**Explanation techniques performed on a Yelp Reviews Sentiment Analysis**

**Final Project Report for the class in**

**Explainable Artificial Intelligence**

*Scuola Normale Superiore di Pisa*

*Winter semester 2021-2022*

Muhammad Waheed Sabir - *muhammadwaheed.sabir@phd.unipi.it*

Muhammad Imran - *muhammad.imran@phd.unipi.it*

Cosimo Palma - *cosimo.palma@phd.unipi.it*

### 

### 

[**Introduction**](#_tqzsr0zac2vh) **2**

[**Data Preprocessing and Exploratory Data Analysis (EDA)**](#_3ttquywat5fr) **2**

[Balancing dataset through under- and oversampling](#_jobr5m3x96i3) 3

[Data Cleaning](#_dxfm2vdffvgn) 3

[Vectorization](#_vf6i5wf295zb) 4

[Words distribution across sentiments](#_f6f4bcw3v07k) 4

[N-gram Analysis](#_8krw62q5oyx8) 5

[**Training and Testing**](#_ixnyzxkjg77y) **5**

[Word embeddings](#_d9mtusssshni) 6

[Bi-LSTM](#_fs39yx6taqad) 6

[**Explanation Methods**](#_ce35tqi9wycc) **8**

[SHAP](#_sqdgm253axk6) 8

[LIME](#_ix5ybxmm6132) 10

[ELI5](#_7r070yqn0h5k) 11

[**Explanations’ comparison: metrics and final consideration**](#_fpobbcsrduu0) **12**

[**Consulted Literature and Online Resources**](#_38e6uo88p4sc) **14**

[**Appendix**](#_5flxfh7spzoj) **16**

[Multinomial Naive Bayes](#_w2kqezynxm50) 16

[KNN-Classifier](#_gqs4b3t99kx7) 16

[Support Vector Machine](#_knp5s3c9y6wp) 16

[Random Forest Classifie](#_faudtosasftp)r 17

[Bi-GRU-LSTM-CNN](#_1inuf19xtcvh) 17

### 

### **Introduction**

The present document reports our joint effort to apply some of the methods learnt during the course of Explainable Artificial Intelligence (from now on addressed as “XAI”). The dataset chosen for this purpose is a collection of **Yelp Reviews Dataset** [1].

We selected the project type A, whose required tasks are as follow:

* definition of a prediction/classification task;
* application of a standard data mining pipeline (data understanding, preparation, training and evaluation);
* on the machine learning model developed, explanations generation (by

exploiting several approaches using either methods presented during the hand-on lessons, or other state-of-the-art ones);

* Exploration, comparison and discussion of the obtained results (particularly, arguing on their usefulness for constructing a user-understandable statement of the decision-making process for a given classification task).

### **Data Preprocessing and Exploratory Data Analysis (EDA)**

For the sake of simplicity and better interpretability we are converting the multi-class sentiments to the binary class sentiments (positive sentiments, negative sentiments) in our data. For the classification task we opted for the Sentiment Analysis, with a simple binary outcome (positive or negative), which was inferred from the column “label” (containing the rating from 1 to 5) as in the following.

content label sentiment

0 My wife took me here on my birthday for breakf... 5 1

1 I have no idea why some people give bad reviews... 5 1

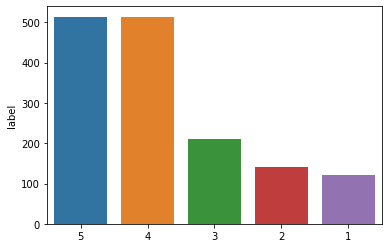
2 love the gyro plate. Rice is so good and I als... 4 1

... ... ... ...

1496 Yum best donuts in Phoenix. 5 1

1998 This used to be my favorite fro-yo place. A c... 3 0

1999 After hitting up the bank to sign some paper w... 3 0



From this histogram as well we can understand that the amount of positive reviews is far higher than the negative ones. In order to perform accurately, the model needs to be balanced, a task which can be accomplished in different ways.

#### **Balancing dataset through under- and oversampling**

Even after ranking the 3-stars reviews as negative our dataset results as unbalanced. Therefore, we are recurring to other techniques to achieve the balance. Considering that the positive class instances are more than double more than the negative ones, we decided to apply both random undersampling and oversampling: using only the first approach would have shrunk our dataset excessively, hence reducing the recall of the training fit as well. Using only the second approach would have practically meant to double the weight of all words classified as negative. As can be noticed from the word clouds in the following paragraph, there are many words that are classified as both positive and negative. Using random oversampling exclusively would inevitably and incorrectly switch the classification of some positive words to negative. For these reasons the chosen heuristics is to re-add to the dataset the half of negative reviews and to delete from it approximately three tenths of the positive ones. From the following representation we can assess that the adopted strategy has delivered the expected results:

****

#### **Data Cleaning**

Data cleansing is a technique of detecting and correcting inaccurate data instances from the dataset which includes identifying incomplet, incorrect data parts and replacing, modifying or deleting the unwanted data instances. In order to improve the quality of the dataset at hand to improve the accuracy of the sentiment analysis we have performed following data cleaning steps on the Yelp review dataset.

* Lemmatization
* Spelling checking and correction
* Removal of missing values or empty strings (newlines)
* Removal of URLs
* Removal of HTML tags
* Removal of Emojis[[1]](#footnote-0)
* Removal of Stopwords
* Removal of punctuations

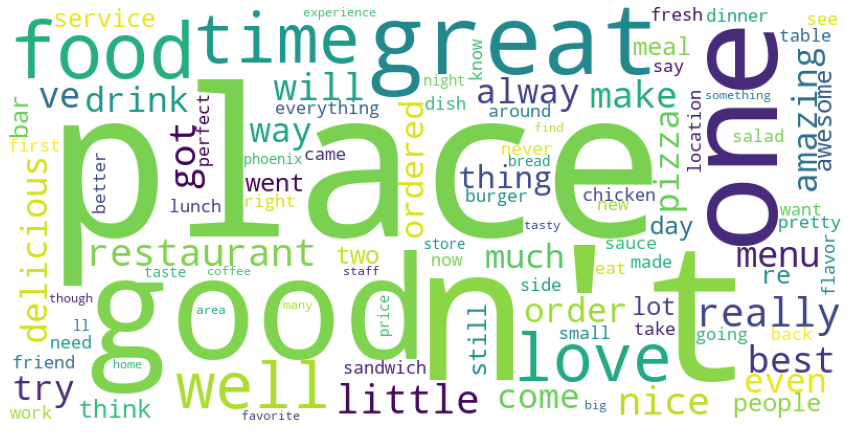
#### **Vectorization**

Different approaches are available to achieve this important operation, allowing the collection´s conversion to a matrix of numerical features. Although we initially opted for the Count Vectorization [8], we soon realized that a better solution is represented by the TF-IDF vectorizer [Term-Frequency (times) Inverse Document-Frequency], which considers the overall document weight of a word. Using it we can penalize the most frequent words and highlight the rarest ones. Tfidf-Vectorizer weights the word counts by a measure of how often they appear in each review, so that the text frequency is automatically normalized.

This procedure is the very last of the preprocessing phase, which allows the dataset to be split into training and test parts. Before moving to the next chapter, let us exploit this vectorization to visualize the distribution of sentiments across words. This direct gaze into the dataset will allow us to understand how unambiguous the separation between negative and positive words is, therefore how challenging the training phase will be.

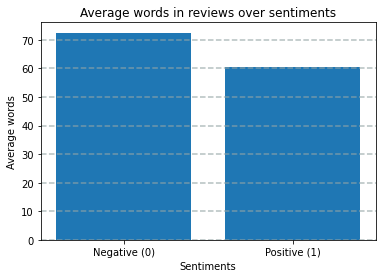
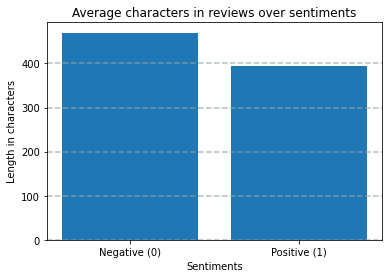
#### **Words distribution across sentiments**

Before looking at single words´ behavior, the *Word Cloud* visualization allows us to already understand that somehow the vectorization manages to catch some important difference between positive and negative reviews, whereas a huge amount of overlapping is unavoidable. Words present only on the left side (positive words) are for instance “amazing” and “delicious”; “good” and “great”, although also present in the right side (negatively labeled words), are definitely bigger. Strictly negative words on the other side are “bad”, “better” and “people”; somehow words such as “food”, “price” and “really” are significantly bigger than the correspondents in the positive window.

These observations let us reflect on the complexity of the task we aim to achieve: there are indeed no words which are *per se* negative or positive, but it depends only on their context, i.e. the ways they are used. We need to bear in mind that the final classification shall be performed hence on the whole review, not on the single words, and shall yield the label which most probably it has, according to the sentiments of each single word. Even this awareness shall let us look skeptically at our results, and look for a more comprehensive and nuanced language model which does not rely on the outdated and imprecise assumption that “the words are the building blocks of a sentence”.

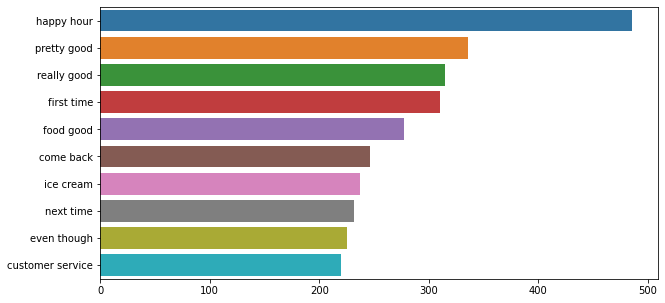
Another important observation on the quantitative features of the examined text, as can be noticed from the following histogram, the negative reviews tends to have more characters. A classical algorithm would be better suitable to put into practice these observations than a Neural Network, and would be far easier to be rendered in a rational/ user-understandable fashion: if this claim is too bold to affirm that we shall have preferred another paradigm to solve this classification problem, it surely plead for their forthcoming integration[[2]](#footnote-1).

In the similar way, another important observation on the quantitative features of the examined text, as can be noticed from the following histogram, the negative reviews tends to have more number of words than positive reviews.



#### **N-gram Analysis**

In this section we will do a bigram (n=2) analysis over the reviews. It shows the most common bigrams in Yelp reviews. The following bigram analysis shows the top 10 most frequently used word pairs in the dataset. In the following section regarding "Word embedding" we will better show how the algorithm can also capture the word´s context while assigning it to a class.



### **Training and Testing**

A dataset split has taken place also before the vectorization, because some models we used took as input string parameters (i.e., the vectorization was built in the model). Before diving directly into evaluating a bunch of training models (you can fully review them in the appendix), let us briefly sketch a tabular overview of the used metrics.

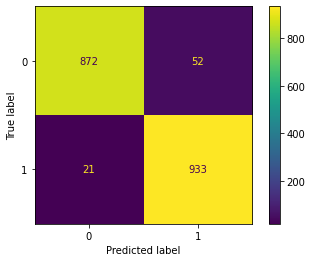
**True Positives** (TP) - correctly predicted positive values  
**True Negatives** (TN) - correctly predicted negative values   
**False Positives** (FP) – not correctly predicted positive values  
**False Negatives** (FN) – not correctly predicted negative values   
  
Accuracy, Precision, Recall and F1 score are directly derived from the above-defined features.  
  
**Accuracy** = TP+TN/TP+FP+FN+TN  
**Precision** = TP/TP+FP  
**Recall** = TP/TP+FN  
**F1 Score** = 2\*(Recall \* Precision) / (Recall + Precision)  
**Macro average** = Calculate metrics for each label, and find their unweighted mean. This does not take label imbalance into account.  
**Weighted average** = Calculate metrics for each label, and find their average weighted by support (the number of true instances for each label). This alters ‘macro’ to account for label imbalance; it can result in an F-score that is not between precision and recall [18].  
The **support** is the number of occurrences of each class in y\_true [18].

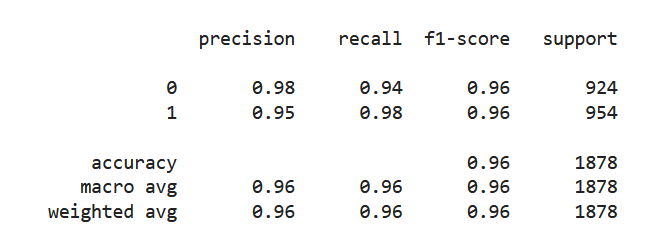
#### **Word embeddings**

An attempt to take into account the contextual dependency of the words has been achieved by adding a “Word embeddings” layer to the model Bi-GRU-LSTM-CNN [26]. This method represents words as dense word vectors (precisely, the word embeddings) which are not hard coded but trained. Word embeddings map semantic meaning into a geometric space. This geometric space is then called the *embedding space*. Thus semantically similar words are mapped closely on the embedding space like numbers or colors (a famous example in this field of study is the ability to map King - Man + Woman = Queen). There would have been two options for this. One way is usually training our word embeddings during the training of your neural network. The other way (the one we chose) is by using pre-trained word embeddings which we can directly use in your model (the one we have just selected). Each line in the file starts with a word and is followed by the embedding vector for the particular word. After creating the embedding matrix, it will just be implanted in the neural network through the Keras function “Embedding”.

#### ***Bi-LSTM***

Bi-directional long short term memory (Bi-LSTM) [25] is a kind of LSTM model which has the ability to process the data in both forward and backward directions. Bi-LSTM a sequence processing model consists of two LSTMs: one takes the input in forward direction, and the other in backwards direction. This model has shown to deliver better results according to accuracy, for which reason will be used in the explanation section as well. In the following graphs we can see the precision, recall, f1-score and accuracy. Since it is a binary classification, the confusion matrix is able to deliver a very clear overview of the precision and recall achieved from the model. The development of training and testing (validation) accuracy and loss is through each epoch is plotted as well.







### **Explanation Methods**

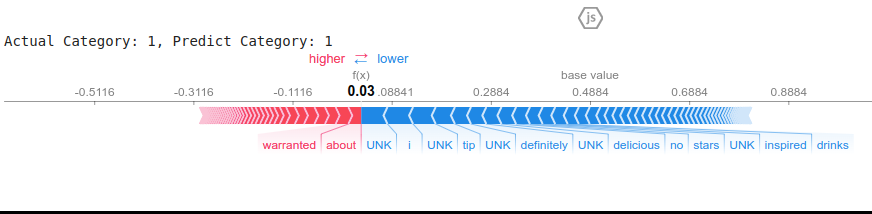
Despite the high accuracy of machine learning algorithms, often their opaque nature makes it difficult even for the domain experts to understand and trust them. There is an ultimate need in the field of eXplainable Artificial Intelligence (XAI) to seek new solutions to fill the existing gap between accuracy and interpretability in order to improve the trust and transparency of AI-enabled systems.

#### **SHAP**

SHAP method provides consistent explanations of models internal mechanisms based on game theory [17]. The main objective of the SHAP is to highlight or explain the individual contribution of various features of any instance which helped the Machine Learning / Deep Learning model to predict a certain class. It can be explained using the example of a cricket match scorecard summary which highlights the individual contribution of every player in terms of scores. From that score card it can be easily seen which players performed well in the match. In the same way SHAP helps to understand the importance of individual features in different predictions of a model which are helpful to validate the models and further improve their performance**.**

**Positive Review**

The following visualizations are produced with SHAP library which helps to make pretty visualizations of simple numeric measures to see which features were important to a model. This helps to make comparisons between features easily, and one can present the resulting graphs to non-technical audiences to explain the internal mechanism of the black box models.

****

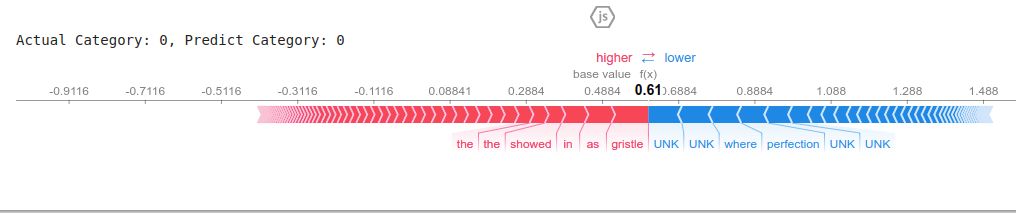
Following is the explanation of the above diagram to explain the text sentiment analysis model trained on yelp reviews dataset,

* SHAP has highlighted the words in a review instance with blue and red colors where words highlighted with blue color represents that they have contributed positively to predict the class label and on contrary the words highlighted with red color shows that they have negatively contributed
* SHAP has highlighted attributes "delicious" , "drinks" etc to classify the review as positive sentiment
* This force plot of Kernel Explanation explains how different features pushed and pulled on the output to move it from the base\_value to the model output value or prediction. The prediction of this kernel explanation is the probability or confidence that the review belongs to the predicted category ("Positive" Class).

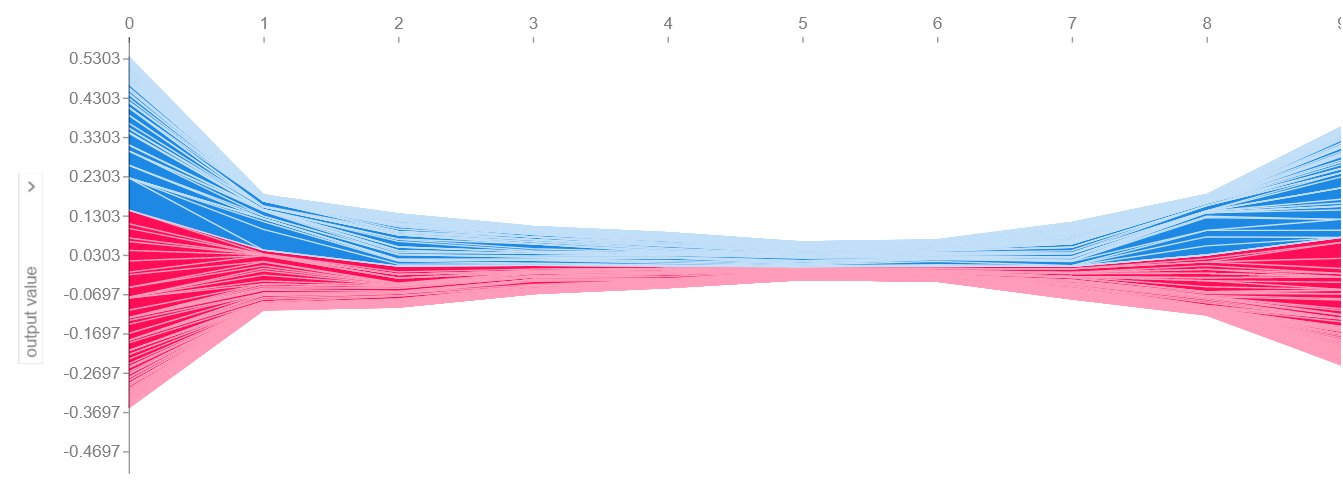
**Negative Review**:

Following SHAP visualization shows the text sentiment of the negative data instance form the yelp reviews dataset,

* SHAP has highlighted the words in a review instance with blue and red colors where words highlighted with blue color represents that they have contributed positively to predict the class label and on contrary the words highlighted with red color shows that they have negatively contributed
* SHAP has highlighted attributes "gristle" etc to classify the review as negative sentiment
* This force plot of Kernel Explanation explains how different features pushed and pulled on the output to move it from the base\_value to the model output value or prediction. The prediction of this kernel explanation is the probability or confidence that the review belongs to the predicted category ("Negative" Class).



The following graph shows the importance of individual features in predicting the class label or sentient category. The graph contains the two colors on the map, the first one (blue) being the one for positive SHAP values, and the second one (pink) for the negative SHAP values.

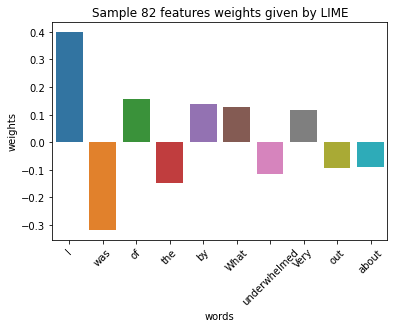
****

#### **LIME**

LIME is a model agnostic library to describe the importance or contribution of individual features of a model to predict a certain class. It is helpful to see the impact of perturbing on the input instances to pinpoint when the class label changes [15].

**Feature Importance**

The following visualization shows the explanation of features weight of the words in directions as seen. We have used the Sample 82 and perform the lime so here is the representation of words feature-weights are shown.



The following visualization shows the explanation of the sentiment classification of a random instance from the yelp review dataset. The words with negative sentiment are highlighted with “blue” color and words with positive sentiment are highlighted with “orange” color. From the diagram it can be seen that “0.25” or 25% is the rate of certainty of negative sentiments whereas the positive sentiment is labeled with a “0.75”, i.e. 75% precision. The degree of precision is also reflected from the color´s intensity.

****

#### **ELI5**

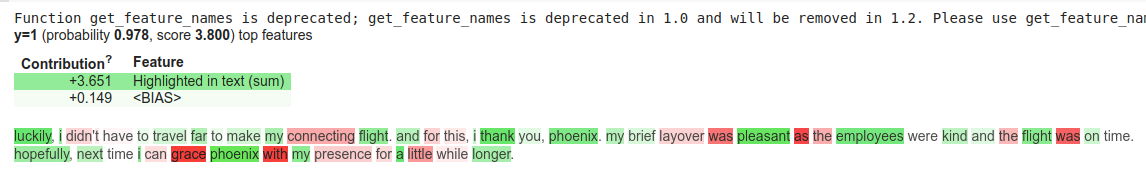
ELI5 (Explain Like I’m 5) was created in July 2011. ELI5 is becoming popular and being used in academia and other people who want to explain complex concepts in layman’s terms. It highlights the strength of the ELI5 to help others understand the difficult and complex scenarios or processes in simple ways.

Nowadays, an abundance of deep learning models are being developed to solve complex problems in almost all fields of life. However, it is difficult to understand the internal mechanism of the deep learning models due their high complexity and black-box nature. In order to understand and debug deep learning models, ELI5 a Python package was developed which provides the ability to understand the internal mechanism of the ML models [3]. It can be used to visualize the various parameters of a model as well as to exploit or understand the individual predictions of a model based on the significance of different features. It provides a convenient way to debug the decisions made by the model.

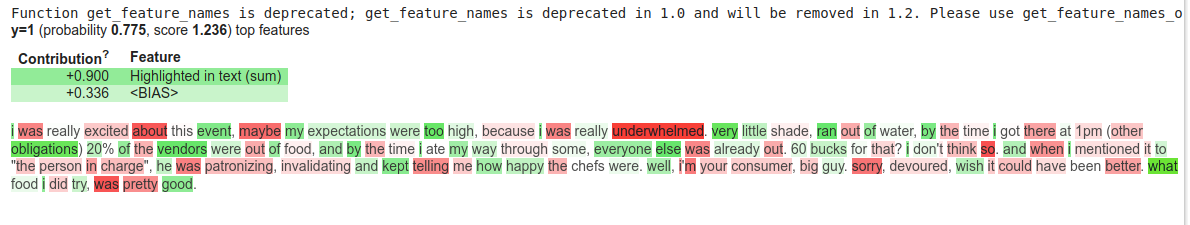
ELI5 is capable of providing both local model interpretations and global model interpretations. The global model interpretations can be considered as feature importance. ELI5 not only supports decision tree algorithms (XGBoost, Random Forest) but also supports all sci-kit learn estimators.

ELI5 introduced permutation importance for the global interpretations. In order to calculate the score, a feature or a word is replaced with another feature or word while making predictions. The main concept is to investigate the feature importance by looking how much it decreases or increases in the absence of a specific word. For the local model interpretations, ELI5 depends on the LIME algorithm. However, the format of display is different from the LIME but the idea is the same.

The following ELI5 based explanation visualization highlights the positive sentiment words with green and negative sentiment words with red colors. It can be seen that the overall sum of the negative words is “3.651” or 36.51% as this review is classified as a negative sentiment review.

****

The following ELI5 based explanation visualization highlights the words with red words with the majority of the words.



### **Explanations’ comparison: metrics and final consideration**

The used explanation methods focus on different visualizations and heuristics. Nevertheless there are different parameters which should be taken in consideration to validate them. Two of these are fidelity and computation time [4]. Fidelity aims at evaluating how well the explanation model is at simulating black-box decisions. This one is hence available only for methods which construct a surrogate model, as in the case of LIME [10].In the context of feature importance based explanations [5], we can evaluate the explanations also by *faithfulness.* It aims to validate if the relevance scores imply true importance: “higher importance values are expected for attributes that greatly influence the final prediction. The faithfulness method incrementally removes each of the attributes deemed important by the explanation. At each removal, the effect on the performance is evaluated. These values are then employed to compute the overall correlation between feature importance and model performance. These metrics correspond to a value between -1 and 1: the higher the value, the better the faithfulness of the explanation”. [10]

**Lime Faithfull metric mean and Standard deviation:**

| LIME Faithfulness Metric Mean | 0.69097357 |
| --- | --- |
| LIME Faithfulness Metric Standard Deviation | 0.0950 |

**Shap faithfulness metric mean and standard deviation:**

| SHAP Faithfulness Metric Mean | 0.410 |
| --- | --- |
| SHAP Faithfulness Metric Standard Deviation | 0.171 |

To these two metrics, which *quantitatively* assess the accuracy of the explanations, we can add a wide range of *quantitative* ones, like *comprehensibility* (how much effort is needed for correct human interpretation)*, succinctness* (how concise and compact is the explanation?)*, actionability* (what can one do with the explanation?)*, reusability* (could the explanation be personalized?) *and completeness* (is the explanation complete, partial, restricted?) and *stability* (does the explanation change, if the same input is given several times?)*.*

### **Consulted Literature and Online Resources**

[1]https://www.kaggle.com/yelp-dataset/yelp-dataset

[2] https://github.com/mayank100sharma/Sentiment-Analysis-on-Yelp-Reviews

[3] https://eli5.readthedocs.io/en/latest/tutorials/black-box-text-classifiers.html

[4] F. Bodria & al., *Explainability Methods for Natural Language Processing: Applications to Sentiment Analysis*, http://ceur-ws.org/Vol-2646/18-paper.pdf

[5] M. Sareela, *Comparison of feature importance measures as explanations for classification models,* https://link.springer.com/article/10.1007/s42452-021-04148-9

[6] https://medium.com/@kalia\_65609/interpreting-an-nlp-model-with-lime-and-shap-834ccfa124e4

[8] https://www.analyticsvidhya.com/blog/2021/08/text-preprocessing-techniques-for-performing-sentiment-analysis/

[9] Zhou, J.; Gandomi, A.H.; Chen, F.; Holzinger, A.; *Evaluating the Quality of Machine Learning Explanations: A Survey on Methods and Metrics*. Electronics 2021, 10, 593.

https://doi.org/10.3390/electronics10050593

[10] https://colab.research.google.com/github/francescanaretto/XAI-course\_2021/blob/main/Tabular/metrics-evaluation.ipynb

[15] https://github.com/marcotcr/lime

[17] SHAP, https://github.com/slundberg/shap#citations

[18] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision\_recall\_fscore\_support.html

[20] Multinomial-Naive-Bayes https://www.upgrad.com/blog/multinomial-naive-bayes-explained/

[21] KNN Classifier https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

[22] Support Vector Machines https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[23] Random forest classifier https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

[24]*Bi-GRU-LSTM-CNN* https://medium.com/@keynekassapa13/weighted-bi-gru-cnn-ab6cdbeda77b

[25] Bi-LSTM https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/

[26] Practical Text Classification With Python and Keras,

https://realpython.com/python-keras-text-classification/#using-pretrained-word-embeddings

# 

### **Appendix**

#### **Multinomial Naive Bayes** [20]

#### 

#### **KNN-Classifier** [21]

#### 

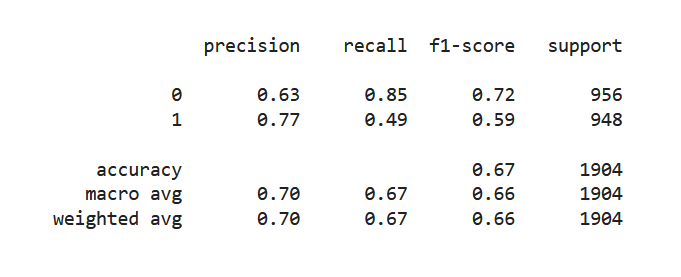
#### **Support Vector Machine** [22]

#### 

#### **Random Forest Classifier** [23]

#### 

#### **Bi-GRU-LSTM-CNN** [24]



1. Emojis could play an important role in boosting sentiment analysis performance, despite the eventually ironic or sarcastic way they could be used in. We decided to not further explore this possibility for brevity's sake. [↑](#footnote-ref-0)
2. One successful example of this attempt is represented by the Neuro-Symbolic AI. [↑](#footnote-ref-1)