Predictive Modeling for Code Review Question Intent Classification

Rumman Ali*
ali5i@uwindsor.ca
University of Windsor
Windsor, Ontario, Canada

ACM Reference Format:
Rumman Ali. 2018. Predictive Modeling for Code Review Question Intent Classification. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

This assignment focuses on predicting the communicative intention in code review questions. The primary objective is to develop a classifier that can categorize code review questions into top-level categories, such as suggestions, requests, attitudes and emotions, hypothetical scenarios, and rhetorical questions. The assignment includes multiple tasks and variations to evaluate different modeling approaches.

2 PROBLEM DESCRIPTION

In code review processes, understanding the intention behind a question is crucial for effective communication. Different categories of questions may require different responses. Therefore, the assignment aims to build a classifier that can automatically predict the communicative intention in code review questions.

3 DATASET DESCRIPTION

The dataset for this assignment is available in the file icsme-questions-labeled.xlsx, which can be found on the course website. It contains code review questions labeled with their communicative intentions. The dataset will be used for training and evaluating the prediction models.

The dataset provided for this assignment consists of code review questions, each associated with specific labels indicating the intention behind the question. The dataset is structured with four columns:

inline-comment-id: This column, denoted by the header #, uniquely identifies each entry in the dataset. It is a combination of alphanumeric characters and underscores. While it is a crucial identifier, it does not play a role in the model training process due to its numeric nature.

Comment: The 'Comment' column contains numeric values that are not relevant to the machine learning model development and are used solely for reference or organizational purposes. It serves as a sequential identifier for individual comments within the dataset.

Question: The 'Question' column is the primary focus of the dataset. It contains the textual content of code review questions. Each question aims to address a specific aspect of the code under review. However, in cases where there are multiple questions within a single comment box, the label for the question that corresponds to the correct intention has been highlighted in red. Notably, the correct label for the first question in the comment box is found immediately, while subsequent questions are labeled sequentially.

Final Label: The 'Final Label' column is the target variable for this machine learning task. It holds the labels that represent the communicative intention associated with each code review question. The intention labels are categorized into five primary categories, with some of these categories containing additional subcategories. The intention labels are essential for the development of the predictive model, as they guide the model in classifying code review questions.

It is important to emphasize that the 'Inline-Comment-ID' and 'Comment' columns, while serving practical purposes for data organization, do not hold any significance in the process of training the machine learning model. These columns are composed of numerical values and are thus not relevant for feature extraction or classification purposes.

Understanding the dataset structure and the relationship between the 'Question' column and the corresponding 'Final Label' is essential for the successful development of a predictive model to accurately classify code review questions based on their communicative intention.

By observing patterns in the dataset, it is evident that for comment boxes containing multiple questions, the labeling sequence follows a specific pattern, with the initial question being labeled immediately and subsequent questions receiving sequential labels. This observation provides valuable insight into the dataset's labeling methodology and guides the model development process.

In summary, the dataset comprises code review questions, their associated labels indicating communicative intention, and supporting columns for organization, with the 'Question' and 'Final Label' columns being of primary importance for the assignment's machine learning task.

4 EXPERIMENT PROCESS

The assignment involves four main tasks:

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

4.1 Task 1: Model with Words as Features and Text Preprocessing

In this task, we will create a model using words as features. Text preprocessing will be applied to clean and prepare the text data for modeling.

Data Preparation: The initial step involved reading the provided dataset from the XLSX file. Each row of the dataset contains code review questions, with specific questions highlighted in red to indicate their corresponding labels.

Question Selection: From each row in the dataset, the questions with red labels were extracted, and their corresponding labels were recorded. It's important to note that the red-labeled questions were identified as those representing the correct communicative intention.

Label Mapping: The labels associated with the extracted questions were mapped to numeric values. This mapping process was designed to represent the main categories of communicative intention, as mentioned earlier. Subcategories were disregarded for the purpose of simplifying the classification task.

Dataset Split: The dataset was divided into training and testing sets. Specifically, 30% of the data was allocated for testing, while the remaining 70% was designated for training the classification model.

Text Preprocessing: A series of text preprocessing steps were applied to the questions in both the training and testing datasets. These steps included: Conversion of text to lowercase to ensure uniformity. Tokenization to split the text into individual words. Removal of punctuation to focus on the text content. Lemmatization, a process of reducing words to their base form, was applied. Stemming was initially considered but later excluded as it was found to reduce accuracy.

Vectorization: To prepare the text data for machine learning, the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique was employed. This transformed the text into a numerical format suitable for model training.

Model Selection: For the classification task, the Support Vector Classification (SVC) algorithm was chosen as the model. SVC is well-suited for text classification problems and was used to train the model on the preprocessed training data.

Model Training and Prediction: The trained SVC model was then used to make predictions on the test dataset. These predictions were used to evaluate the model's performance in classifying code review questions. Performance Metrics:

To assess the model's effectiveness, key performance metrics were calculated, including: Accuracy: The proportion of correctly classified instances. Classification Report: A comprehensive summary of precision, recall, F1-score, and support for each label. Precision: The ratio of correctly predicted instances of a label to the total predicted instances. Recall: The ratio of correctly predicted instances of a label to the total actual instances of that label.

4.2 Task 2: Model with Words and Additional Feature (No Text Preprocessing)

This task extends Task 1 by including an additional feature alongside words as features. However, no text preprocessing will be performed in this task.

Data Preparation: The initial step involved reading the provided dataset from the XLSX file. Each row of the dataset contains code review questions, with specific questions highlighted in red to indicate their corresponding labels.

Question Selection: From each row in the dataset, the questions with red labels were extracted, and their corresponding labels were recorded. These red-labeled questions were identified as those representing the correct communicative intention.

Label Mapping: The labels associated with the extracted questions were mapped to numeric values. This mapping process was designed to represent the main categories of communicative intention, as mentioned earlier. Subcategories were disregarded for the purpose of simplifying the classification task.

Feature Engineering: A new feature was introduced in this task. The length of each question was calculated and added as a numerical feature. This feature aimed to provide additional information to the classification model.

Vectorization: The TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique was applied to the entire dataset, including the length feature. Vectorization converted both the text content and the length feature into a numerical format suitable for model training.

Dataset Split: The dataset was divided into training and testing sets. Specifically, 30% of the data was allocated for testing, while the remaining 70% was designated for training the classification model.

Model Selection: For the classification task, the Support Vector Classification (SVC) algorithm was chosen as the model. SVC is well-suited for text classification problems and was used to train the model on the vectorized data, including the new length feature.

Model Training and Prediction: The trained SVC model was used to make predictions on the test dataset. These predictions were used to evaluate the model's performance in classifying code review questions based on both the text content and the length feature.

Performance Metrics: To assess the model's effectiveness, key performance metrics were calculated, including: Accuracy: The proportion of correctly classified instances. Classification Report: A comprehensive summary of precision, recall, F1-score, and support for each label. Precision: The ratio of correctly predicted instances of a label to the total predicted instances. Recall: The ratio of correctly predicted instances of a label to the total actual instances of that label.

This comprehensive experiment process was executed to develop a classification model for predicting the communicative intention of code review questions. In Task 2, a new feature, the length of the question, was introduced to enrich the model's input data. The results and findings from this experiment, including the impact of the new feature, will be presented and discussed in the assignment report.

4.3 Task 3: Model with Words and Additional Feature (With Text Preprocessing)

Similar to Task 2, this task includes an additional feature. However, text preprocessing will be applied to the text data.

Data Preparation: The initial step involved reading the provided dataset from the XLSX file. Each row of the dataset contains code review questions, with specific questions highlighted in red to indicate their corresponding labels.

Question Selection: From each row in the dataset, the questions with red labels were extracted, and their corresponding labels were recorded. These red-labeled questions were identified as those representing the correct communicative intention.

Label Mapping: The labels associated with the extracted questions were mapped to numeric values, representing the main categories of communicative intention. Subcategories were disregarded for the purpose of simplifying the classification task.

Text Preprocessing: Text preprocessing steps were applied to the questions in both the training and testing datasets. These steps included: Conversion of text to lowercase to ensure uniformity. Tokenization to split the text into individual words. Removal of punctuation to focus on the text content. Lemmatization, a process of reducing words to their base form, was applied. Stemming was initially considered but later excluded as it was found to reduce accuracy.

Vectorization: After text preprocessing, the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique was applied. This transformed the text into a numerical format suitable for model training, including the text content and the length feature.

Dataset Split: The dataset was divided into training and testing sets. Specifically, 30% of the data was allocated for testing, while the remaining 70% was designated for training the classification model.

Model Selection: For the classification task, the Support Vector Classification (SVC) algorithm was chosen as the model. SVC is well-suited for text classification problems and was used to train the model on the vectorized data.

Model Training and Prediction: The trained SVC model was used to make predictions on the test dataset. These predictions were used to evaluate the model's performance in classifying code review questions based on the preprocessed text content and the length feature.

Performance Metrics: To assess the model's effectiveness, key performance metrics were calculated, including: Accuracy: The proportion of correctly classified instances. Classification Report: A comprehensive summary of precision, recall, F1-score, and support for each label. Precision: The ratio of correctly predicted instances of a label to the total predicted instances. Recall: The ratio of correctly predicted instances of a label to the total actual instances of that label.

This comprehensive experiment process was executed to develop a classification model for predicting the communicative intention of code review questions based on the preprocessed text content and the length feature. The results and findings from this experiment will be presented and discussed in the assignment report.

4.4 Task 4: Model with Words and Additional Feature (With Resampling and Text Preprocessing)

In this task, we will incorporate an additional feature and apply text preprocessing. Additionally, resampling techniques will be used to handle class imbalance if present in the dataset.

Data Preparation: The initial step involved reading the provided dataset from the XLSX file. Each row of the dataset contains code review questions, with specific questions highlighted in red to indicate their corresponding labels.

Question Selection: From each row in the dataset, the questions with red labels were extracted, and their corresponding labels were recorded. These red-labeled questions were identified as those representing the correct communicative intention.

Label Mapping: The labels associated with the extracted questions were mapped to numeric values, representing the main categories of communicative intention. Subcategories were disregarded for the purpose of simplifying the classification task.

Text Preprocessing: Text preprocessing steps were applied to the questions in both the training and testing datasets. These steps included: Conversion of text to lowercase to ensure uniformity. Tokenization to split the text into individual words. Removal of punctuation to focus on the text content. Lemmatization, a process of reducing words to their base form, was applied. Stemming was initially considered but later excluded as it was found to reduce accuracy.

Vectorization: After text preprocessing, the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique was applied. This transformed the text into a numerical format suitable for model training, including the text content and the length feature.

Dataset Split: The dataset was divided into training and testing sets. Specifically, 30% of the data was allocated for testing, while the remaining 70% was designated for training the classification model.

Resampling: To address class imbalance in the training data, a resampling technique was applied. This involved creating synthetic samples for the minority class. While specific code for resampling was used, it involved generating additional instances for under-represented classes to balance the dataset, enhancing the model's ability to learn from all categories.

Model Selection: For the classification task, the Support Vector Classification (SVC) algorithm was chosen as the model. SVC is well-suited for text classification problems and was used to train the model on the vectorized data.

Model Training and Prediction: The trained SVC model was used to make predictions on the test dataset. These predictions were used to evaluate the model's performance in classifying code review questions based on the preprocessed text content and the length feature.

Performance Metrics: To assess the model's effectiveness, key performance metrics were calculated, including: Accuracy: The proportion of correctly classified instances. Classification Report: A comprehensive summary of precision, recall, F1-score, and support for each label. Precision: The ratio of correctly predicted instances of a label to the total predicted instances. Recall: The ratio of correctly

predicted instances of a label to the total actual instances of that label.

5 STUDY RESULTS

The study results will be presented in the final submission, including the performance of each model in terms of accuracy, precision, recall, and F1 score.

Table 1: Classification Results with Words as Features and Text Preprocessing

Class	Metrics			Support
	Precision	Recall	F1 Score	
Class 1	0.73	0.42	0.54	52
Class 2	0.64	0.93	0.75	81
Class 3	0.00	0.00	0.00	9
Class 4	0.00	0.00	0.00	3
Class 5	0.00	0.00	0.00	3
Accuracy			0.66	148
Macro Avg	0.27	0.27	0.26	148
Weighted Avg	0.61	0.66	0.60	148

Table 2: Classification Results with Words and Additional Feature (No Text Preprocessing)

Class	Metrics			Support
	Precision	Recall	F1 Score	
Class 1	0.60	0.58	0.59	52
Class 2	0.65	0.79	0.72	81
Class 3	0.00	0.00	0.00	9
Class 4	0.00	0.00	0.00	3
Class 5	0.00	0.00	0.00	3
Accuracy			0.64	148
Macro Avg	0.25	0.27	0.26	148
Weighted Avg	0.57	0.64	0.60	148

Table 3: Classification Results with Words and Additional Feature (With Text Preprocessing)

Class	Metrics			Support
	Precision	Recall	F1 Score	
Class 1	0.76	0.48	0.59	52
Class 2	0.65	0.93	0.77	81
Class 3	0.00	0.00	0.00	9
Class 4	0.00	0.00	0.00	3
Class 5	0.00	0.00	0.00	3
Accuracy			0.68	148
Macro Avg	0.28	0.28	0.27	148
Weighted Avg	0.62	0.68	0.63	148

Table 4: Classification Results with Words and Additional Feature (With Resampling and Text Preprocessing)

Class	Metrics			Support
	Precision	Recall	F1 Score	
Class 1	0.80	0.54	0.64	52
Class 2	0.67	0.94	0.78	81
Class 3	0.00	0.00	0.00	9
Class 4	0.00	0.00	0.00	3
Class 5	0.00	0.00	0.00	3
Accuracy			0.70	148
Macro Avg	0.29	0.30	0.29	148
Weighted Avg	0.65	0.70	0.65	148

6 CONCLUSION

In conclusion, this assignment involved a series of tasks aimed at developing a classifier for predicting the intention in code review questions. My primary objectives were to explore different modeling approaches and assess their performance.

I began with Task 1, where I implemented a model using words as features and applied text preprocessing techniques. The results provided me with valuable insights into the baseline performance of the model. In Task 2, I introduced a new feature, namely the length of the questions, without employing text preprocessing, allowing me to investigate the impact of additional features on model accuracy.

Moving on to Task 3, I revisited text preprocessing and observed a noticeable improvement in accuracy, highlighting the significance of text preprocessing in natural language processing tasks. In Task 4, I incorporated resampling techniques to balance the dataset, which resulted in mixed outcomes, underscoring the importance of careful selection and fine-tuning of resampling methods.

Throughout the assignment, I encountered several challenges and limitations, including the need for more extensive feature engineering and dealing with class imbalances. Despite these challenges, I achieved promising results, with the models demonstrating the ability to predict the intention in code review questions effectively.

This assignment served as a valuable learning experience, deepening my understanding of the complex nature of code review question intention prediction. It also provided practical insights into software engineering, particularly in code review processes, and contributed to the broader field of natural language processing. Further exploration is warranted, such as experimenting with different machine learning algorithms, refining text preprocessing techniques, and enhancing feature engineering.

7 REFERENCES

[1] M. M. Rahman, C. K. Roy and R. G. Kula, "Predicting Usefulness of Code Review Comments Using Textual Features and Developer Experience," 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), Buenos Aires, Argentina, 2017, pp. 215-226, doi: 10.1109/MSR.2017.17.

[2] Al-Otaibi, Shaha and Al-Rasheed, Amal. (2022). A Review and Comparative Analysis of Sentiment Analysis Techniques. Informatica. 46. 10.31449/inf.v46i6.3991.