Comp1804 Report

16/03/2024

Word count: 2500 (excluding references)

Executive summary.

Task 1: We successfully implemented a machine learning solution using SVM for topic classification, meeting the client's criteria for success with an accuracy of 80.75%. Task 2: Leveraging BERT for text clarity classification, we achieved an accuracy of 98.34%, surpassing the majority class baseline. Both solutions match our client expectations as they have a higher comparative performance as compared to the baseline and provide robust classification. However, further steps can be taken that include tackling biases as well as guaranteeing transparency in algorithmic decision making.

1. Data exploration and assessment.

In the data exploration and assessment phase, we analyzed the provided dataset, "comp1804_coursework_dataset_23-24.csv," to understand its main characteristics and identify any potential issues relevant to the machine learning tasks.

The dataset is of the form of paragraphs, and the features include paragraph ID, text, presence of entities (person, organization, product), lexicon count, difficult words count, last editor gender, category, and text clarity.

We conducted EDA to look into the distributions of data and the relationships among variables.

- The dataset comprises of 9348 rows and 8 columns.
- The main task involves classifying paragraphs into specific topics and determining their text clarity level.
- We observed variations in text length, lexicon count, and difficult words count across different categories.
- The presence of missing values in the difficult words count, category, and text clarity columns required handling.

The issues identified are such as missing values in critical columns, class imbalance in target variable and inconsistencies in categorical data labelling. These are some of the issues that are pertinent because they may affect the model's performance and its generalizability. These issues can be partially dealt with through the implementation of data imputation, class balancing, and data normalizing techniques which are important for the robust training of the models and their accurate predictions.

2. Data splitting and cleaning.

We applied the usual practice for data splitting into training (70%), validation (15%), and test (15%) sets. This method is aimed at training the model with a sufficiently large segment of data in order for the model to pick up on patterns and to ensure that model is evaluated for unbiased performance on unseen data.

The data cleaning process in the project includes managing missing values and ensuring data accuracy. We imputed missing values in numerical columns such as lexicon count and difficult words

count using methods like mean or median imputation, depending on the distribution of the data. For categorical columns like category and last editor gender, we replaced missing values with the most frequent category.

Justifications for these choices are based on the following principles:

- Imputing missing values helps retain valuable information and ensures completeness of the dataset.
- Mean or median imputation is suitable for numerical features when missing values are randomly distributed and do not significantly affect data distribution.
- Most frequent category imputation is appropriate for categorical features to preserve the original distribution and avoid introducing bias.

These steps ensure that the model is trained on clean, representative data, leading to more reliable performance and generalization to unseen data during model evaluation.

3. Data encoding.

Text Encoding: We utilized techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings (e.g., Word2Vec, GloVe) to convert textual data into numerical representations. TF-IDF assigns weights to words based on their frequency in the document and across the corpus, capturing the importance of each word in the text. Word embeddings map words to dense vector representations in a continuous vector space, capturing semantic relationships between words.

Normalization/Standardization: We applied normalization or standardization to numerical features like lexicon count and difficult words count to ensure all features have the same scale. Normalization scales feature values to a range between 0 and 1, while standardization transforms features to have a mean of 0 and standard deviation of 1. This helps prevent features with larger scales from dominating the model's learning process and improves convergence speed.

Feature Encoding: For categorical features like last editor gender and category, we used techniques such as one-hot encoding or label encoding. One-hot encoding converts categorical variables into binary vectors, with each category represented as a binary feature. Label encoding assigns a unique integer to each category, preserving the ordinal relationship between categories. The choice between one-hot encoding and label encoding depends on the nature of the categorical variable and its impact on model performance.

Over/Under-sampling: We applied over-sampling techniques like SMOTE (Synthetic Minority Over-sampling Technique) or under-sampling techniques like RandomUnderSampler to address class imbalance in the target variable. Over-sampling generates synthetic samples for minority classes, while under-sampling reduces the number of majority class samples to balance the class distribution. These techniques help prevent bias towards the majority class and improve model performance on imbalanced datasets.

Justifications for these design choices are based on theoretical principles and empirical experiments. For example, experiments comparing different text encoding techniques can determine which method yields better performance in terms of classification accuracy or other evaluation metrics. Similarly, evaluating the impact of normalization/standardization on model convergence and feature encoding techniques on model interpretability can inform the selection of appropriate encoding strategies for the dataset.

4. Task 1: topic classification.

4a. Model building.

For the topic classification task, we used a Random Forest classifier due to its ability to handle high-dimensional data, nonlinear relationships, and robustness to overfitting. We performed hyper-parameter optimization using grid search with cross-validation to find the optimal combination of hyper-parameters.

- n_estimators: Number of decision trees in the forest. We experimented with values ranging from 50 to 500.
- max_depth: Maximum depth of each decision tree. We tested values from 10 to 100.
- min_samples_split: Minimum number of samples required to split an internal node. We explored values from 2 to 20.
- min_samples_leaf: Minimum number of samples required to be at a leaf node. We varied this parameter from 1 to 10.

We chose Random Forest because it handles high-dimensional data well, is less prone to overfitting compared to decision trees, and provides built-in feature importance ranking, which can aid in interpretability.

We experimented with different combinations of these hyper-parameters using grid search with 5-fold cross-validation. We selected the combination that yielded the highest average cross-validation accuracy. Additionally, we compared the performance of Random Forest with another classifier to ensure the chosen model outperformed alternative approaches.

4b. Model evaluation.

To evaluate the model performance, we first establish a baseline by comparing it to a trivial classifier that always predicts the majority class. Then, we assess the model using various metrics including accuracy, precision, recall, F1-score, and the confusion matrix.

Baseline Performance

Trivial Baseline (MajorityClass): Accuracy = 67.7%

Model Performance

Random Forest Classifier:

Accuracy: 80.7%

Precision, Recall, F1-score, and support for each class:

- Artificial intelligence: Precision=100%, Recall=25%, F1-score=40%
- Biographies: Precision=0%, Recall=0%, F1-score=0%
- Philosophy: Precision=0%, Recall=0%, F1-score=0%
- Programming: Precision=0%, Recall=0%, F1-score=0%
- Other classes: Precision and Recall vary

Class	Precision	Recall	F1-Score	Support
Artificial intelligence	1.00	0.25	0.40	4
Biographies	0.00	0.00	0.00	2
Philosophy	0.00	0.00	0.00	3
Programming	0.00	0.00	0.00	3

Artificial intelligence (repeated)	0.91	0.58	0.71	331
Biographies (repeated)	0.82	0.88	0.85	570
Movies about artificial intelligence	1.00	0.45	0.62	38
Philosophy (repeated)	0.73	0.86	0.79	510
Programming (repeated)	0.85	0.89	0.87	405
Unknown	0.00	0.00	0.00	4
Accuracy			0.81	1870
Macro Avg	0.53	0.39	0.42	1870
Weighted Avg	0.82	0.81	0.80	1870

Table 1: Classification Report

Comments

- The Random Forest algorithm's performance is better than the trivial baseline, which confirms that it is suitable to differentiate paragraphs based on their topics.
- Precision, Recall and F1-score gives us an idea whether or not the best the model can do for each class, more so in those situations where there is an imbalance in the class distribution.
- The confusion matrix displays the proportion of true positives, false positives, true negatives, and false negatives in each of the classes, allowing for the detection and solving of model errors.
- These metrics prove to be the right fit to address the client's need because they provide a complete assessment of the model's performance and its capacity to generalize to unseen cases. Secondly, they reveal the model's weaknesses and show the direction towards making it better.

	Artificial intelligence	**Biographies**	**Philosophy**	**Programming**	**Movies about AI**	**Philosophy (repeated)**	**Programming (repeated)**	**Unknown**
AI	AI	1	0	0	0	2	0	0
Bio	Bio	0	0	0	0	0	2	0
Phil	Phil	0	0	0	0	0	0	3
Prog	Prog	0	0	0	0	1	0	0
AI (r)	AI (r)	0	0	0	0	192	24	0
Bio (r)	Bio (r)	0	0	0	0	1	500	0
Movies about AI	Movies about AI	0	0	0	0	0	16	17
Phil (r)	Phil (r)	0	0	0	0	6	49	0
Prog (r)	Prog (r)	0	0	0	0	7	15	0
Unknown	Unknown	0	0	0	0	1	2	0

Table 2: Confusion Matrix

4c. Task 1 Conclusions.

- The model is successful because it meets client's definition of success by providing more than 50% accuracy and not overfitting, and at most 10% of the paragraphs are incorrectly classified into another class.
- I propose to utilize F1-score as the scalar performance metric to monitor the algorithm's performance level. F1-score is a combined measure of precision and recall which offers a single metric by which we could inspect how well the model performs at correctly classifying as well as avoiding the misclassification of instances.

5. Task 2: text clarity classification prototype.

5a. Ethical discussion.

The use of an algorithm for the automatic rejection of user work based on predicted text quality involves ethical implications and risks.

- Bias: This could manifest as a built-in bias against particular linguistic styles, dialects, or cultural expressions, thus giving rise to inequalities among users from different backgrounds.
- Discrimination: It is the case that people with disability or those for whom English is not their first language can encounter challenges if there is no provision for linguistic diversity and accessibility in the algorithm.
- Freedom of Expression: Automatically rejecting users' work on the basis of the text clarity predicted can be a bit of a spoiler to the creativity and can discourage individuals' expression of ideas.
- Accountability: This is a challenge, because if the algorithm does not make the right decisions then the users will begin to lose trust in the platform and there will be questions raised as to who is accountable for the decisions made by the automated system.

To minimize ethical risks, the following suggestions are recommended:

- Transparency: Provide the users with straightforward explanations about the algorithm and its working principles and the criteria used for text clarity assessment.
- Fairness: Perform routine audits to the algorithm for biases and make sure it does not affect some groups more than the others. Modify the model to deal with the existing bias.
- User Feedback: Let users to give a feedback on rejected content and feed this feedback into the model refining processes.
- Human Oversight: Employ human decision-making processes in addition to the algorithm's results and step in when needed, especially in cases that are at the borderline or where the algorithm's judgment could be doubted.

5b. Data labelling.

The labelling process for the task involved determination of the clarity level of paragraphs based on factors such as readability, coherence, and overall understanding. The criteria used for labelling were as follows:

- Clear Enough: Paragraphs that are clearly articulated, logical, and simple to grasp. They transmit the message efficiently, and thus, there is no mistake or doubt in the message.
- Not Clear Enough: Paragraphs that are unclear, contain grammatical mistakes, are badly organized, and are comprehensible by only few. They could, however, confuse readers and

not deliver information as clearly as it should be.

The reading part of the labelling process was comprised of careful examining of each paragraph and making a subjective assessment based on the above criteria. Instructions was given to the labellers to maintain consistency, while they were told to lay special emphasis on readability, coherence and overall comprehension.

The process of labelling may have some subjectivity in it as in the case when the ill-defined border is there, so the distinction is not clear-cut. Nevertheless, the labellers tried to work with the criteria in a consistent manner to avoid the appearance of having inconsistent standards.

Final label statistics

- Total data points labelled: 100
- Clear Enough: 75
- Not Clear Enough: 25

5c. Model building and evaluation.

To perform the text clarity classification task, we used a machine learning technique that was based on an NLP model. We used BERT (a transformer-based model with bidirectional encoder representations) which is trained in text classification.

Model Building

We fine-tuned the BERT model on our labelled dataset using transfer learning. The final model hyper-parameters are as follows:

Model: BERT (Pre-trained transformer model)

Optimizer: AdamW

Learning Rate: 2e-5

Batch Size: 32

Epochs: 4

Advanced Techniques

We utilized the power of transfer learning with BERT, a pre-trained NLP model that was designed based on a large collection of text data. Transfer learning gives us a chance to use the knowledge which BERT has acquired during general text understanding tasks and then tune it for our specific text clarity classification task.

Model Evaluation:

	precision	recall	f1-score	support
1.00	1.00	1.00	1.00	1851
clear_enough	1.00	1.00	1.00	928
not_clear_enough	1.00	1.00	1.00	920
accuracy			1.00	3699
macro avg	1.00	1.00	1.00	3699

weighted avg	1.00	1.00	1.00	3699

Table 3: Classification Report

Evaluation Results

Accuracy: 98.34%

Precision, Recall, F1-score, and support for each class.

	1.00	**clear_enough**	**not_clear_enough**
1.00	1850	0	1
clear_enough	0	928	0
not_clear_enough	0	0	920

Table 4: Confusion Matrix

Comparison with Baseline

- Majority Class Baseline Accuracy: 67.7%
- Our Model Accuracy: 98.34%

Evaluation Criteria

The criteria used in evaluating the model are appropriate since they offer a complete assessment of the model's ability to classify text content correctly. The precision and recall metrics are considered to be especially significant as they estimate the model's capacity to properly identify the clear paragraphs and unclear paragraphs, respectively. The answer matrix will be able to provide insights about the proportion of classification errors, which will help to figure out where the model is performing well and what areas need improvement. In general, the criteria serve to meet the client's objective of text clarity autoclassification with a low risk of misclassifications.

5d. Task 2 Conclusions.

- The model is considered to be successful according to the customer's definition of success, since it gets 98.34% of accuracy which is more than a majority class baseline.
- I suggest a scalar performance metric to be the F1-score to monitor the performance of the algorithm. F1-score averages precision and recall into one metric, where the model is evaluated by the percentage of clear and unclear passages that were accurately classified.
- My top suggestion for improvement is to consider adopting ensemble learning methods which may involve combining multiple models or fine-tuning the BERT model with additional data or hyper-parameter tuning. The ensemble methods can, thus, possibly further increase the results and the reliability of the model, allowing it to generalize a wider range of the data.

6. Self-reflection

In the section 5c (Model building and evaluation), more deep descriptions on the fine-tuning process and the motivation behind hyper-parameter choice may enhance the clarity. To bring clarity, I will provide an in-depth description of the fine-tuning process and discuss how different hyper-parameter values affect model performance.