COMP9417 Group Project (Topic 2)

Task: CommonLit Readability Prize (https://www.kaggle.com/c/commonlitreadabilityprize)

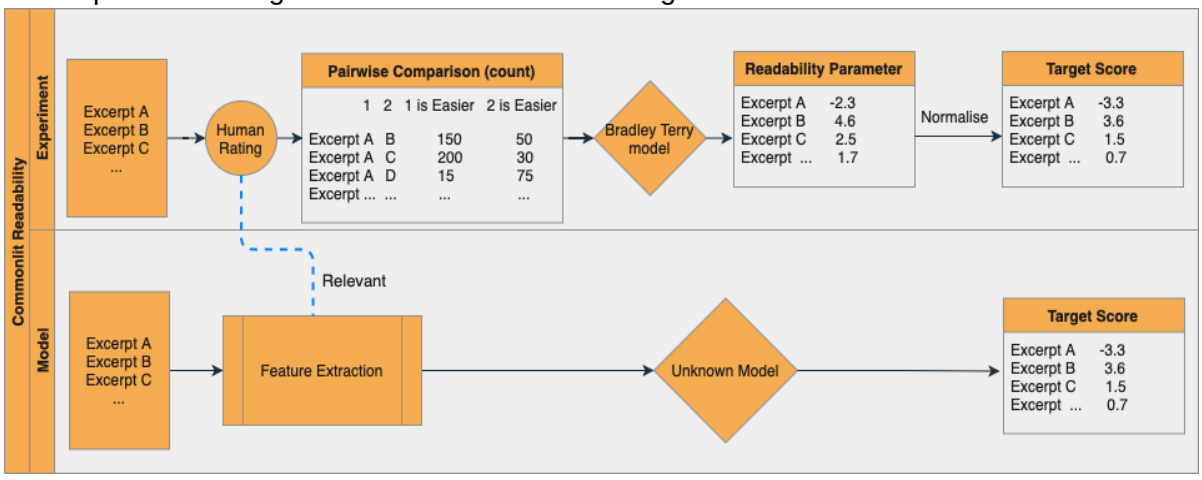
Team (56) Members: WENG XINN CHOW (z5346077) & ZIYAN GUAN (z5342540)

1. Introduction

Readability is an essential aspect of the writing quality measurement, informing the level of easiness for readers to digest one passage. However, it lacks a universally adopted and accepted readability measurement that helps readers match to their appropriate readings. The host of this Kaggle project, [5] questioned the validity of conventional readability measurements including Flesch-Kincaid Grade level test due to its weak construct and limited capturing of word composition in the formula. Therefore, this project was initiated with the objective of identifying one comprehensive NLP-based readability model/formula developed from the readability ratings collected from human reviewers. This work has potential for strong applications in wide use cases and presents us an opportunity to review some state-of-the art NLP models and sophisticated linguistic features.

The data in this Kaggle project was collected from a well-designed experiment in which experienced human reviewers were asked to compare and select the easier excerpt among 2 given experts. There are ~2,800 excerpts in total with 111,000 pairwise comparisons done. Then those comparison results were summarised in a contingency table, transformed into the final target score in the dataset by Bradley-Terry model [17] and normalization. Bradley-Terry model computes $\lambda_A - \lambda_B = logit\ of\ (Excerpt\ A\ is\ easier\ than\ B)$. $\lambda_A$is our post-normalisation target value for Excerpt A in the given data. The way we interpret this is: if Excerpt A has readability parameter = -2.3, B = 4.6 from Bradley-Terry model, we interpret as: Excerpt B is $e^{4.6+2.3}$ times more easier to read than Excerpt A on average.

The experiment design can be described in the diagram below:



In the training and test dataset, the participants were only given the raw text of excerpts, target score and the standard error of readability parameters. The challenge lies in both feature extraction and model

selection & fitting. The appropriate model needs to incorporate all those extracted features and transform them into a score that minimizes the RMSE to the target score.

2. Implementation

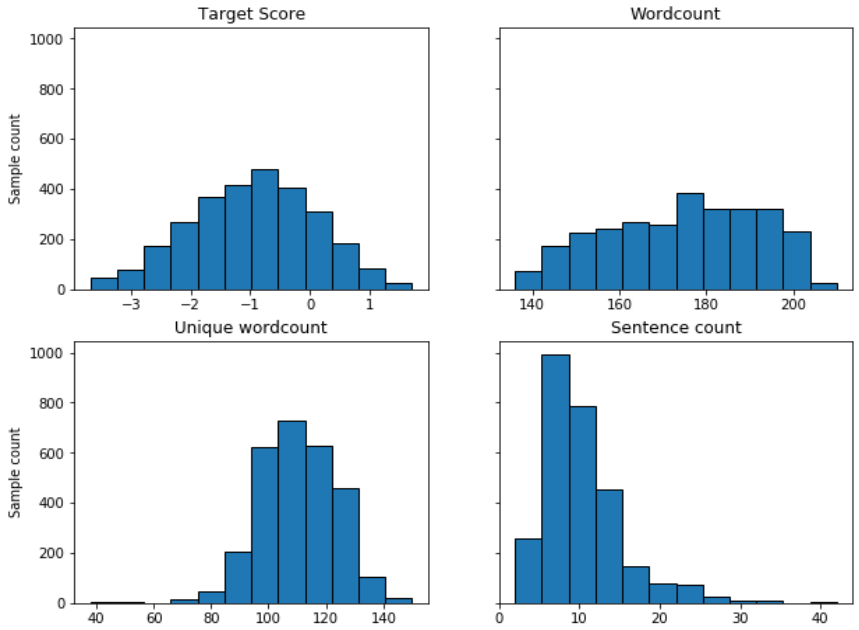
This section illustrates the methodology of model building, including exploratory analysis, feature extraction and model selection & evaluation.

2.1 Exploratory analysis

We run some preliminary analyses to ensure the data is free from outliers/anomalies and examine the underlying distribution of data points. The training dataset contains 2,848 samples and there is no missing value in the samples. Here are some initial observations in the data distribution:

- The outcome variable, Target score (Readability score) ranging within [-4, 2] follows normal distribution.
- We examine the word composition of each sample excerpt by word count, unique word count and sentence count. They are also approximate to normal distribution without showing too much skewness. There are only very few outliers in sentence counts.

This suggests the selection of sample excerpts and data collection process are well-designed and we do not need to implement any data cleansing or transformation.



2.2 Feature extraction

One challenge posed by this Kaggle project is there are no descriptive features available in the provided dataset that can predict the target score and we need to derive our own linguistic features that have impacts on readability from the excerpts.

Inspired by relevant literatures, we acquire a categorised list of 176 explicit features (here we are only presenting a compact view, please refer to Appendix for the feature glossary):

- Basic text statistics: preliminary descriptive stats of text including word count, sentence length, syllables per word etc.
- Conventional readability grading [14]: long-standing readability formulas built on basic text statistics including Flesch-Kincaid Grading Level, Dale-Chall Readability Score etc
- Part-of-Speech compositions [14]: those features are based on the POS tags of words in text, including count of adjectives, nouns, verbs and adverbs
- Sentimental analysis [14]: refer to the emotions and sentiments delivered to readers, including count of positive/negative adjectives
- Referential cohesion [14]: represent of logical connectiveness in the text, the four types of connectiveness are causal, logical, adversative and temporal
- Semantic information [2]: includes polysemy of vocabulary, semantic similarity between sentences in the paragraphs

We are aware that most researchers in the field collect those linguistic features using the advanced commercial NLP tools like Coh Metrix, TAACO [4,5]. But we could not scale the feature extraction process within the constrained time frame using those tools (it requires reaching out to the tool owners for batch query). Instead, we came across an alternative open-source webtool, Aztertest [2], which claimed to have over 90% accuracy rate in determining the factors for the readability classification. Since only 20 files can be evaluated by the webtool in each run, we scripted a small code to automatically submit all files and collect results in iterations. The script is attached to this project submission.

2.3 Review of suitable machine learning algorithms

Apart from feature extraction, the choice of machine learning algorithms can significantly influence text readability evaluation. One of the existing readability studies suggested that regression-based model and SVM, also known as Support Vector Machine, are the two baseline algorithms which are comparable in accounting for evaluating text readability [8]. Hence, the mentioned algorithms will be adopted as basic approaches for this task. It is also worth exploring the transformer-based model as it is used primarily in NLP problems [12].

The chosen machine learning algorithms for this project are listed as below:

- Linear Regression
- Support Vector Regression (SVM)
- Bidirectional Encoder Representation from Transform (BERT)

More detailed explanation about each machine learning algorithm can be found in the Appendix.

3. Experimentation

In this section, we will dive into different techniques that are involved in building our models, including cross validation, regularization and hyperparameter tuning. Having concerns with the size of the test dataset given by the host (with only 7 samples), we decided to exclude the dataset in the process of model performance evaluation. Consequently, we first split our dataset into training and true test sets with a ratio of 70:30 using the train-test split approach.

3.1 Regularization & Hyperparameter tuning

Regularization is necessary when building our models to prevent overfitting. In our project, we will be emphasizing on Ridge Regression, which is also known as the L2 norm, as the regularization method for both linear regression and SVR. A penalty term is added to the loss function to control the trade-off between allowing the model to increase or decrease its complexity. We add a tuning parameter, λ to determine the amount of regularization strength. The choice of the regularization strength is critical for the result of our model performance and this is why cross validation and hyperparameter tuning come in handy.

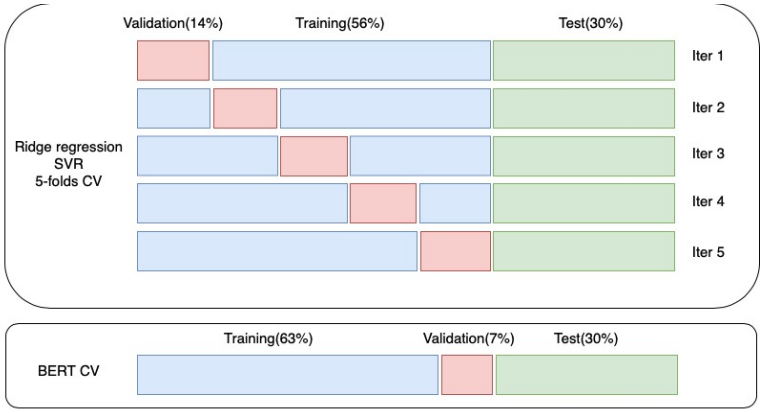
Hyperparameter tuning is an approach of searching for a set of optimal hyperparameters for our machine learning models. Oftentimes, models have important hyperparameters whose values can be exhaustive to be estimated directly from the data [11]. Hence, we first define a range of possibilities of these hyperparameters. Then, we build our models for every combination of the predefined hyperparameters using Grid Search and find the most ideal model that minimizes the RMSE.

Learning Algorithm	**Hyper parameters**	**Description**	**Range of Values**
Ridge Regression	Alpha	Regularization strength that adjusts the bias-variance trade-off. As the value increases, regularization strength increases.	0.0001, 0.001, 0.01, 0.1, 1, 10, 100
Support Vector Regression (SVR)	Kernel	The mapping function that lifts a feature space from a lower dimensional to a higher dimensional space.	Linear, Radial Basis Function (RBF)
	C	Inverse of regularization strength. A lower C value will lead to a stronger regularization.	0.001, 0.01, 0.1, 1, 10
Bidirectional Encoder Representation from Transform (BERT)	Max length	The maximum length of text sequence for each input. Text longer than the max length is truncated, shorter is padded.	180, 200, 220
	Epoch	Number of complete passes through the training dataset.	2, 3, 4
	Learning rate	The step size it takes at each iteration towards the optimal RMSE.	2e-5, 5e-5, 10e-4 [15]

3.2 Cross validation

Estimating the performance of a machine learning model by just training the model with the training set and predicting the true test set is insufficient because the accuracy of the model may be unreliable as it can vary along with different test sets. Our main objective is to estimate the expected performance of our models in general, hence, we introduced cross validation to assess our predictive models in a less biased way such that the whole training dataset is taken into account for both training and validating the models.

Here, we used k-fold cross validation for Ridge Regression and Support Vector Regression. On the other hand, due to the expensive time complexity of the BERT model, we applied the train-test split cross validation approach for BERT. We finalized the k value used for k-fold cross validation to be 5, which was chosen deliberately according to the nature and size of our dataset. The whole process of cross validation can be summarized as the diagram below:



4. Results & Findings

4.1 Result interpretation

By implementing cross validation and hyperparameter tuning, the output for each learning algorithm training with different combinations of hyperparameters can be summarized in a table.

4.1.1 Ridge regression

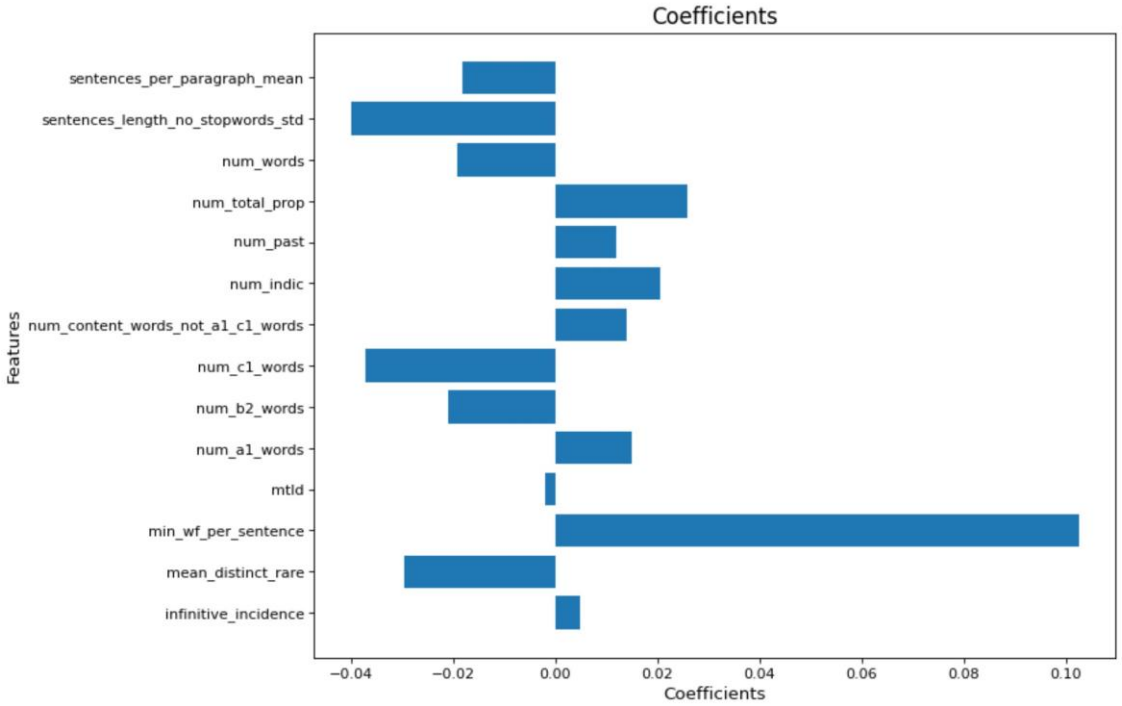
From the 5-fold cross validation, it can be observed that the tuning parameter, alpha = 1 gives the most excellent performance. By looking at the output table, it justifies our previous claim about the importance of the choice of parameter on the performance of the ridge regression model as inappropriate alpha parameters can trigger higher RMSE. This is because if the strength of the regularization is too high, many coefficients will approach 0, and thus it leads to an underfitted model. Conversely, if we set the regularization strength to be extremely low that it nearly becomes 0, there will be almost no effect of the regularization on our regression model.

rank	parameters	cv_s1	cv_s2	cv_s3	cv_s4	cv_s5	RMSE
1	{'alpha': 1}	0.7039	0.6673	0.7322	0.7188	0.7249	0.7094
2	{'alpha': 10}	0.7017	0.6690	0.7343	0.7166	0.7266	0.7096
3	{'alpha': 0.1}	0.7083	0.6663	0.7291	0.7216	0.7244	0.7099

4	{'alpha': 0.01}	0.7100	0.6662	0.7283	0.7227	0.7251	0.7104
5	{'alpha': 0.001}	0.7102	0.6662	0.7283	0.7228	0.7252	0.7105
6	{'alpha': 0.0001}	0.7102	0.6662	0.7282	0.7228	0.7252	0.7105
7	{'alpha': 100}	0.7002	0.6753	0.7384	0.7139	0.7332	0.7122

Besides determining the optimal regularization strength, we also want to investigate the most impactful features towards the text readability. We achieved this by fitting another Ridge Regression with the chosen alpha and finding the coefficients and p-values for all features. Then, we filtered out all features that have a p-value larger or equal to 0.05. Statistically, a higher p-value indicates a strong evidence for the null hypothesis and a low significance level. Those left features were then plotted as a horizontal bar chart for further analysis.

Judging from the bar plot below, min_wf_per_sentence that represents the minimum word frequency per sentence has the most significant impact on the text readability based on our data and machine learning model.



4.1.2 Support vector regression

There are two hyperparameters being tested in our modelling process for SVR, including C that represents the inverse of regularization strength and the kernel. In our case, we only assessed two necessary kernels:

- Linear kernel that is in one-dimensional space, which is used for regression problems and when we are dealing with large amount of features
- RBF which is suitable when there is no prior knowledge about the data.

We discovered that among 10 combinations of hyperparameters, the model with C = 0.1 and linear kernel produced the smallest RMSE, which is reasonable considering the decent RMSE obtained from linear regression. Consequently, from the results of both models, we deduced that one-dimensional feature space works well with our data.

rank	**parameters**	**cv_s1**	**cv_s2**	**cv_s3**	**cv_s4**	**cv_s5**	**RMSE**
1	{'C': 0.1, 'kernel': 'linear'}	0.6999	0.6840	0.7391	0.7240	0.7274	0.7149
2	{'C': 1, 'kernel': 'linear'}	0.6999	0.6856	0.7387	0.7234	0.7306	0.7156
3	{'C': 0.01, 'kernel': 'linear'}	0.6986	0.6905	0.7417	0.7210	0.7329	0.7170
4	{'C': 0.001, 'kernel': 'linear'}	0.7159	0.7048	0.7679	0.7284	0.7572	0.7349
5	{'C': 10, 'kernel': 'rbf'}	0.7345	0.7192	0.7830	0.7402	0.7675	0.7489
6	{'C': 1, 'kernel': 'rbf'}	0.7716	0.7656	0.8239	0.7768	0.8036	0.7883
7	{'C': 10, 'kernel': 'linear'}	0.8548	0.8249	0.9457	0.8732	0.8597	0.8717
8	{'C': 0.1, 'kernel': 'rbf'}	0.9103	0.9043	0.9351	0.8718	0.9092	0.9061
9	{'C': 0.01, 'kernel': 'rbf'}	1.0267	1.0103	1.0398	0.9497	1.0061	1.0065
10	{'C': 0.001, 'kernel': 'rbf'}	1.0521	1.0346	1.0649	0.9702	1.0293	1.0302

4.1.3 BERT

We can observe the effect of three hyperparameters **max length**, **epoch** and **learning rate** on the model performance from the iterative implementations:

- Max length is the maximum length of text sequence that can be tokenized and ingested into the model. The longer the text sequence can be fed to the model, the more complete contextual information can be processed and preserved in the model. BERT has a built-in max limit of 512. The word count of excerpt in this task ranges [130, 220], well below the model limit. In the first 3 iterations, we respectively configured max length as 180, 200 and 220. Not surprisingly, the model with max length = 220 has the optimal prediction outcome.
- Epoch is a parameter that controls how many times each sample in the dataset gets to update the underlying model parameters. If epoch is too large, it could train the data too excessively and overfit the training data. On the other hand, a too small epoch could lead to an undertrained model. The authors of BERT models suggest a reasonable range of epoch should be 2-4. We tested the effect with this range in interaction 3-5 and it turned out epoch = 3 has the best performance.
- Learning rate is the step size towards convergence to the minimal error. While 10^{-4} is a common choice, [15] suggested to fine tune with smaller learning rates reducing the catastrophic forgetting problem (forget previous learning while learning new knowledge). We tested 3 different learning rates in iteration 3, 6 & 7 and learning rate = 2^{-5}produces the optimal result.

Overall, we have the best model outcome with epoch = 3, learning rate = 2^{-5}, max length = 220.

Iteration	Model	Hyper parameters	Training RMSE	Validation RMSE	Test RMSE
1	BERT base	epoch = 3, learning rate = 10^{-4}, max length = 180	0.5604	0.6282	0.5733
2	BERT base	epoch = 3, learning rate = 10^{-4}, max length = 200	0.5322	0.6260	0.5938
3	BERT base	epoch = 3, learning rate = 10^{-4}, max length = 220	0.5592	0.5871	0.5378
4	BERT base	epoch = 2, learning rate = 10^{-4}, max length = 220	0.6802	0.7388	0.7224
5	BERT base	epoch = 4, learning rate = 10^{-4}, max length = 220	0.4674	0.8041	0.7610
6	BERT base	epoch = 3, learning rate = 2^{-5}, max length = 220	0.4943	0.5630	0.5253
7	BERT base	epoch = 3, learning rate = 5^{-5}, max length = 220	0.4341	0.5877	0.5509

4.2 Model evaluation

Once we have acquired the optimal hyperparameters for each model from hyperparameter tuning, the model performance was evaluated by training the models with its corresponding set of optimal hyperparameters and predicting the training and true test sets with the model. The table below summarizes the training and test RMSE for each model and the models are ranked based on their RMSEs.

Rank	Model	Training RMSE	Testing RMSE
1	BERT	0.4943	0.5253
2	Linear Regression	0.6910	0.7123
3	Support Vector Machine	0.6958	0.7164

Based on the observations above, BERT consistently produces the lowest RMSE among all models, which leads to our conclusion that it is the best predictive model in terms of our model evaluation metric.

5. Conclusion & Future work

In this project, we aimed to solve Kaggle's Commonlit readability prediction problem by three different learning algorithms, linear regression, SVM and BERT, in the order of their respective model complexity. Linear regression is always a good starting point and benchmarking model before searching for more advanced learning algorithms. Although linear regression does not produce the minimal test error among three algorithms, it serves the best to the host's initiative - looking for an interpretable and explicit readability formula. We selected two other learning algorithms SVM and BERT because they are popular choices in the related literature. While BERT performs the best in prediction, BERT is less efficient and interpretable. Due to BERT's intensive computational requirement, we had to run the entire BERT modelling on Google Colab. To that end, linear regression may be the best model in real practice if the predictive error is not the sole selection criterion.

For future work, there are several improvements we can make on the BERT modelling to enhance its predictive performance, the main ones are:

- Try different BERT models that are pre-trained on different domain-specific corpora
- Fine tune the pre-trained BERT models to learn the training excerpt
- Test BERT large model (24 layers of transformer blocks + 340 million parameters)
- Compare the performance of using different transformer layers for word embeddings [13]
- Study the impact of random seed on the model performance

6. Reference

1. Awad, M., Khanna, R. (2015) Support Vector Machines for Classification. *Efficient Learning Machines.* Apress, Berkeley, CA.
2. Bengoetxea, K., Gonzalez-Dios, I., Martínez, A. A. (2020). AzterTest: Open source linguistic and stylistic analysis tool.
3. Brodbeck, W. J. (2020). The effect of readability on simple linear regression.
4. Cohmetrix.memphis.edu. 2021. *Coh-Metrix version 3.0 indices*. [online] Available at: <http://cohmetrix.memphis.edu/cohmetrixhome/documentation_indices.html> [Accessed 1 August 2021].
5. Crossley, S. A., Skalicky, S., & Dascalu, M. (2019). Moving beyond classic readability formulas: New methods and new models. *Journal of Research in Reading*. 42 (3-4), 541-561.
6. En.wikipedia.org. 2020. *Coh-Metrix*. [online] Available at: <https://en.wikipedia.org/wiki/Coh-Metrix> [Accessed 1 August 2021].
7. En.wikipedia.org. 2021. *Dale–Chall readability formula - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Dale–Chall_readability_formula> [Accessed 1 August 2021].
8. Franc¸ois, T., Miltsakaki, E. (2012). Do NLP and machine learning improve traditional readability formulas? *Association for Computational Linguistics.*
9. Kaggle.com. 2021. *BERT - In Depth Understanding*. [online] Available at: <https://www.kaggle.com/mdfahimreshm/bert-in-depth-understanding> [Accessed 1 August 2021].
10. Kaggle.com. 2021. *CommonLit Readability Prize | Kaggle*. [online] Available at: <https://www.kaggle.com/c/commonlitreadabilityprize/discussion/240423> [Accessed 1 August 2021].
11. Kuhn, M., Johnson, K. (Eds). (2018). *Applied Predictive Modelling.* Springer.
12. Liu, Y. (2020). Assessing text readability and quality with language models.
13. Medium. 2019. *NLP: Contextualized word embeddings from BERT*. [online] Available at: <https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b> [Accessed 1 August 2021].
14. Meng, C., Chen, M., Mao, J., Neville, J. (2021). ReadNet: A Hierarchical Transformer Framework for Web Article Readability Analysis.
15. Sun C., Qiu X., Xu Y., Huang X. (2019) How to Fine-Tune BERT for Text Classification?. In: Sun M., Huang X., Ji H., Liu Z., Liu Y. (eds) Chinese Computational Linguistics. CCL 2019. Lecture Notes in Computer Science, vol 11856. Springer, Cham. https://doi.org/10.1007/978-3-030-32381-3_16
16. TextCompare. 2021. *Linsear Write Readability Score Calculator Online*. [online] Available at: <https://www.textcompare.org/readability/linsear-write/> [Accessed 1 August 2021].
17. Turner, H., Firth, D. (2020). Bradley Terry Model in R: TheBradleyTerry2 Package.

7. Appendix

7.1 Feature glossary

Name	Description
adj_density	Adjective Density
adj_ttr	AdjTTR (Adj Type-Token Ratio)
adv_density	Adverb Density
adv_ttr	AdvTTR (Adv Type-Token Ratio)
adversative_connectives	Adversative/contrastive connectives
adversative_connectives_incidence	Adversative/contrastive connectives (incidence per 1000 words)
agentless_passive_incidence	Number of agentless passive voice verbs (incidence per 1000 words)
all_connectives	Number of connectives
all_connectives_incidence	Number of connectives (incidence per 1000 words)
argument_overlap_adjacent	Argument overlap adjacent sentences binary mean (CRFAOl)
argument_overlap_all	Argument overlap all of the sentences in a paragraph or text binary mean (CRFAOa)
auto	Automated readability index
causal_connectives	Causal connectives
causal_connectives_incidence	Causal connectives (incidence per 1000 words)
coleman	Coleman-Liau index
conditional_connectives	Conditional connectives
conditional_connectives_incidence	Conditional connectives (incidence per 1000 words)
content_overlap_adjacent_mean	Content word overlap adjacent sentences proportional mean (CRFCWO1)
content_overlap_adjacent_std	Content word overlap adjacent sentences proportional standard deviation (CRFCWO1d)
content_overlap_all_mean	Content word overlap all of the sentences in a paragraph or text proportional mean (CRFCWOa)
content_overlap_all_std	Content word overlap all of the sentences in a paragraph or text proportional standard deviation (CRFCWOad)
content_ttr	CTTR (Content Type-Token Ratio)
dalechall	Dale-Chall readability score
flesch	Flesch readability ease
gerund_incidence	Number of verbs in gerund form (incidence per 1000 words)
gunning	Gunning fog index
honore	Honore Lexical Density
hypernymy_index	Mean hypernym values of nouns and verbs in the WordNet lexicon
hypernymy_nouns_index	Mean hypernym values of nouns in the WordNet lexicon
hypernymy_verbs_index	Mean hypernym values of verbs in the WordNet lexicon
infinitive_incidence	Number of verbs in infinitive form (incidence per 1000

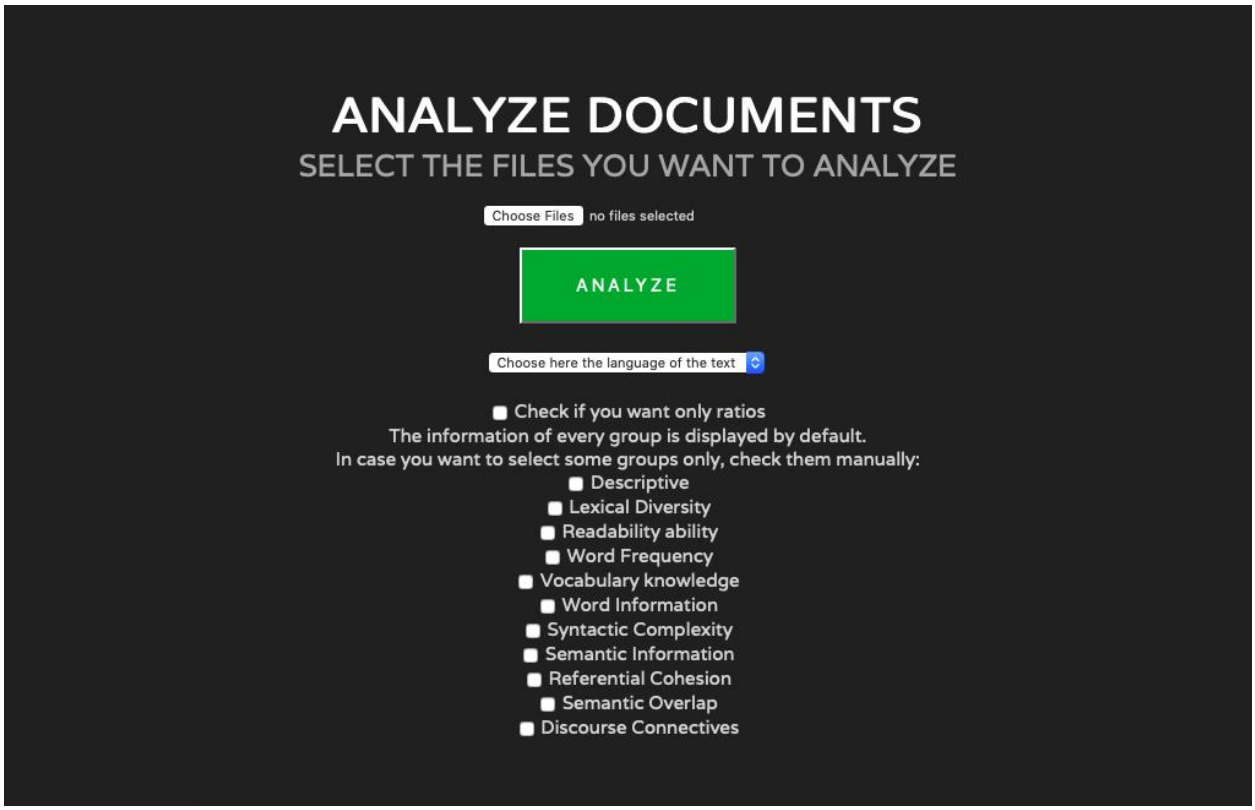
	words)
kincaid	Flesch-Kincaid grade level
left_embeddedness	Left embeddedness (Mean of number of words before the main verb) (SYNLE)
lemma_adj_ttr	LAdjTTR (Lemma Adj Type-Token Ratio)
lemma_adv_ttr	LAdvTTR (Lemma Adv Type-Token Ratio)
lemma_content_ttr	LCTTR (Lemma Content Type-Token Ratio)
lemma_nttr	LNTTR (Lemma Noun Type-Token Ratio)
lemma_ttr	LSTTR (Lemma Simple Type-Token Ratio)
lemma_vttr	LVTTR (Lemma Verb Type-Token Ratio)
lemmas_length_mean	Mean number of letters (length) in lemmas
lemmas_length_std	Standard deviation of letters (length) in lemmas
lexical_density	Lexical Density
linsear	Linsear Write formula
lix	Measure the difficulty of the text by Swedish scholar Carl-Hugo Bj rnsson
logical_connectives	Logical connectives
logical_connectives_incidence	Logical connectives (incidence per 1000 words)
maas	Maas Lexical Density
mean_depth_per_sentence	Mean of the number of levels of dependency tree (Depth)
mean_distinct_rare	Mean of distinct rare lexical words
mean_np_per_sentence	Mean of the number of NPs per sentence
mean_propositions_per_sentence	Mean of the number of propositions per sentence
mean_rare	Mean of rare lexical words
mean_vp_per_sentence	Mean of the number of VPs per sentence
min_wf_per_sentence	Minimum word frequency per sentence (mean)
mtld	Measure of Textual Lexical Diversity (MTLD)
negation_incidence	Number of negative words (incidence per 1000 words)
noun_density	Noun Density
noun_overlap_adjacent	Noun overlap adjacent sentences binary mean (CRFNOl)
noun_overlap_all	Noun overlap all of the sentences in a paragraph or text binary mean (CRFNOa)
noun_phrase_density_incidence	Noun phrase density incidence (DRNP)
nttr	NTTR (Noun Type-Token Ratio)
num_a1_words	Number of A1 vocabulary in the text
num_a1_words_incidence	Incidence score of A1 vocabulary (per 1000 words)
num_a2_words	Number of A2 vocabulary in the text
num_a2_words_incidence	Incidence score of A2 vocabulary (per 1000 words)
num_adj	Number of adjectives
num_adj_incidence	Number of adjectives (incidence per 1000 words)
num_adv	Number of adverbs
num_adv_incidence	Number of adverbs (incidence per 1000 words)
num_agentless	Number of agentless passive voice verbs
num_b1_words	Number of B1 vocabulary in the text
num_b1_words_incidence	Incidence score of B1 vocabulary (per 1000 words)
num_b2_words	Number of B2 vocabulary in the text
num_b2_words_incidence	Incidence score of B2 vocabulary (per 1000 words)

num_c1_words	Number of C1 vocabulary in the text
num_c1_words_incidence	Incidence score of C1 vocabulary (per 1000 words)
num_content_words_not_a1_c1_words	Number of content words not in A1-C1 vocabulary
num_content_words_not_a1_c1_words_incidence	Incidence score of content words not in A1-C1 vocabulary (per 1000 words)
num_decendents_noun_phrase	Number of decendents per noun phrase (mean)
num_dif_rare_words	Number of distinct rare content words
num_dif_rare_words_incidence	Number of distinct rare content words (incidence per 1000 words)
num_different_forms	Number of distinct words (total)
num_different_forms_incidence	Number of distinct words (incidence per 1000 words)
num_first_pers_pron	Number of pronouns in first person
num_first_pers_pron_incidence	Incidence score of pronouns in first person (per 1000 words)
num_first_pers_sing_pron	Number of pronouns in first person singular
num_first_pers_sing_pron_incidence	Incidence score of pronouns in first person singular (per 1000 words)
num_future	Number of verbs in future tense
num_future_incidence	Number of verbs in future tense (incidence per 1000 words)
num_ger	Number of verbs in gerund form
num_impera	Number of verbs in imperative mood
num_impera_incidence	Number of verbs in imperative mood (incidence per 1000 words)
num_indic	Number of verbs in indicative mood
num_indic_incidence	Number of verbs in indicative mood (incidence per 1000 words)
num_inf	Number of verbs in infinitive form
num_lexic_words	Number of content words
num_lexic_words_incidence	Number of content words (incidence per 1000 words)
num_longwords	Number of long words
num_modifiers_noun_phrase	Number of modifiers per noun phrase (mean) (SYNNP)
num_neg	Number of negative words
num_noun	Number of nouns
num_noun_incidence	Number of nouns (incidence per 1000 words)
num_paragraphs	Number of paragraphs (total)
num_paragraphs_incidence	Number of paragraphs (incidence per 1000 words)
num_pass	Number of passive voice verbs
num_pass_incidence	Number of passive voice verbs (incidence per 1000 words)
num_pass_mean	Mean of passive voice verbs
num_past	Number of verbs in past tense
num_past_incidence	Number of verbs in past tense (incidence per 1000 words)
num_past_irregular	Number of irregular verbs in past tense
num_past_irregular_incidence	Number of irregular verbs in past tense (incidence per 1000 words)

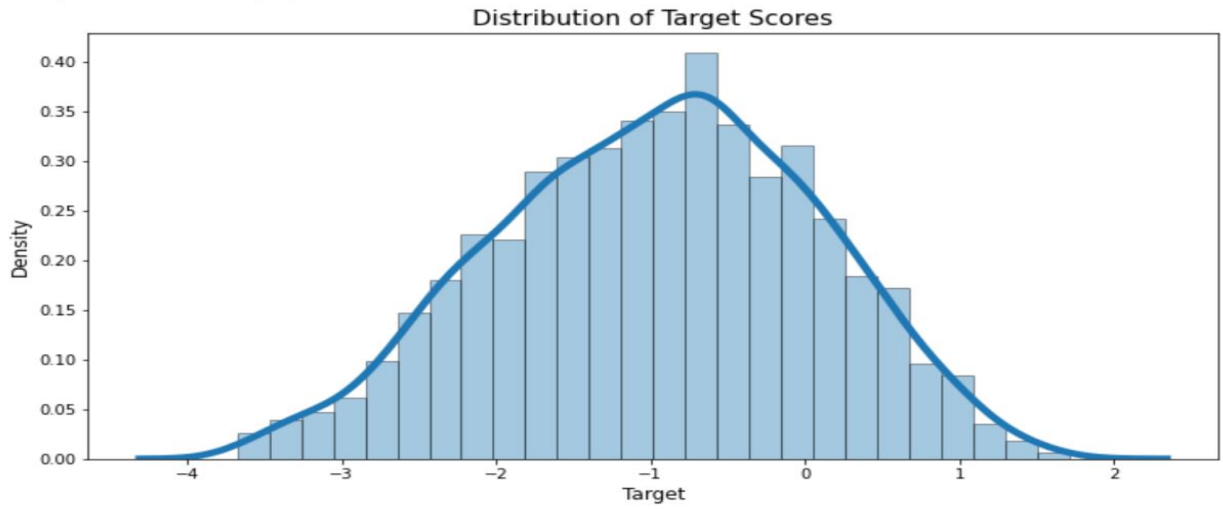
num_past_irregular_mean	Mean of irregular verbs in past tense in relation to the number of verbs in past tense
num_personal_pronouns	Number of personal pronouns
num_personal_pronouns_incidence	Incidence score of pronouns (per 1000 words)
num_pres	Number of verbs in present tense
num_pres_incidence	Number of verbs in present tense (incidence per 1000 words)
num_proper_noun	Number of proper nouns
num_proper_noun_incidence	Number of proper nouns (incidence per 1000 words)
num_punct_marks_per_sentence	Punctuation marks per sentence (mean)
num_rare_adj	Number of rare adjectives
num_rare_adj_incidence	Number of rare adjectives (incidence per 1000 words)
num_rare_advb	Number of rare adverbs
num_rare_advb_incidence	Number of rare adverbs (incidence per 1000 words)
num_rare_nouns	Number of rare nouns
num_rare_nouns_incidence	Number of rare nouns (incidence per 1000 words)
num_rare_verbs	Number of rare verbs
num_rare_verbs_incidence	Number of rare verbs (incidence per 1000 words)
num_rare_words	Number of rare content words
num_rare_words_incidence	Number of rare content words (incidence per 1000 words)
num_rel_subord	Number of relative subordinate clauses
num_rel_subord_incidence	Number of relative subordinate clauses (incidence per 1000 words)
num_sentences	Number of sentences (total)
num_sentences_incidence	Number of sentences (incidence per 1000 words)
num_subord	Number of subordinate clauses
num_subord_incidence	Number of subordinate clauses (incidence per 1000 words)
num_syllables_words_mean	Mean number of syllables (length) in words
num_syllables_words_std	Standard deviation of the mean number of syllables in words
num_third_pers_pron	Number of pronouns in third person
num_third_pers_pron_incidence	Incidence score of pronouns in third person (per 1000 words)
num_total_prop	Number of propositions
num_verb	Number of verbs
num_verb_incidence	Number of verbs (incidence per 1000 words)
num_words	Number of words (total)
num_words_with_punct	Number of words with punctuation (total)
polysemic_index	Mean values of polysemy in the WordNet lexicon
ratio_proper_nouns_per_nouns	Ratio of proper nouns for all nouns(proper and common nouns)
rix	Anderson’s readability index
sentences_length_mean	Number of words (length) in sentences (mean)
sentences_length_no_stopwords_mean	Number of words (length) of sentences without stopwords (mean)

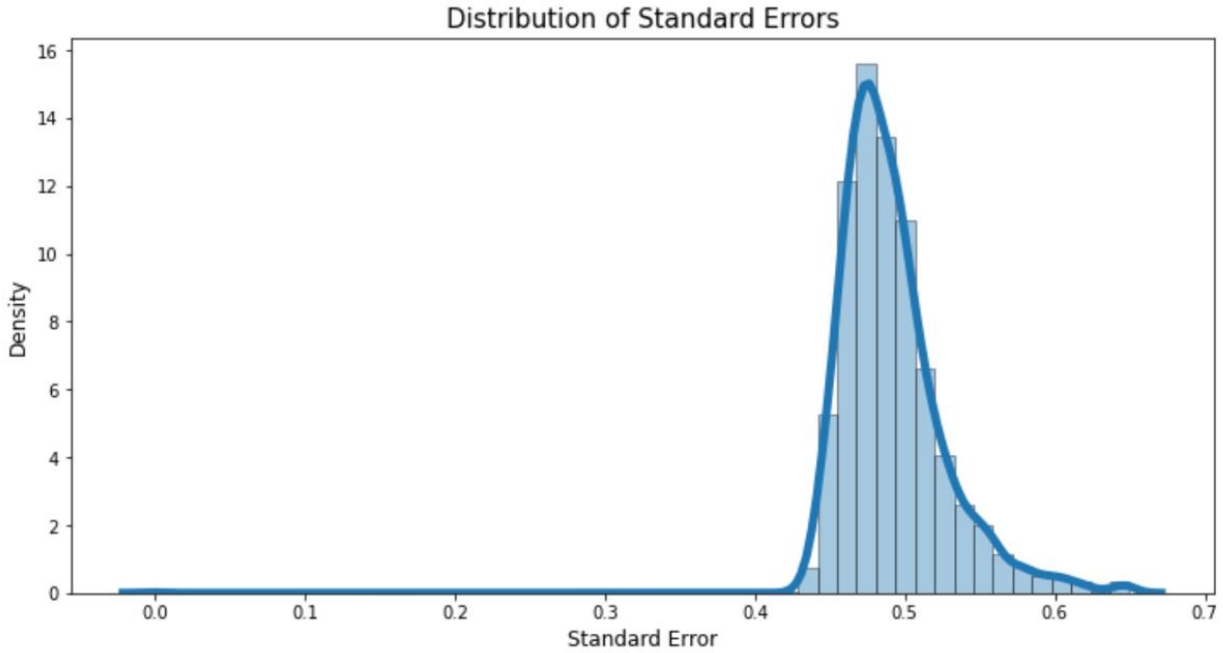
sentences_length_no_stopwords_std	Number of words (length) of sentences without stopwords (standard deviation)
sentences_length_std	Number of words (length) in sentences (standard deviation)
sentences_per_paragraph_mean	Length of paragraphs (mean)
sentences_per_paragraph_std	Standard deviation of length of paragraphs
similarity_adjacent_mean	Semantic Similarity between adjacent sentences (mean)
similarity_adjacent_par_mean	Semantic Similarity between adjacent paragraphs (mean)
similarity_adjacent_par_std	Semantic Similarity between adjacent paragraphs (standard deviation)
similarity_adjacent_std	Semantic Similarity between adjacent sentences (standard deviation)
similarity_pairs_par_mean	Semantic Similarity between all possible pairs of sentences in a paragraph (mean)
similarity_pairs_par_std	Semantic Similarity between all possible pairs of sentences in a paragraph (standard deviation)
simple_ttr	STTR (Simple Type-Token Ratio)
smog	Simple Measure Of Gobbledygook (SMOG) grade
stem_overlap_adjacent	Stem overlap adjacent sentences binary mean (CRFSOl)
stem_overlap_all	Stem overlap all of the sentences in a paragraph or text binary mean (CRFSOa)
temporal_connectives	Temporal connectives
temporal_connectives_incidence	Temporal connectives (incidence per 1000 words)
verb_density	Verb Density
verb_phrase_density_incidence	Verb phrase density incidence (DRVP)
vttr	VTTR (Verb Type-Token Ratio)
words_length_mean	Mean number of letters (length) in words
words_length_no_stopwords_mean	Mean number of letters (length) in words without stopwords
words_length_no_stopwords_std	Standard deviation of the mean number of letter in words without stopwords
words_length_std	Standard deviation of number of letters in words

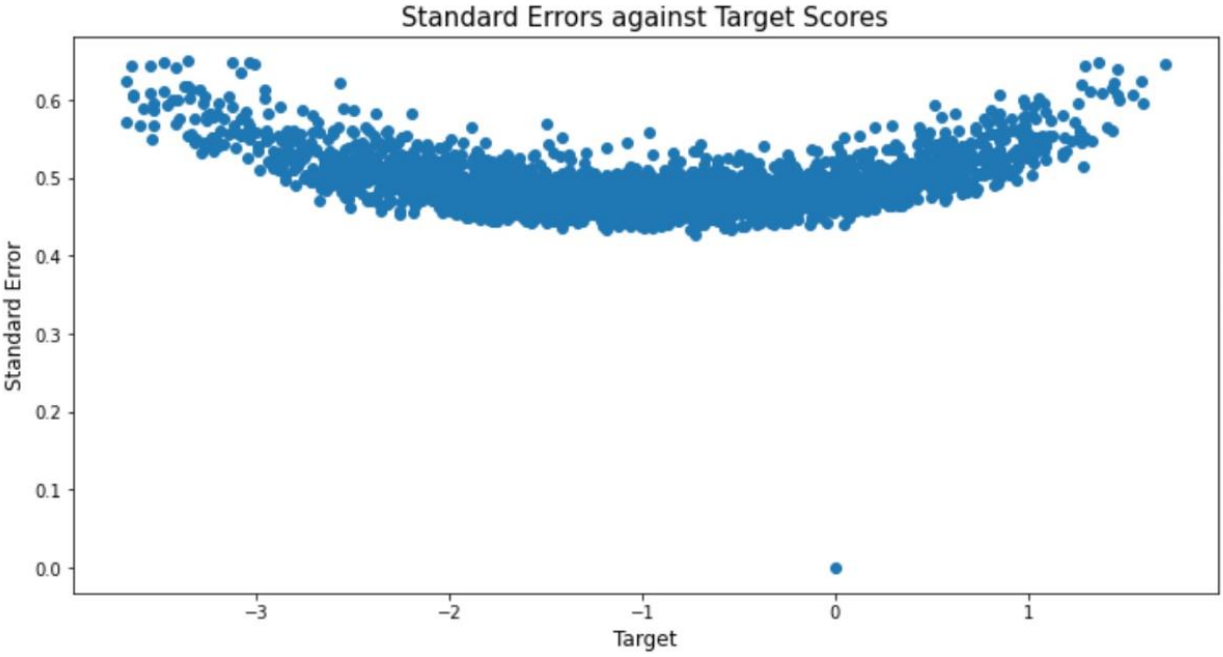
7.2 AzterTest webtool interface



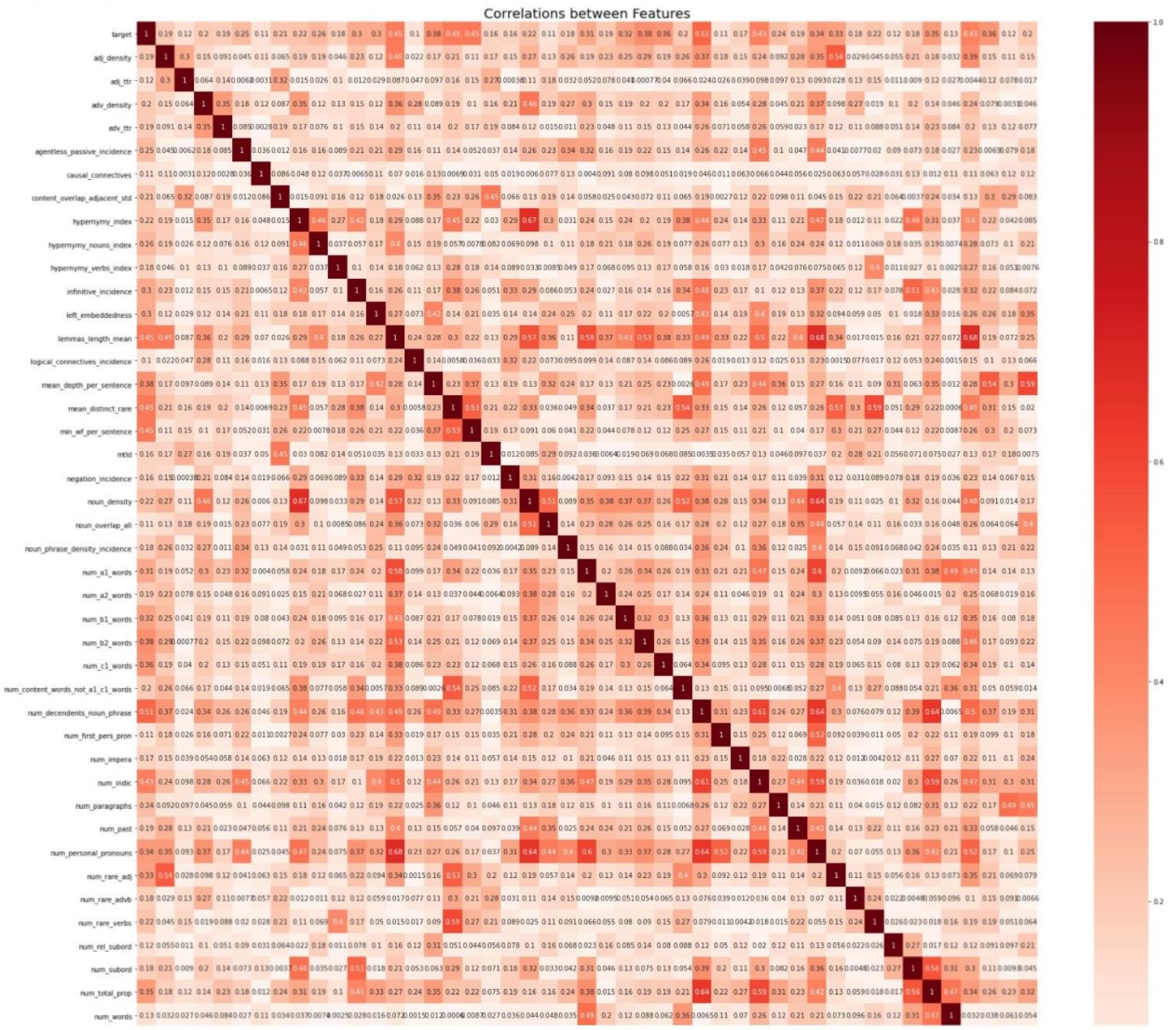
7.3 Analysis of the distribution of target scores and standard errors and their relationship




Distribution of Standard Errors
Density
Standard Error
0
2
4
6
8
10
12
14
16
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7


Standard Errors against Target Scores
Standard Error
Target
0.0
0.1
0.2
0.3
0.4
0.5
0.6
−3
−2
−1
0
1

7.4 Analysis on the correlation between features and target scores and the collinearity between all features



7.5 Description of machine learning algorithms adopted in the project

7.5.1 Linear Regression

Linear regression is a linear model that learns the relationship between one or more explanatory variables and a target variable. In this case, which we have more than one feature, we will be fitting a multivariate linear regression model to our data. The model predicts the continuous target outcomes by estimating the unknown regression coefficients. Before fitting linear regression model, there are a few assumptions can be made, which are expressed as follows [3]:

- There is a linear association between the explanatory variables and the response variable.
- There is a low degree of collinearity between explanatory variables.
- Standard errors of regression are normally distributed.

In essence, one of the most common methods for fitting a regression model is least-squares approximation. This approach approximates the results of linear regression by minimizing the sum of squares of residuals of the observed and predicted target values.

7.5.2 SVM (Support Vector Machine)

SVM is a sparse kernel decision machine learning algorithm that has been widely used in classification and regression analyses due to its flexibility of working for both linear and non-linear problems. Since we are dealing with real-valued target scores, our main focus here will be on Support Vector Regression (SVR). The idea that works behind SVR is it fits a line or a hyperplane (for multidimensional space) that minimizes the loss function by using kernel trick, which is a method that entails lifting the feature space to an infinite dimensional space via kernel functions, including polynomial, sigmoid and radial basis function (RBF), when a feature space seemingly is not best served with linear regression [1].

7.5.3 BERT (Bidirectional Encoder Representation from Transformers)

Lately BERT has been the most prevalent NLP model in wide applications for apparent reasons [9]:

- Not dependent on the text normalisation (removal of stop words, lemmatization and etc) or extraction of explicit features
- Structured and comprehensive learning of semantic context by Masked Language Modelling and Next Sentence Prediction
- Transfer learning takes advantage of pretrained models using domain-specific and high quality corpora, allowing users to build robust model on small-sized dataset