Comparing Different Classification and Regression Models to Predict Group Performance

Fantillo Joshua
The University of the Fraser Valley
Abbotsford, Canada
joshua.fantillo@student.ufv.ca

Abstract

This research addresses how different classification and regression models can predict group performance. By using different classification and regression machine learning models, we try to predict the group performance of certain groups in the GAP corpus. After doing multimodal analysis of the group data and running the different machine learning models we were able to compare how the different models predicted how well a group would perform using the groups linguistic features.

Keywords

Group interaction, group performance, multimodal analysis, data augmentation, data modification, semi-supervised learning, classification, regression, machine learning models

I. Introduction

A big part of machine learning is trying to figure out what machine learning model you should use. Many different models have their pros and cons and should be used in certain situations that others should not be used.

In this research we try to successfully determine what machine learning model to use on a smaller corpus called the GAP Corpus [1]. Doing this will help us determine what models should be used for future research.

The research's goals were to try and analyze how more complex machine learning models compare to more basic models on predicting the average group score of the smaller GAP Corpus dataset [1].

To achieve these goals, we use ConvoKit [2] and extract different features from the dataset and ConvoKit [2]. These included features that were already part of the dataset as well as certain linguistic features we had to manually calculate from the dataset. By extracting these features and performing analysis on them we should be able to get an accurate representation on how these machine learning models compare with one another.

II. Multimodal Analysis of Group Performance

In this section we describe different features we extracted from the corpus and how we did it.

A. Dataset Features

The following features were extracted straight from the dataset and required no further change.

Group Worked Well Score: Extracted Group_WW from the metadata in the dataset.
Group Time Expectation Score: Extracted Group_TE from the metadata in the dataset.
Group Time Management Score: Extracted Group_TM from the metadata in the dataset.
Group Efficiency Score: Extracted Group_Eff from the metadata in the dataset.
Group Quality of Work Score: Extracted Group_QW from the metadata in the dataset.
Group Overall Satisfaction Score: Extracted Group_Sat from the metadata in the dataset.
Average Group Score: Extracted AGS from the metadata in the dataset.
Meeting Size: Extracted the meeting size from the dataset.

B. Linguistic Features

The following linguistic features were extracted from the manual transcript in the corpus.

Parsed Sentences: All sentences are parsed using the NLTK word tokenizer [3]. Doing this we extract some features like getting the total number of words used in each group and in the whole corpus as well as total number of sentences used. We also got the average amount of words used per sentence.
Filled Pause: Found the number of filled pauses, 'uh' or 'um', for each group and got rid of empty sentences in the parsed sentences.
Laughs: Found the number of laughs in the conversational dataset.
Random Noises: Found the number of random noises in the conversational dataset.
Cough: Found the number of coughs in the conversational dataset.
Unclear Words: Found the number of unclear words in the conversational dataset.
Sentiment Scores: The sentiment score was found by fetching the 'Sentiment' value in the utterances metadata from ConvoKit [1]. From this we got the average sentiment score over the whole group conversation.
Stop Words: We further cleaned the words by removing stop words from each sentence in the dataset.

Lemmatization: We lemmatized the dataset by using NLTK WordNetLemmatizer [4]. This was done so we could simplify the sentence's words into their root words to accurately get frequencies of the words.

Cosine Similarity: We used sklearn's cosine_similarity [5] to get each sentences similarity with the sentences directly in front and behind of it. After getting each sentences similarity we took the average similarity of all the sentences in each group.

Parts Of Speech Tags: We got the POS Tags for each word in every sentence. To do this we used NLTK's pos_tag [6].

Word and Tag Frequency: We got each word and tags frequency for each group's transcript using the NLTK's FreqDist [7]. By doing this we were able to get the top 100 most common words for each group and top 15 most common tags.

Bag of Words and Tags: From the data we got from word and tag frequencies we were able to get our Bag of Words and Bag of Tags using the most common words (100) and tags (15).

Type Ratio: We got the type token ratio and the type tag ratio for each group in the dataset.

Time Duration: We got the time duration for each utterance. From that we can get average duration of each utterance in the group.

Politeness Score: Using ConvoKit's [1] built in politeness function we were able to get the average politeness score of each group.

III. EXPERIMENTAL SETUP

A. *Corpus*

The corpus this experiment was run on was the UFV's Group Affect and Performance (GAP) Corpus [1] using ConvoKit and its built-in functions. This corpus got individuals into a group of size 2 to 4 to complete a winter survival task. The groups were put into a hypothetical situation that their plane crashed, and they salvaged 15 items from the crash. From those 15 items they needed to rank them in order of most useful to least useful.

The corpus has many features from both the groups and individuals. Some of these features were the utterances, sentiment score, time duration, as well as scores for certain aspects of the group tasks and individual tasks.

These scores were the average group score (AVG), group time expectation (Group_TE), group worked well together (Group_WW), group time management (Group_TM), group efficiency (Group_Eff), group quality of work (Group_QW), and group overall satisfaction with meeting (Group_Sat). All these scores were calculated and saved as well as the same scores for each individual person.

The reason this dataset was chosen is that the set has a well-defined task that ranks both individual and group scores.

In this corpus there are 28 meetings. All these meetings are in English, and all have between 2 to 4 people in the groups. Because the purpose of this research compares different machine learning models the dataset with this corpus is quite small. It should be noted that no secondary data source was used.

B. *Machine Learning Models*

We use 10 different machine learning models in this experiment. These include both the Support Vector Machine Classifier and Regressor [8], Gaussian Naïve Bayes [9], Multinomial Naïve Bayes [9], Multi-layer Perceptron Classifier and Regressor [10], K-Nearest Neighbor [11], Logistic Regression [12], and both the Random Forest Classifier and Regressor [13].

C. *Evaluation*

For evaluating the classification models, we used precision, recall, f-score, and accuracy, and for evaluating the regression models we used mean error, mean squared error, and R2 value.

Since we used sklearns train_test_split [14] function to get the training and testing dataset, two sets of results were found. The first set (Table 1 and Table 2) were found from the training and testing dataset that gave the best results, and the second set (Table 3 and Table 4) were found from an average of 28 different testing and training datasets.

	Precision	Recall	F-Score	Accuracy
MLP	0.889	0.667	0.711	0.667
KNN	0.889	0.667	0.711	0.667
RF	0.889	0.667	0.711	0.667
GNB	0.667	0.333	0.444	0.333
MNB	0.611	0.500	0.548	0.500
SVM	0.111	0.333	0.167	0.333

Table 1: Single Classification Scores

	ME	MSE	R2
MLP	52.168	2721.536	-106.429
Log	7.394	54.667	-1.158
RF	7.782	61.209	-1.416
SVM	5.573	31.063	-0.226

Table 2: Single Regression Scores

	Precision	Recall	F-Score	Accuracy
MLP	0.468	0.440	0.417	0.440
KNN	0.505	0.423	0.424	0.422
RF	0.311	0.327	0.288	0.327
GNB	0.337	0.321	0.289	0.321
MNB	0.400	0.286	0.294	0.286
SVM	0.356	0.351	0.306	0.351

Table 3: Average Classification Scores

	ME	MSE	R2
MLP	40.864	1945.124	-22.433
Log	16.036	275.405	-2.002
RF	12.981	179.097	-0.748
SVM	12.709	172.216	-0.710

Table 4: Average Regressor Scores

IV. Results

Two different results were found. The results that got us the best results with a specific training and testing dataset (Table 1 and Table 2), and the average results that were got using many different combinations of training and testing datasets (Table 3 and Table 4).

The highest results we could get to accurately predict the AGS with classification was an accuracy of 0.667 and for regression was a mean error of 5.573 (From Tables 1 and 2). It is important to note that these values were from MLP, RF, and KNN classifier, and the SVM regressor.

Of course, some of the models work better with specific training and testing datasets. To test this and to get more variation in the results this was run multiple time using different training and testing dataset. It was found when doing this that the MLP performed the best in the classifiers and SVM performed the best in the regressors.

V. Discussion

It was found that MLP classifier had the highest accuracy out of the classifiers and that SVM had the lowest MSE out of the regressors.

Surprisingly MLP performed better than KNN. This surprised me because of how limiting the training data was. Usually, MLP works better with a lot more training data and KNN would be assumed to work better since there is less training data.

MNB performed the worst with an accuracy of 0.286 out of all the classifiers and MLP regressor performed the worst out of the regressors.

MLP performed 4.1% better than KNN in the classifiers and SVM performed 2.1% better than the second best regressor which was the random forest regressor.

To get different results more features could have been used. For example, not all features in the code were used when calculating these numbers and a lot of features were added into the data frame and removed from the data frame to try and find an optimal answer.

VI. Conclusion

We have produced results that show how different machine learning models compare to one another. For classifiers we found that the MLP classifier performs the best with a 4.1% gain on the second-best classifier, and that the SVM regressor performs 2.1% better than the second best regressor. This was done by using multimodal features of the group’s interactions with each other.

Using both the dataset features and linguistic features it was found that just using the linguistic features on their own produce better results. We also noticed that the highest accuracy and lowest mean error was found using only linguistic features and not the dataset features.

Acknowledgment

Joshua Fantillo is a student researcher at The University of the Fraser Valley.

REFERENCES

[1] Gabriel Murray, McKenzie Braley, The GAP Corpus, (2019), Retrieved Feb 1, 2022 from https://sites.google.com/view/gap-corpus/home

[2] Jonathan P. Chang, Caleb Chiam, Liye Fu, Ansdew Wang, Justine Zhang, Cristian Danescu-Niculescu-Mizil. 2020. "ConvoKit: A Toolkit for the Analysis of Conversations". Proceedings of SIGDIAL. Retrieved Feb 1, 2022 from https://convokit.cornell.edu/

[3] nltk.tokenize package, (Oct 19, 2021), Retrieved Feb 1, 2022 from https://www.nltk.org/api/nltk.tokenize.html

[4] nltk.stem.wordnet, (Oct 19, 2021), Retrieved Feb 1, 2022 from https://www.nltk.org/_modules/nltk/stem/wordnet.html

[5] scikit learn, (2007-2022), Retrieved October 26, 2021 from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html

[6] Categorizing and Tagging Words, (Sept, 4, 2019), Retrieved October 26, 2021 from https://www.nltk.org/book/ch05.html

[7] nltk.probability module, (Oct 19, 2021), Retrieved October 29, 2021 from https://www.nltk.org/api/nltk.probability.html

[8] scikitlearn Support Vector Machines, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/svm.html

[9] scikitlearn Naïve Bayes, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/naive_bayes.html

[10] scikitlearn Neural Network Models (supervised), (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[11] scikitlearn Nearest Neighbors, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/neighbors.html

[12] scikitlearn Logistic Regressoin, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[13] scikitlearn Esemble Methods, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/ensemble.html

[14] scikitlearn train_test_split, (2007-2022), Retrieved Feb 1, 2022 from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html