PREDICTING AIRBNB UNLISTING

Text Mining
Project Report

- 2022 / 2023 -

Daila Alexandre 20191182
Diogo Silva 20221393
Luís Fernandes 20221649

Professors:
Bruno Jardim
Inês Rodrigues

Index

1. Introduction

This project focuses on the analysis and prediction of property listings on Airbnb, utilizing the techniques of Text Mining and Natural Language Processing (NLP). The objective is to explore and preprocess two datasets, namely 'train' and 'train_reviews,' followed by training various models to predict the unlisted status of properties in the test dataset.

With real-world data from Airbnb properties, this project aims to harness the power of NLP models to anticipate whether a property listed on Airbnb will be unlisted in the upcoming quarter.

The project involves working a corpus with multiple datasets, including the 'train' and 'train_reviews', which are subjected to data exploration and preprocessing. The exploration phase is intertwined with preprocessing steps to ensure a comprehensive analysis of the data. Through the application of specific functions and techniques, the data is transformed and prepared for model training.

The prediction task revolves around utilizing the trained models to make informed judgments on the unlisted status of properties in the test dataset. By leveraging the power of NLP and the insights gained from the exploration and preprocessing stages, this project aims to deliver accurate predictions for property unlisting.

By delving into the intricacies of the Airbnb dataset and employing advanced NLP models, this project seeks to uncover patterns, extract valuable insights, and provide a reliable means of predicting property unlisting. The resulting analysis and predictions hold the potential to assist stakeholders in making informed decisions within the Airbnb ecosystem.

2. Data Exploration

2.1. Preliminary Analysis

The train dataset was observed to contain 12,496 rows and 4 columns, namely 'index', 'description', 'host_about', and 'unlisted'. On the other hand, the train_reviews dataset consisted of 721,402 rows and 2 columns, specifically 'index' and 'comments'. The ‘index’ is a foreign key with correspondence to ‘id’ in the train dataset.

It was observed that there were no missing values in either dataset. However, some values in both datasets were found to consist only of a space ' ', which were subsequently converted to NaN values for further processing. It was also observed that a subset of the properties in the 'train' dataset, specifically 4,029 properties, did not have corresponding entries in the 'train_reviews'. This observation suggests that these 4,029 properties do not have any associated comments. Regarding duplicates, the train dataset exhibited instances of duplicate entries. However, upon closer examination, it was determined that these duplicates corresponded to different rooms within the same property, and thus were considered valid data points.

Similarly, duplicates were identified in the train_reviews dataset. However, these duplicates were primarily attributed to emojis, as well as single characters such as '+' and '!'. These duplicates will be addressed and treated in subsequent stages of the data processing pipeline.

Besides missing values and duplicates, it was noticed that there are many values with html tags, emojis and single words.

2.2. Dependent Variable

The dependent variable in this project is the 'unlisted' column, which indicates whether a property listed on Airbnb would be unlisted in the next quarter. Upon evaluation, it was observed that the dataset 'train' exhibits an unbalanced distribution with respect to the 'unlisted' observations. Specifically, the number of properties labeled as not unlisted is nearly three times higher than the number of properties labeled as unlisted.

Furthermore, when considering the number of reviews, a notable disparity emerged between unlisted and not unlisted properties. The unlisted properties had a significantly smaller count of reviews compared to the comments received by not unlisted properties.

It is important to account for this class imbalance and its potential implications when developing predictive models and interpreting the results, as it may impact the model's ability to accurately predict the unlisted status of properties.

2.3. Independent Variables (Text Variables)

2.3.1. Simple Cleaning

As noted in the preliminary analysis, there are several html tags in the dataset, so for better exploration, these were removed a priori.

2.3.2. Language Detection

Language identification was performed using the 'detect_lang' function from the langdetect package. Among the different text variables, it was observed that the language detection process was unable to identify the language for 51 descriptions, 256 instances of 'host_about', and 11,260 comments. The reasons behind these undetected languages could be attributed to various factors, such as text samples containing highly specialized or uncommon languages, noisy or ambiguous text, or limitations of the language detection algorithm itself.

Nevertheless, as compared to the total sample, these are very small values, the language identification proved valuable in characterizing the linguistic diversity present in the dataset. English emerged as the predominant language across all text variables (81.3% for ‘description’, 72.8% for ‘host_about’ and 65% for ‘comments’. Additionally, a significant presence of Portuguese was observed in both 'host_about' and 'description' fields (20.6% and 14.8% respectively), highlighting its relevance in the dataset. In the 'comments' section, a higher variety of languages was encountered, with English, French, Portuguese, and Spanish being the dominant languages in that order (65%, 15.1%, 6.4% and 5.4% respectively).

To facilitate future analysis and filtering based on language, the dataset was updated with new columns indicating the identified language for each text variable. This additional information provides a foundation for exploring language-specific patterns and conducting language-specific analyses in the subsequent stages of the project.

2.3.3. Word Clouds

The utilization of word clouds proved to be a valuable tool in gaining insights into the dataset and determining the appropriate data cleaning and processing techniques. It highlighted the presence of common stopwords in all three text variables. This observation emphasized the need to remove these stopwords during the text cleaning process to focus on more meaningful and informative words.

Furthermore, the word clouds allowed us to identify recurring keywords that were shared among the variables. For instance, words such as "Lisbon," "apartment," "place," and "city" emerged as prominent terms across the dataset. These findings indicate the significance of these keywords in describing the properties and their surroundings, underscoring their relevance in subsequent text processing and analysis stages.

By recognizing the prevalent terms, cleaning and preprocessing techniques can be tailored to effectively handle these recurring words.

2.3.4. Words and Character Count

Analyzing the word counts and character counts of the dataset, several interesting observations can be made. Firstly, it was found that the "host_about" field tends to have a higher number of words compared to the "description" field. However, when examining the "comments" field, it becomes apparent that it contains a significantly larger number of words in comparison to both "host_about" and "description".

Furthermore, a notable disparity arises between the "not unlisted" and "unlisted" properties in terms of word counts. Specifically, the "not unlisted" properties tend to have higher word counts than the "unlisted" properties. This trend holds true when considering both "description" and "comments" fields, where the "not unlisted" category consistently exhibits higher word counts compared to the "unlisted" category.

A similar pattern emerges when examining character counts.

2.3.6. Natural Language Analysis

In this section, a more detailed analysis is made for the count of words, characters, stop words and punctuation. Comparing with the other variables, the description shows a higher number of stop words and punctuation per observation, with averages of 42 and 57, respectively. On the other hand, reviews have lower average of number of stop words and punctuation (16 and 10, respectively), however it shows a much higher maximum (503 and 1131, respectively) which is quite distant from the 3rd quartile, with counts of 22 and 12, respectively.

2.3.7. Named Entity Recognition and Frequency

Named Entity Recognition is a subtask of Natural Language Processing involves identifying and extracting named entities such as people, organizations, and locations from text data and classifying them into predefined categories. Named Entity Frequency is just the frequency of those named entities. Before applying it as a pre-process step, the named entity frequency was explored.

The most common entity is PERSON, meaning that people, probably the hosts, are often referred in the reviews, following by ORG that refers to organizations namely the name of companies.

3. Steps of Preprocessing

(Considered as extra work: replace_emoticons, remove_emojis, replace_entities, replace_empty, removing of HTMLs, translation)

Following the initial exploration, it became evident what type of treatment was required.

Any numeric value that corresponded to **monetary values**, such as '5$', '3 euros', and '200 reais', were changed to '#money', as well as **dates** in several formats, such as '16h00', 05/06/22 and '5pm' were converted to #datetime. Similarly, commonly used emojis such as 😀 and 😢 were changed to #good and #bad, respectively. Furthermore, making use of named entity frequency, characters recognized as **numerical values** were replaced to #number and **locations** identified as GPE (geo-political entities) and LOC (locations) are replaced with #place.

Afterwards, the text underwent several preprocessing steps, including the removal of the remaining emojis, numbers, HTML tags, links, English stop words, non-alphanumeric/non-whitespace characters and short words with less than three characters. Additionally, lemmatization was applied to retain the meaningful base form of the words, however, stemming was not applied in order to have a more linguistic accuracy and interpretation once stemming can produce non valid words.

In summary, these preprocessing steps described above are an exploration of our possibilities and show the possibility to transform the text data into a more standardized format suitable for analysis. However, it is important to note that the specific preprocessing steps may vary depending on the type of model used. Therefore, in preparation for each model, we will perform additional filtered preprocessing as necessary to ensure optimal results. By adapting our preprocessing approach to the specific requirements of each model, we aim to maximize the effectiveness of our predictive models and improve the overall accuracy of our predictions.

4. Re-Exploration of Independent Variables

After the cleaning, a re-exploration of the independent variables was done to gather more insights.

4.1. Most Common Words

To know the most common words, this exploration had to be done after pre-processing so that the stop words would not interfere**.**

As also observed in the word clouds before pre-processing, words such as “apartment”, “bedroom” and “room” are still very common. For the reviews in specific, the word “great” is the second most common regardless of whether the property remained listed or was unlisted, the words “host” and “nice” are also common among both types of properties. One would expect to have a common word with negative connotation for the unlisted properties, however this is not the case.

4.2. N-Grams

Following on from the most common words, for all the three variables there is no big distinction in the most common 2-grams for properties of the two classes. In description is common to have “living room”, “double bed”, “fully equipped” and “equipped kitchen”, the latter two seem to be only one 3-gram that would be “fully equipped kitchen”. Regarding host about, “looking forward” and “see soon” are common 2-grams showing friendliness on the part of the host. An interesting observation is that for properties that have been unlisted, it is common to have the 2-gram "portuguese place" which is not found on the properties that remain listed. As for the comments, “great location”, “highly recommend” and “great host” are common grams for both class and, again, this is not expected for properties that have been unlisted, in fact, the 2-gram "really nice" is more common in these properties than in those that have remained listed.

4.3. Topic Modelling – Latent Dirichlet Allocation

Topic modelling is a technique for identifying topics or themes that run through a collection of documents. One popular algorithm for topic modelling is Latent Dirichlet Allocation, it assumes that each document is a mixture of a small number of topics, and each topic is a mixture of a set of words. The model uses statistical inference to estimate the probability distribution of words in each topic and the probability of topics in each document. This exploration was only performed for the variable comments and only for those in English since the package used was only available for this language.

The number of topics explored was 10 and the difference between them is visible. For example, the cluster for topic 4 contains observations that use the terms “nice”, “good” and “well” conveying a positive feeling, on the other hand, the cluster for the topic 9 is represented by terms such as “walk”, “metro”, “station” and “train”, given an idea of transport and commuting. Four of the 10 topics have a certain overlap, using some of the same terms, which means that in a deeper topic modelling analysis it would be interesting to analyse 7 different topics.

5. Modelling

5.1. Pre-process for Modelling

Prior to the modelling phase, specific preprocessing steps were performed for each model. For the train dataset, we have created the following sub-datasets:

- ‘english_only_train’ – This dataset includes only the rows from train.csv that are in English. Was applied all the preprocessing steps discussed earlier to these rows. Will be used in all the section of the notebook ‘English Only’.
- ‘translated’ – In this dataset, firstly was applied a simple preprocessing step, which involved removing links and HTML tags. Then, we used a free translation service provided by Google to translate the text into English. Finally, we applied the same preprocessing steps as mentioned earlier to this translated text. Will be used in all the section of the notebook ‘Translated to English’.
- ‘unfiltered’ – This dataset underwent preprocessing without removing stopwords and without applying the functions specifically designed for English text. Will be used in the section ‘Multiple Languages - unfiltered’.
- ‘withoutstopwords’ – Similar to the previous dataset, but with the additional removal of stopwords in multiple languages, encompassing all the languages identified in the data. Will be used in the section ‘Multiple Languages - filtered’.

For the train_reviews dataset the same approach was used, and the pre-processing employed was the same as in the 'withoutstopwords' dataframe:

- 'final_df_merged' – involved concatenating all comments for the same index into a single text. Additionally, the columns 'description' and 'host_about' from train were added and combined into a single column called 'description_host'. Will be used in the section 'Train Reviews Global - Merge'.
- 'sample_merged' – dataframe followed the same concatenation process, but only with a random subset of 50,000 rows. Will be used in the section 'Train Reviews Sample - Merge'.
- 'final_df_majority' and 'sample_majority' – are identical to 'final_df_merged' and 'sample_merge' respectively, but without concatenating the 'comments' column per index.

In this analysis, a series of theories were initially tested using the "train" dataset, as all the texts shared a common context. However, upon observing a noticeable discrepancy between the texts in the two datasets, it was realized that property owners would naturally avoid speaking negatively about themselves or their properties. To address this limitation, the theories were subsequently retested using the "train_reviews" dataset, which yielded more accurate and relevant insights. In order to streamline the analysis and ensure conciseness, the redundant reconfirmations were removed from the notebook.

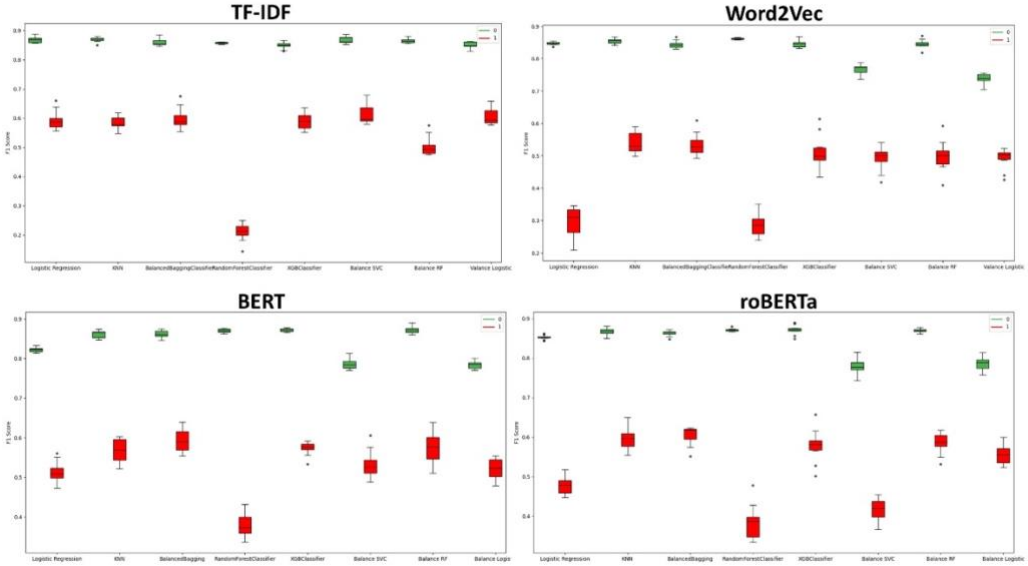
For the modelling phase, several classifiers were used: Logistic Regression, K-Nearest Neighbors, Random Forest Classifier, Balanced Bagging Classifier, and XGB Classifier. Notably, the latter two models were chosen specifically to tackle the challenge posed by class imbalance in the "train" dataset, with a ratio of approximately 1 to 3 between the classes.

Given the presence of class imbalance, the evaluation of model performance placed particular emphasis on the F1-Score metric. This metric, encompassing both precision and recall, effectively accounts for the imbalanced nature of the data, providing a comprehensive assessment of model effectiveness.

5.2. Modelling Train Dataset

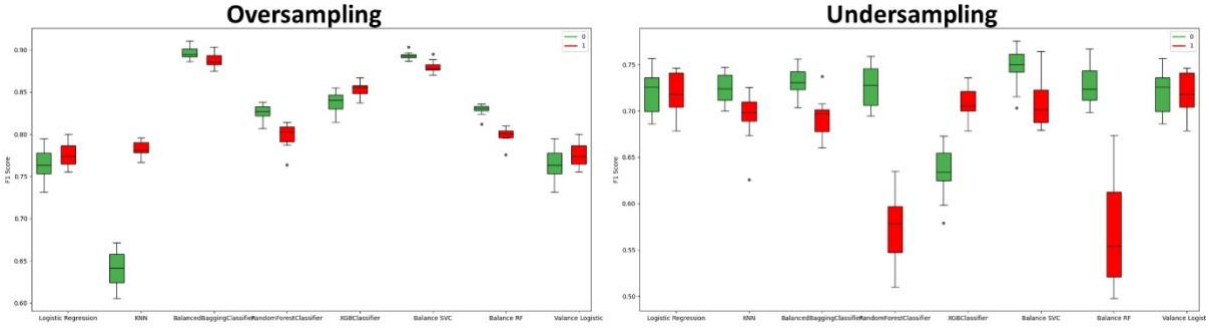
(Extra work feature engineering: m-BERT tokenizer, XLM-RoBERTa tokenizer, Oversampling, Undersampling)
(Extra work modeling: DistilBERT, M-BERT, BalancedBaggingClassifier)

Initially, a cross-validation Grid Search was conducted on the mentioned models to determine the optimal parameters. TF-IDF was used as a simple encoder during this process. To select the appropriate tokenizers, the performance of the models—Logistic Regression, Random Forest Classifier, and SVC—was compared using TF-IDF, Word2Vec, BERT, and RoBERTa.



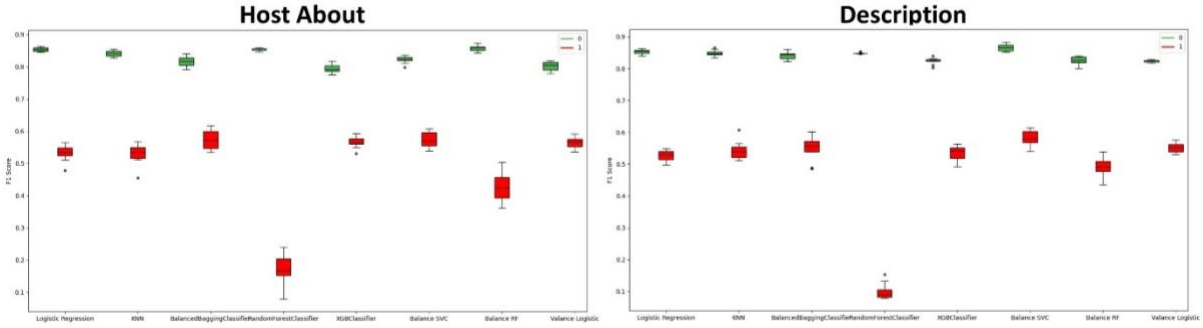
The results show that TF-IDF yields the highest f1-scores among the tokenizers, particularly for the unlisted class. This is crucial because the goal is not to create a model that solely predicts the majority label. It is worth mentioning that Random Forest performs relatively poorly compared to the other classifiers, and the inclusion of the 'balanced' class_weight parameter does not appear to have a substantial impact on the results.

To address the issue of dataset imbalance, under-sampling and oversampling techniques were implemented after applying TF-IDF:



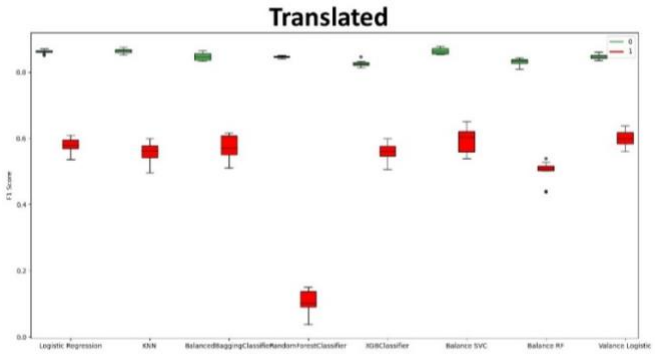
Both seem like good measures, so this will be further explained when selecting the best model.

Now that the parameters were optimized and the encoder was defined (TF-IDF), the model's performance were evaluated using only one column of the train set.



Observing these results, one can conclude that performance is better when both columns are used.

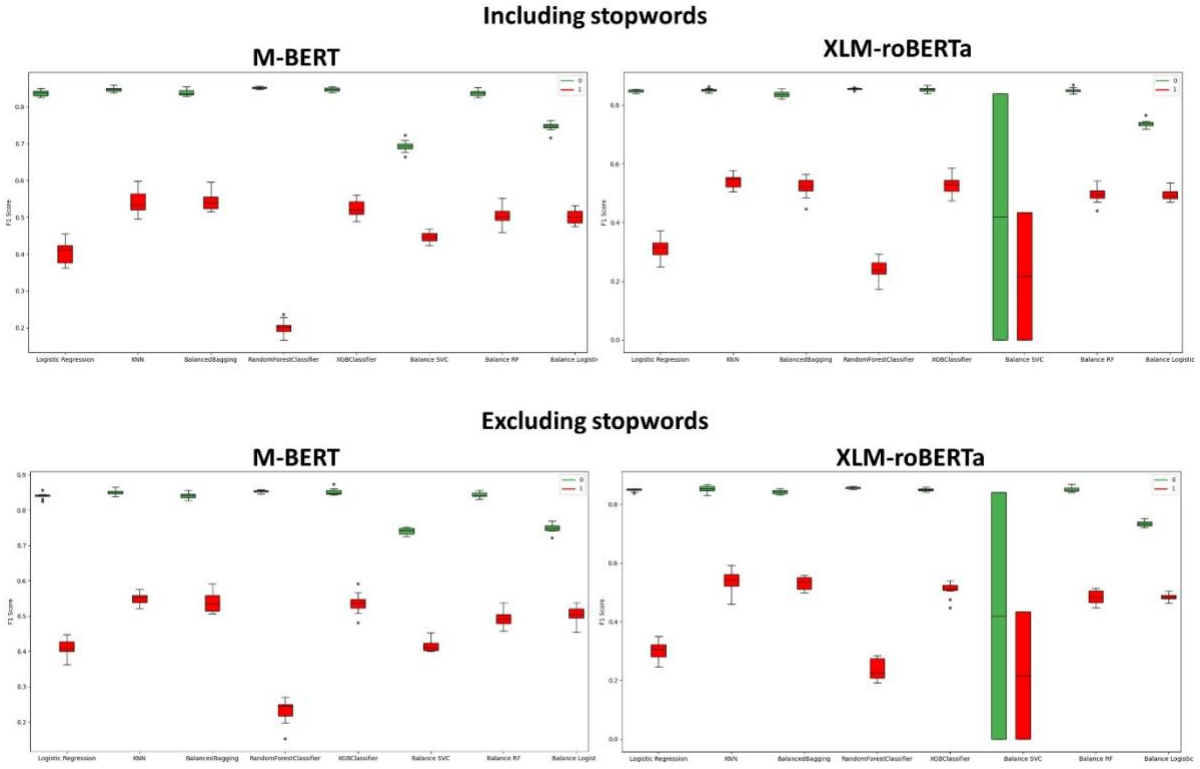
For further experimentation, the estimator's performance was also assessed using observations in the English Language (observations in other languages were translated) and using TF-IDF.



The performance of the models did not seem to be affected by including more instances of translated text for training. However, it is important to note that during the translation process using the googletrans API, 5886 observations failed to be translated. Attempts were made to address this issue by implementing a waiting period between requests, but still 3202 observations remained untranslatable. Another attempt without the waiting period resulted in only 2600 instances being unable to be translated. It was discovered that the API has rate, capacity, and maximum request limits, which are influenced by the number of ongoing requests to the API. Considering the limited

improvement in performance, the instability of the free API, and the fact that the majority of instances are originally in English, it was decided not to translate the text.

To address the challenge of handling multiple languages, multiple language word embedding models were evaluated. Specifically, the tokenizers m-BERT and XLM-roBERTa were compared. Additionally, the impact of including or excluding stopwords was assessed by comparing the performance of two different encoders in two different settings: one with stopwords and one without stopwords.



The inclusion or exclusion of stopwords did not show a significant difference in performance. Therefore, it was determined that using a rawer version of the data, including stopwords for all languages, would be more suitable for further modeling.

It is evident that both the 'BERT' and 'roBERTa' tokenizers yield similar results. Therefore, moving forward, only the 'roBERTa' tokenizer will be used due to its more comprehensive approach.

Addressing the issue of class imbalance, previous experiments involved under-sampling and over-sampling techniques. In order to optimize the solution and determine the best approach, the performance of under-sampling, over-sampling, and combinations of both methods were compared using the **Balanced Bagging Classifier** with the best parameters obtained from grid search.

	No under/over - sampling		Under-sample 'majority'		Under-sample (0.5)		Oversample		Under-sample + Over- Sample	
F1 - label 0	0.85		0.78		0.83		0.85		0.85	
F1 - label 1	0.53		0.54		0.54		0.51		0.52	
Model evaluation F1	0.525		0.538		0.535		0.507		0.517	
Model accuracy	0.768		0.7		0.756		0.766		0.774	
Confusion Matrix	1599	288	1314	493	1537	270	1616	191	1634	173
	372	321	257	436	341	352	393	300	391	302

Based on the analysis of previous results, it can be observed that under-sampling 'majority' provides a more effective solution for addressing the issue of the unbalanced dataset. Therefore,

under-sampling 'majority' will be implemented in the subsequent modeling training phase for '**M-BERT'** and '**DistilBERT**' models:

	Full M-BERT		Full DistilBERT	
F1 - label 0	0.64		0.77	
F1 - label 1	0.54		0.56	
Model evaluation F1	0.536		0.555	
Model accuracy	0.596		0.699	
Confusion Matrix	908	899	1279	528
	110	583	224	469

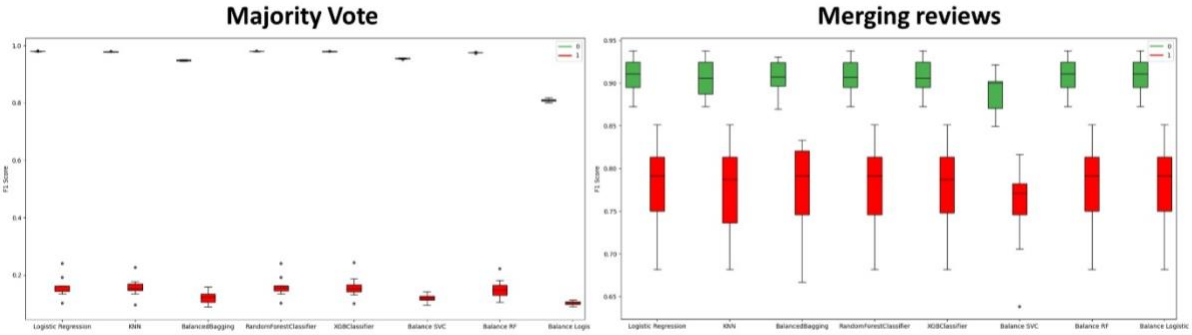
Based on the findings of this analysis, the **DistilBERT** emerges as the most promising approach, delivering the best results among the evaluated models.

It is important to note that due to resource limitations, the full RoBERTa approach could not be utilized in this study as the Google Colab session crashed even with a batch size of 2.

However, it is worth mentioning that while mBERT is well-suited for multilingual tasks and handling diverse languages, XLM-RoBERTa excels in cross-lingual tasks and language transfer. XLM-RoBERTa boasts a larger model architecture and vocabulary compared to mBERT, which contributes to its higher model size and computational requirements. On the other hand, DistilBERT offers a more lightweight and faster alternative to BERT while maintaining comparable performance.

5.3. Modelling Train Reviews

Continuing with the analysis of the review dataset, a subset of 50,000 instances was initially used for efficiency purposes during the exploratory phase, as using the entire dataset would be time-consuming. However, a unique challenge arose due to multiple comments associated with each property index. Simply splitting the data would lead to different predictions for the same index, as not all reviews can be uniformly positive or negative. To address this, two approaches were explored: modifying the final output of each index to reflect the majority prediction or concatenating all comments for the same index into a single text. Both methods were experimented with to determine the most suitable approach.



The merging approach not only yielded better results but also improved the computational efficiency of the code. Therefore, it was the chosen option for further analysis.

Similar to the analysis conducted on the train dataset, the performance and effectiveness of various under/over sampling techniques will be explored using, BalancedBaggingClassifier, for the reviews dataset:

	Under-sample 'majority'		Under-sample (0.5)		Oversample		Under-sample + Over- Sample	
F1 - label 0	0.89		0.89		0.90		0.90	
F1 - label 1	0.76		0.76		0.76		0.76	
Model evaluation F1	0.756		0.757		0.76		0.76	
Model accuracy	0.85		0.853		0.856		0.855	
Confusion Matrix	1548	259	1558	249	1566	241	1565	242
	115	578	119	574	120	573	120	573

Based on the analysis of the train reviews dataset, it can be observed that there is not a significant difference in the performance between the various under-sampling and over-sampling techniques. Although oversampling slightly outperforms under-sampling, the difference is minimal. Therefore, in order to avoid introducing additional synthetic text that may impact the models, it has been decided to proceed with under-sampling at a ratio of 0.5 applied to 'M-BERT' and 'DistilBERT' thus addressing the issue of class imbalance:

	Full M-BERT		Full DistilBERT	
F1 - label 0	0.90		0.89	
F1 - label 1	0.76		0.76	
Model evaluation F1	0.759		0.757	
Model accuracy	0.855		0.851	
Confusion Matrix	1567	240	1547	260
	122	571	112	581

Upon evaluating the performance of the different models, it can be observed that they yield very similar results. Among them, the **Balanced Bagging Classifier** with oversampling demonstrates a marginally better performance.

5.4. Ensemble

In the final stage of the experimentation, an ensemble model was constructed by combining the best-performing models for each dataset. DistilBERT was selected as the model for the train dataset, while the Balanced Bagging Classifier was chosen for the train reviews dataset. The ensemble model employed different approaches, assigning specific weights to each model prediction.

	Train/Train_reviews 50%/50%	Train/Train_reviews 30%/70%	Train/Train_reviews 70%/30%
F1 - label 0	0.90	0.90	0.76
F1 - label 1	0.69	0.76	0.55
Model evaluation F1	0.692	0.759	0.554

The analysis of the results clearly indicates that assigning greater weight to the train reviews dataset yields improved performance. This observation reinforces the notion that the comments and reviews play a crucial role in accurately classifying the unlisted category. However, even though the Balanced Bagging Classifier applied solely to the train reviews dataset slightly outperforms the ensemble approach, it is decided to adopt this model as the final choice for generating the ultimate predictions.

5.5. Final Model

For the final model, BalancedBaggingClassifier with oversampling was applied exclusively to the train reviews dataframe, as it demonstrated the most favorable performance. The results obtained from this model were as follows:

- **F1 Score – label 0**: 0.90
- **F1 Score – label 1**: 0.76
- **F1-Score model**: 0.76
- **Accuracy:** 0.855

The F1 Score for label 0 is 0.90, indicating a high level of precision and recall for identifying label 0. On the other hand, the F1 Score for label 1 is 0.76, indicating a slightly lower but still satisfactory level of performance.

The overall F1 Score for the model is 0.76, which aligns with the F1 Score for label 1. This suggests that the model performs consistently in predicting labels across both 1 and 0 classes.

Furthermore, the accuracy of the model is measured at 0.855, indicating that the majority of the predictions align with the true classes present in the dataset.

Overall, these results indicate that the final model, utilizing Balanced Bagging Classifier with oversampling on the train reviews dataset, successfully captures the patterns and exhibits a reliable performance in sentiment prediction.

The test dataset underwent the same pre-processing steps as the dataset used to train the final model. This ensures that the test data was processed consistently and in the same manner as the data used to develop the best-performing model.

The final model, which was determined to be the most effective based on previous evaluations, was then applied to the test dataset. The model made predictions on the test data (That didn't have the unlisted values). These predictions were extracted and saved to a CSV file for further evaluation and analysis of the project's performance.

6. Conclusions

In conclusion, this project explored various pre-processing techniques to enhance the data quality and improve the performance of the models. Different pre-processing steps were carefully analyzed and applied selectively based on specific situations and models. For instance, text cleaning, tokenization, and encoding techniques such as TF-IDF, Word2Vec, BERT, and Roberta were evaluated to find the most suitable approach for each task.

Throughout the experimentation phase, multiple models were tested, each offering unique insights into the analysis problem. Models such as Logistic Regression, Random Forest Classifier, Balanced Bagging Classifier and some variations of BERT were among those examined. While some models demonstrated more promising results than others, the final model, which employed the Balanced Bagging Classifier, showed strong performance in accurately predicting the target.

The unbalanced nature of the dataset posed challenges during the project. The class distribution imbalance, with a small proportion of instances representing one class compared to the majority class, made it difficult to achieve accurate predictions for the underrepresented class. Strategies such as under-sampling and over-sampling were explored to address this issue.

It is important to note that the dataset used in this project represents real-life data, making it more representative of actual label patterns in the target domain. However, the presence of unbalanced data, coupled with limited resources such as quality translators and computational power, posed constraints on the project. For instance, the full potential of the RoBERTa model could not be harnessed due to the unavailability of sufficient computational resources.