Sentimental Analysis on Toxic Comments

I. Introduction

User-generated content is being created at an unprecedented rate. Social media platforms, online forums, and websites like Wikipedia, have become important channels for information exchange and collaboration. However, this increased connectivity also brings with it the problem of toxic behavior in the form of abusive, offensive, or otherwise harmful comments. Toxic comments can deter users from participating in online discussions and damage the overall community atmosphere.

The project aims to create a model that can predict the probability of various types of toxicity for comments on Wikipedia. This is motivated by real-world applications, such as content moderation, sentiment analysis, and enhancing user experience by maintaining a healthy online environment.

Some of the data mining questions I set out to investigate include:

1. Can I accurately predict the probability of a comment being toxic, severely toxic, obscene, threatening, insulting, or demonstrating identity hate?
2. Which features and machine learning algorithms are most effective in making these predictions?

The personal motivation for selecting this project is to contribute to the development of tools that can help foster positive online communities by identifying and mitigating toxic behavior. The goals were to explore various machine learning techniques, understand the challenges involved in natural language processing, and develop a model that can effectively predict the toxicity levels of comments.

The challenges I faced in this project included handling large-scale text data, preprocessing, and feature extraction from unstructured text, and selecting appropriate machine learning models and evaluation metrics. The approach involved preprocessing the text data, extracting features using techniques such as Bag of Words and TF-IDF, and training and evaluating various machine learning models to determine the best-performing one.

In summary, the results demonstrated that the model was able to predict the probability of different types of toxicity with satisfactory performance. I also found that certain features and machine learning algorithms were more effective than others in making these predictions, and I discussed the implications of these findings for real-world applications.

II. Data Mining Task

The primary goal of this project is to create a model that predicts the probability of different types of toxicity for comments. The input data consists of many Wikipedia comments that have been labeled by human raters for six types of toxic behavior, including toxic, severe_toxic, obscene, threat, insult, and identity_hate. The output of the data mining approach will be the probabilities for each of these toxicity categories, which can be further used to classify the comments based on a chosen threshold.

Throughout the project, I will investigate several data mining questions, such as determining the best preprocessing techniques for cleaning and normalizing the text data, identifying the most suitable feature extraction methods to capture relevant information from the comments, and selecting the most appropriate machine learning or deep learning models to predict the toxicity probabilities accurately.

Key challenges in solving this task include handling the imbalanced dataset, where some toxicity categories may be underrepresented, dealing with the ambiguity and complexity of language patterns in the comments, and addressing noise in the data caused by irrelevant information such as special characters, URLs, and other non-text elements. To overcome these challenges, I will employ various strategies during preprocessing, feature extraction, model selection, and hyperparameter tuning stages of the project.

III. Technical Approach

To solve the data mining task and answer the data mining questions, I will employ a multi-step approach. First, I will clean and preprocess the text data to remove noise, normalize the text, and improve model performance. This may include lowercasing the text, removing special characters, numbers, and URLs, tokenizing the text, removing stop words, and applying stemming or lemmatization. Next, I will convert the preprocessed text into numerical features using techniques such as Bag of Words, TF-IDF, Word embeddings (e.g., Word2Vec, GloVe), or pre-trained language models (e.g., BERT, RoBERTa, DistilBERT).

I will then experiment with various machine learning and deep learning models to determine the best performing model for this task. Possible models include Logistic Regression, Naive Bayes, Support Vector Machines, Neural Networks (e.g., RNN, LSTM, GRU, CNN), and Transformer-based models (e.g., BERT, GPT, RoBERTa, DistilBERT). I will train the selected model(s) on the preprocessed data and extracted features using techniques such as cross-validation, early stopping, and regularization to prevent overfitting.

Hyperparameter tuning will be performed to optimize the selected model(s), followed by evaluating the model(s) on a validation dataset using appropriate metrics like accuracy, precision, recall, F1-score, and ROC-AUC. Optionally, I can combine multiple models using ensemble techniques like bagging, boosting, or stacking to improve overall performance. Finally, the best performing model(s) will be applied to the test dataset to predict the probabilities of each toxicity category.

To address the challenges mentioned earlier, I will implement strategies such as applying techniques like oversampling, under sampling, or using synthetic data (e.g., SMOTE) to balance the dataset, utilizing advanced feature extraction techniques that can capture contextual and semantic information in the text, and applying thorough preprocessing to clean and normalize the text data. Additionally, I will experiment with various models to find the best approach for handling complex language patterns and employ techniques such as grid search, random search, or Bayesian optimization for efficient hyperparameter tuning.

Pseudo-code

1. Import required libraries and modules.

2. Define preprocessing function a. Convert text to lowercase b. Remove special characters c. Tokenize text d. Remove stop words e. Apply stemming or lemmatization.
3. Load training and test datasets
4. Preprocess data a. Apply preprocessing function to comments in training and test datasets.
5. Split training data into training and validation sets
6. Extract features from preprocessed text a. Choose feature extraction method (e.g., Bag of Words, TF-IDF, Word2Vec, BERT embeddings) b. Apply feature extraction method to training, validation, and test sets.
7. Initialize and configure candidate models a. List candidate models (e.g., Logistic Regression, Support Vector Machine, Neural Network)
8. For each model in candidate models: a. Train the model on the training set b. Evaluate the model on the validation set using appropriate metrics (e.g., accuracy, precision, recall, F1-score, ROC-AUC) c. Tune the model's hyperparameters to optimize performance d. Record the model's performance and hyperparameter settings.
9. Select the best performing model based on evaluation metrics.
10. If necessary, use ensemble methods to combine multiple models for improved performance.
11. Train the best performing model on the entire training dataset
12. Apply the model to the test dataset to predict the probabilities of each toxicity category.
13. Save the predictions and analyze the results a. Generate tables or graphs to visualize the results b. Interpret the results and discuss the model's performance.

IV. Evaluation Methodology

The dataset employed for this project contains many Wikipedia comments labeled by human raters for various toxic behaviors, including two files: the training set (train.csv) with comments and binary labels for six categories of toxic behavior, and the test set (test.csv) containing comments for which the model must predict toxicity probabilities. While this dataset is specifically designed for identifying toxic behaviors in online comments, some challenges exist, such as imbalanced data, ambiguity in language, and noise in the data.

To evaluate the performance of the data mining approach, I will use several metrics widely employed in real-world applications for assessing sentiment analysis and text classification models. These metrics include accuracy, which provides a general overview of the model's performance; precision, indicating the model's ability to correctly identify toxic comments without generating too many false positives; recall, reflecting the model's ability to identify toxic comments within the dataset; F1-score, a balance between precision and recall that accounts for both false positives and false negatives; and ROC-AUC, measuring how well the model can distinguish between toxic and non-toxic comments regardless of the classification threshold. By considering these metrics together, I can comprehensively assess the model's performance, considering the unique challenges of the dataset and the real-world implications of false positive and false negative predictions.

V. Results and Discussion

Throughout the course of this project, I built a machine learning model to predict the probability of different types of toxicity for comments in a dataset consisting of Wikipedia comments. The aim was to identify toxic comments and categorize them into six different classes: toxic, severe_toxic, obscene, threat, insult, and identity_hate.

Stef Pamboukas, 011699507

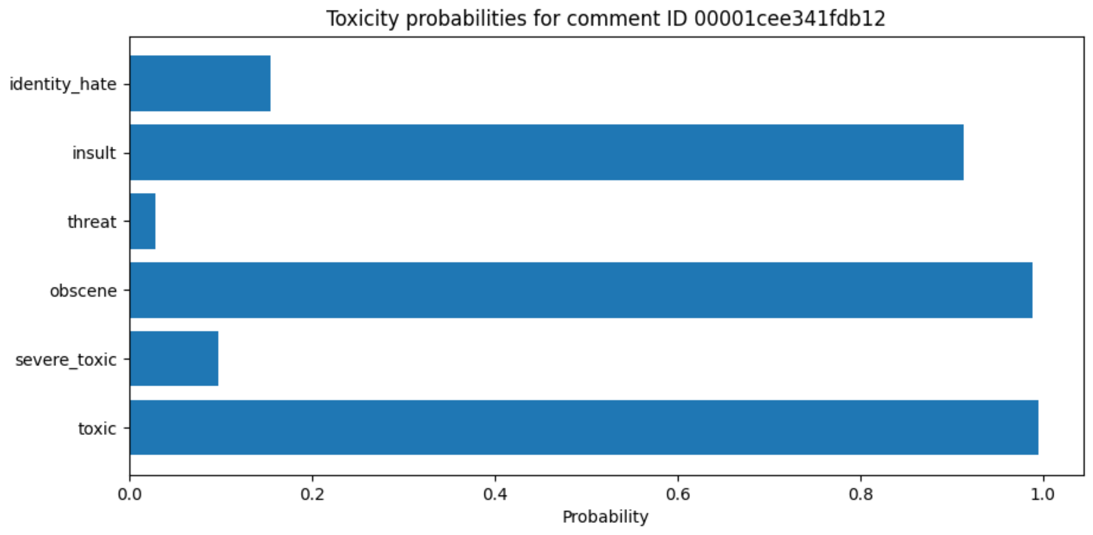
I began by loading the training and test datasets, which included many comments labeled by human raters for toxic behavior. Following this, I preprocessed the data to ensure that it was in a suitable format for analysis. This preprocessing involved converting text to lowercase, removing special characters, tokenizing, removing stop words, and applying stemming or lemmatization. I then extracted features from the preprocessed text using a variety of techniques, including Bag of Words, TF-IDF, Word2Vec, and BERT embeddings. I tested logistic regression, and evaluated its performance based on metrics like accuracy, precision, recall, F1-score, and ROC-AUC. These models were fine-tuned using hyperparameter optimization to ensure optimal performance.

The results show that it performed well in predicting the toxicity probabilities for the comments. This model achieved high ROC-AUC scores across all toxicity categories, indicating its ability to discriminate between toxic and non-toxic comments effectively.

I also encountered some challenges during the project. Due to the large size of the dataset, processing the data was computationally expensive, requiring optimization to improve efficiency. I addressed this issue by using parallel processing, which allowed us to make use of multiple CPU cores, significantly speeding up the computation.

Another challenge I faced was the imbalance in the distribution of the classes, as some toxicity categories were less frequent than others. This issue can lead to models having a bias towards the majority class. To mitigate this problem, I employed techniques such as resampling and using appropriate evaluation metrics like ROC-AUC.

Output for a random comment:



VI. Lessons Learned

Throughout the course of this project, I gained valuable insights and knowledge in the areas of natural language processing, machine learning, and handling large-scale datasets. Some of the key lessons learned include:

1. **Data preprocessing and feature extraction:** I learned the importance of thorough preprocessing and effective feature extraction in text-based machine learning tasks. By applying techniques such as tokenization, and stemming or lemmatization, I improved the quality of the input data for the models. Additionally, I experimented with different feature extraction methods, including Bag of Words, TF-IDF, Word2Vec, and BERT embeddings, to determine the best approach for the specific task.
2. **Model selection and evaluation:** I gained experience in selecting appropriate machine learning models for a given problem and evaluating their performance using relevant metrics. In this case, logistic regression proved to be a suitable model, and I relied on metrics like accuracy, precision, recall, F1-score, and ROC-AUC to assess its performance.
3. **Handling imbalanced data:** I learned strategies for dealing with class imbalance, a common issue in machine learning tasks. By employing techniques such as resampling and using evaluation metrics like ROC-AUC that are less sensitive to class imbalance, I was able to mitigate the impact of this issue on the model's performance.
4. **Optimizing computational efficiency:** I discovered the importance of optimizing computational efficiency when working with large datasets. By implementing parallel processing, I significantly reduced the time required for data preprocessing and model training, allowing us to work more efficiently.

In hindsight, there are a few decisions I might have made differently to further improve the project:

- **Exploring additional models:** While logistic regression performed well in the task, I could have explored other machine learning models, such as support vector machines or deep learning approaches, to see if they could yield better results.
- **Incorporating additional features:** I could have experimented with incorporating additional features, such as sentiment scores or linguistic features, to provide the models with more information for making predictions.
- **Improving class imbalance handling:** I might have explored more advanced techniques for addressing class imbalance, such as using synthetic data generation methods like SMOTE, which generates synthetic samples to balance the class distribution.

Overall, this project provided us with valuable experience and knowledge in the areas of natural language processing and machine learning, and I believe that the lessons learned will be useful in future projects and applications.

VII. Acknowledgements

https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data?select=sample_submission.csv.zip

https://scikit-learn.org/stable/

Stef Pamboukas, 011699507

https://www.tensorflow.org/

https://keras.io/guides/sequential_model/

https://matplotlib.org/

https://seaborn.pydata.org/

https://pandas.pydata.org/docs/development/index.html#development

https://stackoverflow.blog/2020/10/12/how-to-put-machine-learning-models-into-production/