Kialo claim impact prediction

Andrea Proia, Federico Spurio, Cristian Urbinati

February 2022

Abstract. This work aims to predict the impact of a claim inside a debate website like Kialo. We start with the idea that the impact is strongly dependent on the claim itself, then realized that maybe more factors influence the impact such as the depth in which the debate is located, the claim which it refers, its stance (Pro/Con) and more. The results are disappointing and point to the idea that the impact of a claim is strictly tied to the users' subjectivity and the topic of the debate.

Link to the project

The project discussed in this paper can be found at the following link: GitHub Project
The folder with all the scraped debates and the merged and cleaned dataframe used in the notebook can be found at: Google Drive folder

Introduction

The task is to predict the impact of a claim inside a debate. Debates are taken from Kialo [2], a website that covers several topics of discussion and where users can express their statement that can be in favor or disagree with the previous. The structure that is generated take the shape of a tree (Figure 1).
We first need to create a corpus dataset and extract all the information we consider relevant to do the training, then we perform some analysis of the distribution of the data and we try to apply both machine learning techniques and deep learning models to make regression on the impact value.
We use GloVe pre-trained model to obtain a vector representation of each word in the debates and then we compute the embedding for each claim, using a bidirectional LSTM to highlight relevant terms and patterns in sentences.

1 The dataset

The debates which compose our dataset are extracted from Kialo, a website that exposes a list of macro topics where users can express their personal opinion on that, by posting a claim Pro or Con to the claim at the previous level (parent claim). Each claim can be voted by other users, the mean of votes received determine the impact of the claim at that level and identify the most relevant replica to the one in the level before. The Figure 1 shows an example of the tree-structure that is generated.

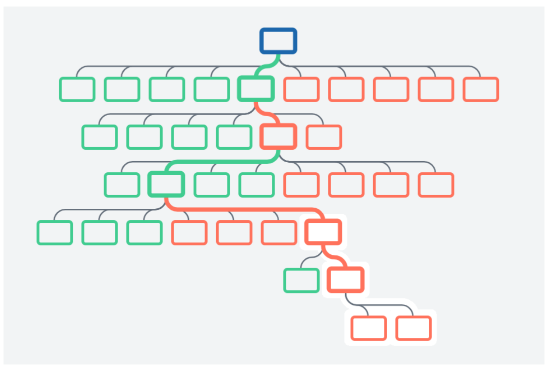


Figure 1: Example of debate structure: Pro (green) and Con (orange)

1.1 Dataset creation

Kialo itself doesn't make available all the information we needed. It only allows to download the content of each claim, its stance and its level (the Table 1 provides a description for each field). For that reason we manage to create a *scraper* (section 1.3) to run on a series of topics and that extracts also the votes received (essential to calculate the impact) and other information we repute useful, like the number of comments related to the claim and its depth, useful to perform statistics and avoid recalculating it each time from the level string.

Field	**Type**	**Description**
claim	string	sentence that states the position of a user with respect to another claim
stance	pro/con	it is Pro if the claim is in agreement with its parent, Con otherwise
level	string	string that represent the position of a claim in the discussion tree (i.e. the node that we are considering in Figure 1 has level "1.5.5.2.7.2")
votes	list of int	list of five integers which represents the number of votes received. Votes goes from 0 to 4 (i.e. $[0, 0, 1, 0, 2]$ means that the claim received one 2 and two 4)
comments	int	number of comments associated to a claim, usually users write comments to suggest a correction for typos, imprecisions or claims that have no fundaments.
depth	int	the depth of the level in which the claim is (i.e. the depth for the level 1.2.3 is 2)
debate name	string	name of the kialo topic from which the claim has been scraped
tot votes	int	total votes received by the claim
mean vote	float	the weighted mean of the `votes` field
parent claim	string	claim of the parent node
parent vote	float	mean vote (the impact) of the parent node
parent stance	pro/con	stance of the parent node
Pro	int	number of children which stance is pro
Con	int	number of children which stance is con
depth norm	int	normalized `depth` by maximum depth relative to the `debate name`

Table 1: Dataset fields description

1.2 Dataset cleaning

The results obtained from the scraper was not yet ready to be used as input for our model. In particular, claims, even if they must be approved by moderators, may contain a lot of characters that are not meaningful; so, we perform a cleaning process in order to eliminate links and references to other pages, topics or claims, we replace all punctuation and symbols that aren't words or spaces and finally we convert all words to lower-case to match the GloVe word embedding. Then we extract the impact score by calculating the weighted mean of received votes.
Since we find out from the beginning that the single claim was not sufficient to predict the impact, we extract also information of the parent claim which it refers (i.e. its claim, impact and stance).
Finally, we decided to discard all the claims that have not received at least two votes, due to the fact that, in the case of a single voting, user subjectivity play the main role.

1.3 Kialo scraper

The Kialo scraper is based on the Python library Selenium [4], and works on the HTML file of the website. Thanks to the "guide-map" (the one shown in Figure 1) placed on top of a debate page, it is possible to navigate between all the claims. So, the scraper retrieves the information about the child nodes of the selected claim (number of pros and cons) and starts a depth-first search, saving in a vocabulary all the already visited claims.

2 Basic assumptions and statistics

Our primary goal is to find correlations between a claim and its impact (the mean vote). In the early stages we assume that the content of the claim would play the key role to determine the impact the claim would receive, but soon we realized it was not the case. We think also that the claim impact could be related to its parent, but still no improvement even with this approach.

As a consequence, we started making different kinds of assumptions to consider some other aspects more related to the topology of the debate, instead of the textual content only.

2.1 Exploiting topology: depth

We started considering the topology of the debate to identify if there is a relation between impact and increasing depth. First, we thought that some users only read and vote the top parts of the debate tree. However, we were wrong, and each debate presents a different plot as we can see in Figures 3. Therefore, we cannot use depth as a feature for our models.

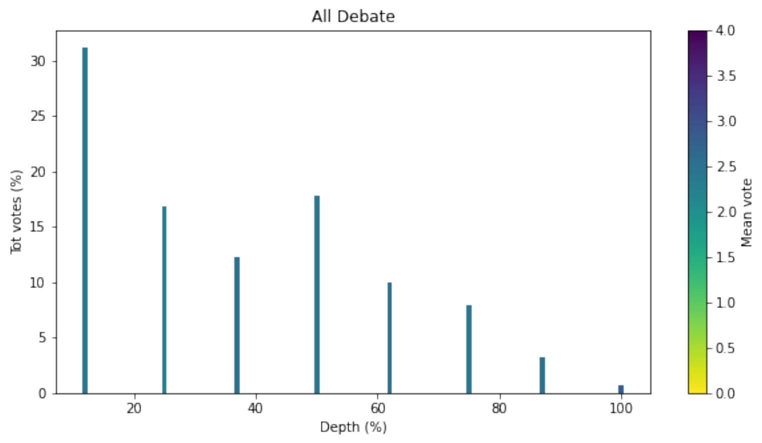


Figure 2: Trend of total number of votes and impact (mean vote) wrt the depth of the claim in the tree-structure. Number of votes and depth are normalized in percentage to be coherent among all the debates

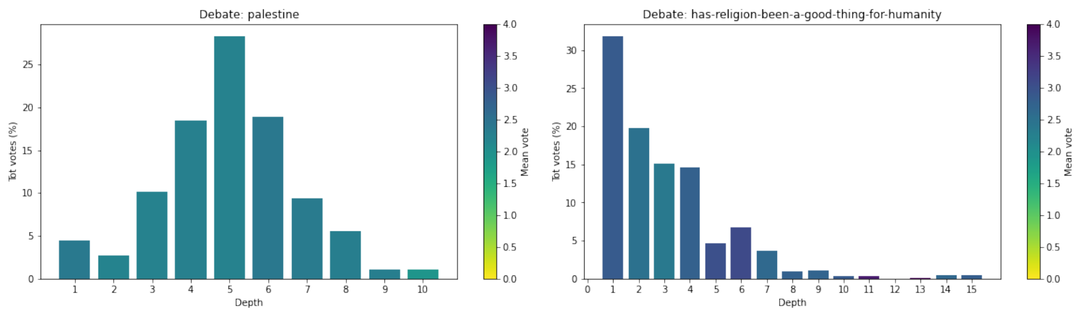


Figure 3: On the left an example of bad debate in which all the votes are around the mean, on the right we can see that number of votes decreases with depth but still no correlation exists between depth and impact

2.2 Exploiting topology: children stance

Continuing the study of the topology, we noticed that most of the debates observed present a trend between the current claim and the number of Pro/Con “children”. In particular, if a claim has more Pro children, its impact would be higher, while if there are more Con children, its impact would be lower. In case they are balanced, the impact will be around the mean value.

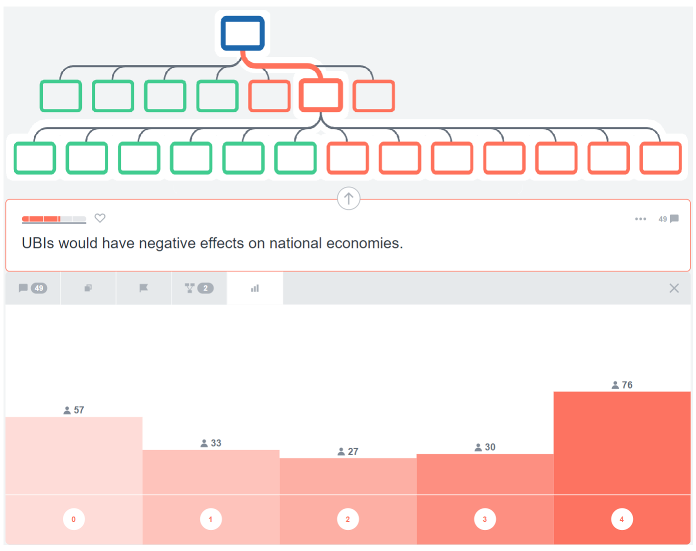
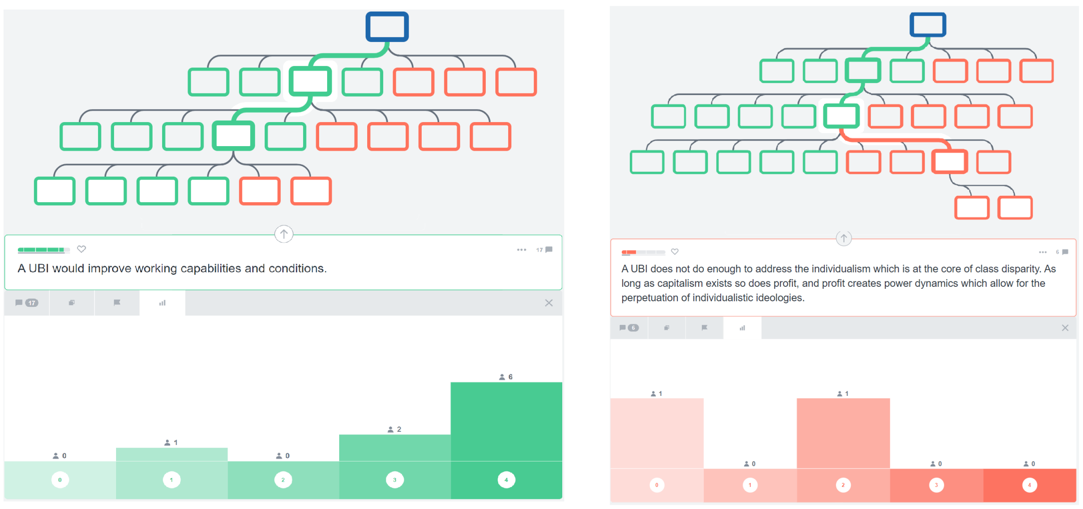

UBIs would have negative effects on national economies.

Figure 4: Claim with balanced number of children and score


A UBI would improve working capabilities and conditions.

A UBI does not do enough to address the individualism which is at the core of class disparity. As long as capitalism exists so does profit, and profit creates power dynamics which allow for the perpetuation of individualistic ideologies.

Figure 5: Claims with unbalanced number of children

As we can see in Figure 4 and 5, in general, claim with a balanced number of children has an impact near the mean value. On the other side, claims with an unbalanced number of children tend to have an impact higher in case of majority of Pro and lower in case of majority of Con. This trend is confirmed by performing a statistic on the entire dataset, as shown in Figure 6.

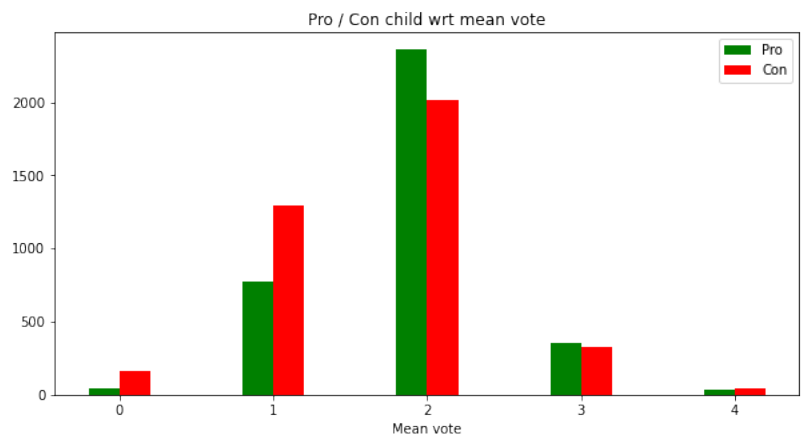


Figure 6: Pro/Con children for each class of vote

However, even if the general trend of the dataset is almost confirmed, some debates do not follow the same trend, while others are perfectly in line with it, as shown in Figure 7.

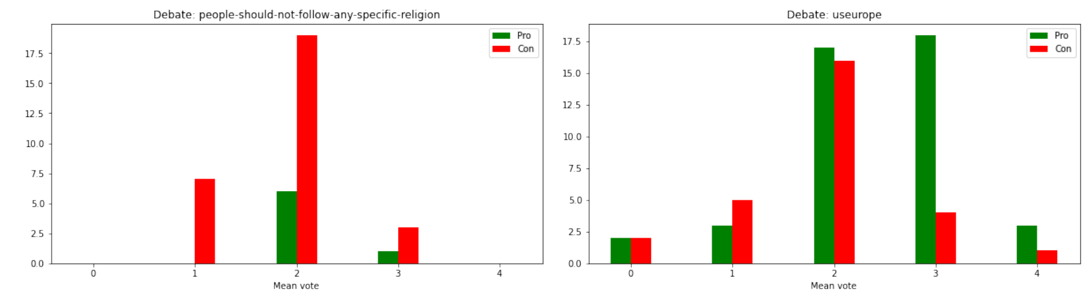


Figure 7: Comparing a bad example (on the left) with a good example (on the right)

3 Models definition and training

The first thing to do, since we are working with phrases, is to encode them in a numerical format, so that we can give them as input to a model. We used GloVe pre-trained model to obtain a vector representation of each word in a sentence and then we combine them to compute the embedding for each claim. Based on that, we have built the vocabulary and the embedding matrix: the former associates an index to each word, the latter stores for each row the embedding vector of the word matching that index. We manage out of vocabulary (OOV) words, associating them a random embedding vector.
Finally, we pad all the sequences of words in each claim to a maximum length by means of the string `-PAD-` encoded with 0 in the vocabulary. The maximum length, is determined by the longest sentence in the training set, emulating a real scenario where we can't prevent to have longer sentences at test time. If that happens sentences are truncated.

3.1 Neural network regressors

As mentioned before, our first attempt (model 0) was to consider only the content of the claim itself to identify if there are patterns or formulations of sentences that receives more approval or disagreement from other users. In this case we use a bidirectional LSTM to perform the sentence embedding, and two dense layers with ReLU activation that acts as a regressor to predict the impact.
The second approach (model 1) we attempted, is to feed the network also with the parent claim and then merge the two embeddings to see if the correspondences between the claim and its parent are useful for the prediction. The merging strategy we used simply concatenates the two embeddings (in this way all the information is maintained) and the cosine similarity calculated between the two.
Another model we tried (model 2) follows the idea which state that, if the parent claim has a low impact and the stance of the claim is Pro, its impact would be higher and, vice-versa, if the parent has a high impact and the claim disagrees, it would have a higher impact. So in this case, after having merged the embeddings, we concatenate the stance of the claim and the parent impact before the regression stage.
Finally, we tried (model 3) also to concatenate the ratio of Pro/Con children because we observed from the trend that, if a claim has more Pro children, it has an higher impact otherwise, if it has more Con children, its impact is lower. On the other side, if the ratio has a value near to one, Pro and Con children number is usually balanced.
We train all the models with Mean Absolute Error (MAE) both as loss and evaluation metric and Adam as optimizer. We choose Mean Absolute Error metric instead of others, like Mean Squared Error, because it is more robust to outliers [1], which are heavily present especially in the deepest levels. We also use early stopping to prevent the network to train longer if the validation loss starts to increase.

3.2 Machine learning approaches

We also tried to implement some classical machine learning regressors to check if they can perform better with respect to the more complex neural networks. Moreover, some of the classic regression models are often particularly suitable for this type of situation and work very efficiently. In particular, we tested different kinds of regressor:

- **Linear Regressor**: assume that the relationships between input and output variables are linear. This model is quite simplistic, and works by minimizing the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. [3]

- **Polynomial Regressor**: this model works by transforming the input variables in polynomial features, and feed them to a linear regressor.
- **Decision Tree Regressor**: grows by iteratively splitting tree nodes in a way that the variance in the partial datasets is minimized. This is done until the leaves contain no more impurities or a termination condition is reached.
- **Random Forest Regressor**: this is an ensemble of models that works by merging several uncorrelated Decision Trees which are influenced by certain random processes as they grow. The final model reflects an averaging of the trees.
- **Support Vector Regressor**: searches for a function that approximates the measurement points within a given accuracy, according to a specified kernel.

We tried to fit such models using all the features used for the neural networks and also tried only the features linked to topology (stance, parent impact, normalized depth and Pro/Con children ratio). We also used GridSearch to find the best configuration of the hyperparameters for Random Forest and Support Vector regressors.

4 Performance evaluation and results

For what concern neural networks models, we can see a small improvement with the increase of complexity and inputs we pass to the models, but they are far away to be nice results. As we can see in the table below the mean absolute error calculated in the validation phase is still high for predicting an impact in the range $[0, 4]$:

Model	**mae test**
Baseline (vector filled with the mean value)	0.5291

Table 2: Mean absolute error of the baseline model (vector filled with the mean value: 2)

Model	**mae valid**	**mae test**
model0 - only claim	0.4906	0.4167
model1 - merge claim with parent	0.4867	0.4257
model2 - concat. stance and parent impact	0.4691	0.4068
model3 - concat. Pro/Con children ratio	0.4587	0.4141

Table 3: Mean absolute errors on validation and test set of each neural network model

As a matter of facts, even evaluating the best model on the test set, the predicted value oscillates around the mean without any enhancement with respect to low and high impacts. So it does not recognise any significant pattern, it tries only to optimize the weight to predict a mean value that minimizes the loss.
Similar results are obtained with machine learning approaches. The results are slightly better than the baseline model, which simply generates a vector with the mean value, but the mean absolute error on the test is still too high.

Model	**mae test**
Linear Regressor	0.5025
Polynomial Regressor	0.4861
Decision Tree Regressor	0.4463
Random Forest Regressor	0.4379
Support Vector Regressor	0.5031

Table 4: Mean absolute errors of machine Learning models trained using the following features: stance + parent impact + Pro/Con children ratio + normalized depth

Features	**mae test**
Claim	0.5126
Claim + parent claim	0.5072
Claim + parent claim + stance + parent impact	0.4569
Stance + parent impact	0.4532
Claim + parent claim + stance + parent impact + Pro/Con children ratio	0.4718
Stance + parent impact + Pro/Con children ratio	0.4486
Stance + parent impact + Pro/Con children ratio + norm depth	0.4379

Table 5: Mean absolute error of Random Forest regressor according to different features

5 Conclusions

In conclusion, we were not able to provide good results, even if we have used different models (neural network and machine learning models) and different approaches (different combination of features). This proves that the cause of the poor results is not linked to the type of model used to predict the impact, but to the quality of the data and, above all, to the difficulty of the problem itself, given the very scarce correlation between the features and the impact of the claims.
The main reason relies on the fact that impact of claims taken from Kialo is subject to some noise due to the subjectivity of users that votes and the scarcity of votes itself.
It is quite interesting that machine learning models can reach similar results of neural network models without taking into account any textual content (claim or parent claim). This could have happened by chance due to the unbalanced dataset, or could lead to a further investigation about how users are heavily influenced by the structure of a debate, more than the content of an argument itself.
In any case, both neural network and machine learning learn some combination of features that leads to better results than the baseline model.
The fact that the predictions are between values 2 and 3 may be an effect of the fact that the distribution of mean votes is highly unbalanced in that range as shown in Figure 8.

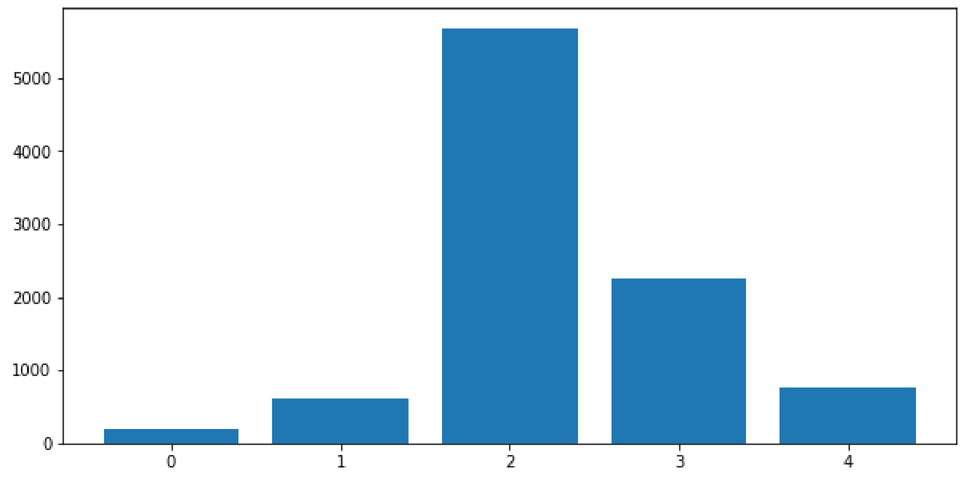


Figure 8: Distribution of mean votes of claims

As future research, to test if we can improve the quality of our regressors, we could do some data augmentation on claims with low impact, in order to generate small variant of those claims and therefore have a more balanced dataset or, even better, take into account other datasets more balanced and with a higher number of votes.

References

[1] *Best regression metric.* URL: `https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/`.

[2] *Kialo.* URL: `https://www.kialo.com/`.

[3] *Linear Regressor.* URL: `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`.

[4] *Selenium.* URL: `https://www.selenium.dev/`.