs of each traveler. To achieve this, three key questions were considered to design the recommendation function:

1. What is most important to the user?

Prioritizing the user's needs is paramount. This involves understanding their travel motivations, budget constraints, and desired experiences. Are they seeking a luxurious getaway, a family-friendly adventure, or a budget-conscious city break? Do they prioritize amenities, proximity to specific attractions, or hygiene? As part of our factor analysis, I have created additional scoring categories for the hotel in the key categories of service, comfort, cleanliness, environment, and value. Thus, a user of our recommendation system can choose the best hotel option given their specific interests.

2. How do we use our data to filter recommendations?

The system leverages different data features from the dataset to personalize recommendations. This includes hotel information (amenities, location, reviews), preferences and positive factors score. The system then identifies users with similar preferences and recommends the top five hotels they might enjoy.

3. What kind of input can be handled?

A user-friendly system should handle diverse types of input. Comparable to a Google search, our recommendation system takes explicit user preferences (e.g., type of trip, location) along with textual inputs such as travel descriptions and keywords. Using natural language processing, we extract hidden preferences and identify trends, outputting the best hotel option given the interests of the user. Additionally, this versatility of the function makes the hotel search process hassle free and more personalized.

By prioritizing user needs, leveraging data effectively, and accommodating diverse input formats, a well-designed hotel recommendation system can provide personalized suggestions and simplify the travel planning journey for every user.

<u>Hotel Recommendation System Flowchart</u>

1. **Input**:
    - Destination (Country)
    - Trip Type (Group, Room, Length of Stay)

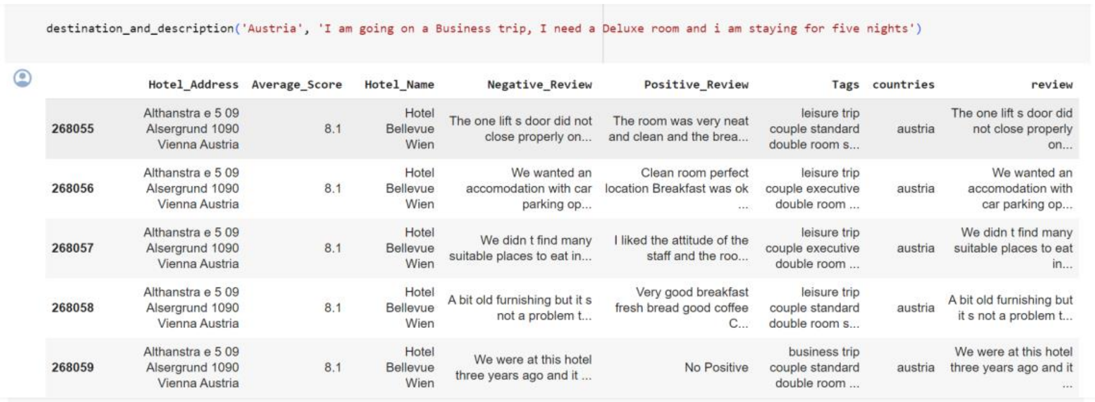
2. **Process**:

- Analyze user inputs – tokenize, lemmatize, save as a word vector
- Access hotel data (location, features, reviews)
- Calculate average overall review score for each hotel (matching the word vector with words in Tags column)
- Calculating Jaccard Intersection Score
- Filter hotels based on user preferences (optional)
- Sort hotels based on factor analysis scoring categories (optional)

3. **Output**:

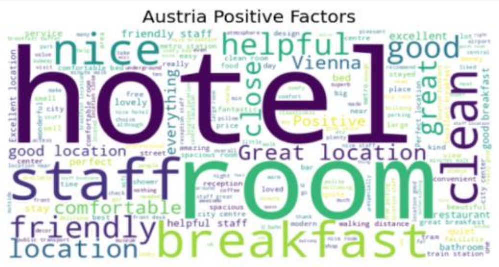
- Top 5 Best Hotels:
    - Hotel Address
    - Hotel Name
    - Negative and Positive Review
    - Tags
    - Review
    - Average Score
    - Word Cloud: Visualization of key positive and negative factors for the top hotel.

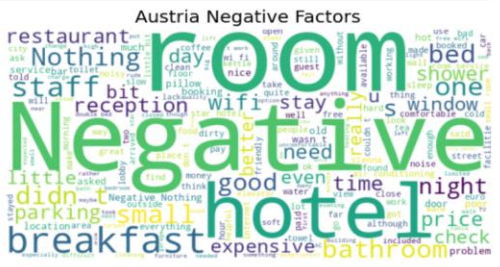
<u>Sample Outputs</u>



```
destination_and_description('Austria', 'I am going on a Business trip, I need a Deluxe room and i am staying for five nights')
```

	Hotel_Address	Average_Score	Hotel_Name	Negative_Review	Positive_Review	Tags	countries	review
268055	Althanstra e 5 09 Alsergrund 1090 Vienna Austria	8.1	Hotel Bellevue Wien	The one lift s door did not close properly on...	The room was very neat and clean and the brea...	leisure trip couple standard double room s...	austria	The one lift s door did not close properly on...
268056	Althanstra e 5 09 Alsergrund 1090 Vienna Austria	8.1	Hotel Bellevue Wien	We wanted an accomodation with car parking op...	Clean room perfect location Breakfast was ok ...	leisure trip couple executive double room ...	austria	We wanted an accomodation with car parking op...
268057	Althanstra e 5 09 Alsergrund 1090 Vienna Austria	8.1	Hotel Bellevue Wien	We didn t find many suitable places to eat in...	I liked the attitude of the staff and the roo...	leisure trip couple executive double room ...	austria	We didn t find many suitable places to eat in...
268058	Althanstra e 5 09 Alsergrund 1090 Vienna Austria	8.1	Hotel Bellevue Wien	A bit old furnishing but it s not a problem t...	Very good breakfast fresh bread good coffee C...	leisure trip couple standard double room s...	austria	A bit old furnishing but it s not a problem t...
268059	Althanstra e 5 09 Alsergrund 1090 Vienna Austria	8.1	Hotel Bellevue Wien	We were at this hotel three years ago and it ...	No Positive	business trip couple standard double room ...	austria	We were at this hotel three years ago and it ...





CONCLUSION

Summary

I utilized positive_reviews and negative_reviews features. After rigorous data cleaning and preprocessing, I combined them into a single total_reviews column. Further, I performed sentiment analysis on total_reviews to generate polarity scores. As part of feature engineering, I utilized the newly generated polarity scores as our features and Reviwer Score as our target variable. We fitted Decision Tree, Random Forest and Adaboost and saw Random Forest performed best overall.

Also, to answer our project agenda in a more practical way, we built a recommendation system which can be utilized by tourists to anticipate their visit. To do so, we offered analysis of the most important factors mentioned in hotel reviews, backed by machine learning. We then used the key factors in reviews to assign each hotel additional scoring for service, comfort, cleanliness, environment, and value, further allowing a traveler to select a hotel based on their interests. Finally, we used the key categories as found in our factor analysis system along with natural language processing to create a recommendation system. This recommendation system operates comparably to a Google search of hotels, and in turn outputs the hotel offering the best anticipated visit given the specific interests of the traveler.

Limitations

With the conclusion of our analysis, it is worth mentioning that a sole source of data was limited in precision and reliability. As Booking.com is one of the most popular hotel review websites, we speculate that fake and paid online reviews could not be fully avoided, possibly evident by most reviews being positive. For example, it is plausible that some customers were asked to write good reviews on Booking.com to get a discount. This is a prevailing situation since hotels care for the customer evaluations online, and potential customers will check those reviews to make decisions. In future work, data from Booking.com, TripAdvisor, and other customer review websites could be used fortraining the model. If the model was trained by multiple data sources, the results of our analysis could bemore reliable and trustworthy. Also, since the target hotels of this model were luxury hotels in Europe, themodel might not work as well for budget hotels, mid-range hotels, or hotels in other continents.

Future Scope

For future studies, a system could be created from the viewpoint of hotel management as one end-user and tourist as other, so that both parties can utilize the system in a more robust and practical way. Another potential idea would be the implementation of a recency bias to better showcase the status of a hotel.