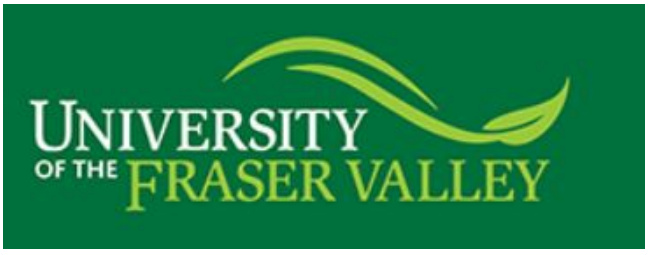
Capstone Project

Analyzing Conversational Data using ConvoKit

Pulkit Kalhan



Department Name : Computing Science
University Name : UFV
Instructor : Dr. Gabriel Murray
Date : April 12, 2022

Abstract

In recent years due to development in Natural Language Processing, machines can understand and communicate with us. In today's world variety of NLP applications such as Machine Translations, Text Summarization, Text Classification, Chat-Bots etc., have become extremely common. However, text conversations are yet to be explored. Conversational Data can be used for a variety of applications such as predicting conversation Politeness or predicting if a conversation can derail into personal attacks or analyzing speaker diversity and speaker coordination. In this project, an attempt has been made to analyze conversational data using ConvoKit, a framework built using Python specifically used for analyzing conversational data.

Acknowledgement

I wish to express my gratitude and appreciation to Prof. Gabriel Murray for his encouragement and guidance throughout the project. Thank You for providing me with such a golden opportunity to work under your guidance, advice, and supervision throughout the course.

1 Introduction

Cornell Conversational Analysis Toolkit (ConvoKit) is a toolkit that contains various tools to extract conversational data which can be used to analyze social interactions embedded within [5]. None of the existing frameworks which have been developed tackle the problems revolving around conversational data. Most of the works revolving around text data are generally in the field of Natural Language Processing (NLP). ConvoKit is a framework developed by researchers of Cornell University specifically to analyze conversational data. The framework is divided into two major ideas, Corpora and Transformations. Corpus contains a collection of conversations that has Speakers and Utterances. Conversations, Speakers, and Utterances have unique IDs which is a string of arbitrary length. The above three objects are interlinked and hence for a given utterance, a speaker or conversation can be extracted. The transformer object takes the Corpus class and performs high-level modification on the corpus. For example, the transformer class will take the text attribute in the Corpus to perform a task such as tokenization, dependency parsing, annotating data with politeness strategies, etc. For more information please check the correct documentation for ConvoKit.

The goal of the project was to extract features from multi-modal datasets and analyze the relationships of the extracted features using the framework provided by ConvoKit. ConvoKit has many datasets to choose from. The datasets I chose for my experimental analysis are Stanford Politeness Corpus (Wikipedia), Stanford Politeness Corpus (Stack Exchange), Wikipedia Talk Pages Corpus, and Group Affect and Performance GAP Corpus. However, the primary focus of my experiments were on Stanford Politeness Corpus (Wikipedia) and GAP Corpus. The other two Corpora (Stack Exchange) and Wikipedia Talk Pages were only used for testing and preliminary data analysis. The aim was to use extract features from Wikipedia Politeness Data set using the Transformers such as Politeness Strategies(or Linguistic Marker), Bag of Words Model (BoW) and TF-IDF (Term Frequency - Inverse Document Frequency) provide in the ConvoKit and analyze these models using Classifier, Vector-Classifier and Coordination Transformers provided in the ConvoKit Framework. The extracted features were also analyzed using other Machine Learning (ML) Algorithms such as Multinomial Naive Bayes, K-Nearest Neighbour Classifier, Simple Perceptron, and other ML Classifiers. Unsupervised ML Algorithms such as K-Means Clustering , C-Means Clustering (Fuzzy Clustering), Principal Component Analysis (PCA) were also used on some datasets.

2 Politeness Corpus

2.1 Stanford Politeness Corpus - Wikipedia and Stack-Exchange

The Stanford Politeness Corpus(Wikipedia) is derived from Wikipedia Talk page Corpus and has 4,353 utterances in total. The utterance in the Corpus is annotated with politeness labels where -1 corresponds to impolite, 0 corresponds to neutral, and 1 corresponds to polite. The Stack Exchange Corpus is similar to the Wikipedia corpus in structure with an exception of the number of utterances. The other common features at the Utterance level which both these corpora have are text, speaker, reply to, annotations, and Normalized Scores (based on Annotations).

The Politeness Strategies is a "computational framework for identifying linguistic aspects of politeness"[2] which is broadly based on linguistic theories provided by Brown and Levinson (1987)[9]. The Politeness Strategies have 21 core politeness markers such as Gratitude, Positive lexicon (HASPOSITIVE), Please, Negative lexicon, Indicative, etc. The 21 core politeness markers were used as features/predictors to train a simple model. For more information regarding politeness strategies refer [2]. The accuracy, precision, and F1 score obtained from these features were used as a baseline to compare. For comparing models such as BoW scores from [2] were also used for baseline.

2.2 Methodology

- ConvoKit's inbuilt *TextParser* Transformer was used to perform parsing and tokenization. By default, the *TextParser* returns a dictionary that contains text, Parts of Speech(POS) tag, and dependency between the token and the parent.

- ConvoKit's inbuilt Politeness Strategies Transformer was used to extract politeness markers as predictors and the Binary variable was used as a response variable. Supervised Learning Classification Algorithms such as Logistic Regression, Naive Bayes, Support Vector Machine (SVM), K-Nearest Neighbour with k nearest value set to 3, Decision Tree, Multi-Layer Preceptron(MLP), and Simple Perceptron were used. The models used can be found in Python's Sklearn package. ConvoKit provides the Classifier Transformer which can be used to perform the same task. By default, the Classifier and the Vector-Classifier Transformer also use Standardized Logistic Regression provided by sklearn library. In fact as these transformers also use Scikit learn, throughout the experiments a combination of the transformers and sklearn models were used.

- ConvoKit's inbuilt Politeness Strategies Transformer was used to extract politeness markers as predictors and the Binary variable was used as a response variable. For the scope of our experiments records with outcome

variable of -1 and 1 were considered. The records which had neutral label i.e. 0 was filtered out from both the datasets. Supervised Learning Classification Algorithms mentioned above were then used to build classifiers on these features.

- To evaluate model performance K-Fold Cross Validation (CV) was used. K-fold CV has computational advantage and low bias - variance test error rates as compared to other CV methods such as Leave One Out Cross Validation (LOOCV).

- To further evaluate the model's Accuracy, Precision and F1 Score were used. ROC-AUC curves were also used to study the classifier output quality.

- ConvoKit's framework provides various Transformers such as BoW and TF-IDF for feature extraction. BoW and TF-IDF models were used to convert our text data to numeric values. The numeric values were used to train our above models. Metrics such as accuracy, precision, and F1 scores were calculated for all models which were used for performance evaluation. Various classification models were built using BoW, BoW and Linguistic Markers together, and TF-IDF as features with their corresponding politeness labels. Various supervised learning ML classifiers were used for label prediction and various metrics such as accuracy, F1 Scores, etc were calculated for various models. The metrics calculated were also used for comparing a classifier's performance across different datasets.

2.3 Experimental Results

2.3.1 In domain Analysis: Politeness Markers (Wikipedia and Stack Exchange)

As mentioned above all the records with neutral responses were dropped. Observations/Records with a Binary outcome of -1 and 1 were retained. After filtering Wikipedia Politeness Corpus, the resultant dataset contained only 2178 observations which were used to build the training models. Before training the models with the observations, K Fold CV was performed to obtain the various scores. Cross Validation provides information about how well a classifier will perform on our test set. As the dataset was well balanced with 1089 observations for each of the two binary classes, K Fold CV gave us ideal results which are summarized in the table below. The value of $k = 5$ was chosen as it produces low test error rates which neither have high variance nor high bias.

Model	Accuracy Score	Precision Score
Standardized Logistic Regression	76.12%	-
Logistic Regression	76.35%	78.18 %
Multinomial Naive Bayes	74.47 %	73.80%
SVM (Kernel= RBF)	76.17%	79.26%
SVM (Kernel = Polynomial)	75.43%	80.48%
K-Nearest Neighbour	70.01%	72.96%
Decision Tree (criterion = Entropy)	71.94%	74.74%
MLP(Relu)	73.87%	74.87%
MLP (Hyberbolic Tangent)	74.28%	76.24%
Perceptron	72.95%	75.45%

Table 1: Table showing Mean K-Fold Cross Validation Scores of Various Models

The Standardized Logistic Regression and the non-Standardized Logistic Regression produced similar results as expected. The accuracy and precision for Logistic Regression and Multinomial Naive Bayes Classifier were one the highest, followed closely by SVM Classifier with Kernel parameter set to RBF and polynomial respectively. The other classifiers such as Decision Tree, MLP, and Perceptron performed averagely and had decent CV scores with KNN Classifier being the lowest..

For training the above Models, 21 politeness markers (or linguistic markers) of a corresponding utterance were used as predictors and their corresponding Binary outcome was considered as a response variable. All the 10 models were trained on 90% of the utterances (1960 out of 2178) which formed our training set. The remaining 10% of the data set (218 out of 2178) was used for testing purposes. The accuracy, precision, recall, and f1 scores obtained for all the above models were treated as baselines for other experiments. The class labels of the filtered dataset were re-mapped to 0 and 1 where 0 corresponds to an utterance being Impolite and 1 corresponds to a Polite utterance. The following table shows the scores of all the models

Model Name	Class Label	Precision	Recall	F1 Score	Accuracy
Standardized Logistic	Impolite	72.0%	78.0%	75.0%	75.0%
	Polite	78.0%	71.0%	74.0%	
Logistic Regression	Impolite	71.1%	76.4%	73.6%	73.4%
	Polite	76.0%	70.5%	73.1%	
Multinomial NB	Impolite	73.8%	71.7%	72.7%	73.9%
	Polite	73.9%	75.9%	74.9%	
SVC RBF	Impolite	71.1%	81.1%	75.8%	74.8%
	Polite	79.4%	68.8%	73.7%	
SVC Poly	Impolite	70.4%	83.0%	76.2%	74.8%
	Polite	80.6%	67.0%	73.2%	
KNN	Impolite	63.5%	75.5%	69.0%	67.0%
	Polite	71.7%	58.9%	64.7%	
Decision Tree	Impolite	69.3%	74.5%	71.8%	71.6%
	Polite	74.0%	68.8%	71.3%	
MLP (Relu)	Impolite	75.8%	70.8%	73.2%	74.8%
	Polite	73.9%	78.6%	76.2%	
MLP (tanh)	Impolite	75.5%	72.6%	74.0%	75.2%
	Polite	75.0%	77.7%	76.3%	
Perceptron	Impolite	90.6%	27.4%	42.0%	63.3%
	Polite	58.6%	97.3%	73.2%	

Table 2 : Table showing Important Metrics for Various Models (Stanford Politeness - Wikipedia)

As the data set was balanced the precision and recall scores seemed to be quite balanced. Most of the models didn't show any abnormalities such as high recall and low precision or low recall and high precision. The accuracy scores of most of the models were above 70% with exception of KNN and Perceptron which had lower accuracy scores of about 63.3% and 67%. The Simple Perceptron had a high precision and lower recall for the Impolite label (90.6% and 27.4%) and low precision and high recall for the polite label(58.6% and 97.3%). A model which predicts low precision and high recall means results that were predicted by the model i.e. labels predicted were incorrect when compared to the training model. A model which has high precision but low recall means our model was too rigid when it came to predicting labels correctly. The model was so rigid that it misclassified our true positive when compared to training labels. The KNN model also showed similar behavior but performed below average. However, as compared to the Simple Perceptron, KNN performed much better with a better combination of recall and precision scores. The F1 Scores which is the overall measurement of a model's accuracy as it combines precision and recall into a single value. Most of our models had an F1 score which was above 70% with exception of KNN and Perceptron, with Perceptron performing out of the two.

As the Stack Exchange Corpus was also used for testing purposes, simple models containing 21 politeness markers as features were also used. The models

for Stack Exchange were built in the same way as that of Stanford Wikipedia Politeness Corpus. Observations with non-zero Binary outcomes were used. The baseline models for Stanford Corpus built were trained on 90% of the observations in the dataset. The remaining 10% of the observations were used for testing our models. The table below shows the scores of all the models. The Standardized Logistic Regression Model was not included as it gave similar results when compared to Logistic Regression.

Model Name	Class Label	Precision	Recall	F1 Score	Accuracy
Logistic Regression	Impolite	57.1%	56.0%	56.5%	58.6%
	Polite	60.0%	61.0%	60.5%	
Multinomial NB	Impolite	55.8%	57.2%	56.5%	57.7%
	Polite	59.5%	58.1%	58.8%	
SVC RBF	Impolite	58.2%	51.6%	54.7%	58.9%
	Polite	59.5%	65.7%	62.4%	
SVC Poly	Impolite	54.7%	54.7%	54.7%	56.5%
	Polite	58.1%	58.1%	58.1%	
KNN	Impolite	57.1%	61.0%	59.0%	59.2%
	Polite	61.5%	57.6%	59.5%	
Decision Tree	Impolite	52.6%	57.9%	55.1%	54.7%
	Polite	57.1%	51.7%	54.3%	
MLP (Relu)	Impolite	56.5%	59.7%	58.1%	58.6%
	Polite	60.7%	57.6%	59.1%	
MLP (tanh)	Impolite	56.0%	56.0%	56.0%	57.7%
	Polite	59.3%	59.3%	59.3%	
Perceptron	Impolite	58.3%	4.4%	8.2%	52.6%
	Polite	52.4%	97.1%	68.0%	

Table 3 : Table showing Important Metrics for Various Models (Stanford Politeness -Stack Exchange)

Most of the models shown above performed below average with all of them having an accuracy and F1 score less than or equal to 60%. As usual Simple Perceptron performed extremely poorly with extreme values of precision and recall. MLP being a feedforward artificial neural network performs much better than a Simple Perceptron as it had various hidden layers and errors can be backpropagated to yield a better predictive model. In an attempt to increase the baseline accuracy and F1 scores the training size of the data set was changed to 95%. Changing the training size drastically improved the scores of our baseline models with Logistic Regression and SVC both had an accuracy of roughly 70%. However, changing the training size to a higher ratio comes with the risk of overfitting our model (model learning test data too well). As our utterance dataset only has about 6600 utterances which is small for text data using such a high ratio isn't recommended and hence was not used as a baseline

The scores obtained from Wikipedia and Stack Politeness Corpus were used as a baseline for the models built using BoW(Bag of Words), TF-IDF, and for cross-domain analysis. All these models were used using built-in ConvoKit Transformers. The transformer gave us corresponding vectors which were used as features of these machine learning/Deep Learning models. The following section summarizes the methodology and results:

2.3.2 Cross-domain Analysis: Politeness Markers (Wikipedia and Stack Exchange)

Linguistic Markers Model: The models built on 21 politeness markers as features and Binary variable (polite or impolite) as predictors using Stanford Wikipedia was used for cross-domain analysis. Stanford Stack Exchange Dataset was treated as a testing set. The 21 politeness markers as features extracted earlier were used to predict an outcome. As true labels for Stack Exchange were already known, precision-recall, F1, and accuracy scores were calculated and analyzed. Compared to our baseline models built on Wikipedia Politeness Corpus (Ref. Table 2), where most of our models performed with an accuracy and F1 scores of around 70% or more, our new model failed to outperform our baselines. Only Multinomial NB had an accuracy of 60% which somewhat came a little close to its baseline accuracy of 73.9%. However, in comparison to baseline accuracy and F1 score generated by models of Stack Exchange Corpus (Ref. Table 3) our model performed extremely well. The model had accuracy and F1 close to the baseline for most of the models and in some cases even outperformed the baseline models. The following table shows only the accuracy score obtained :

Model(Wiki)	Accuracy Score
Logistic Regression	56.9%
Multinomial Naive Bayes	59.7%
SVM (Kernel= RBF)	57.5%
SVM (Kernel = Polynomial)	57.5%
K-Nearest Neighbour	56.0%
Decision Tree (criterion = Entropy)	56.7%
MLP(Relu)	57.5%
MLP (Hyberbolic Tangent)	56.8%
Perceptron	55.1%

Table 4: Cross-Domain Scores (Test Set : Stanford Stack Exchange)

The models weren't able to predict most of the labels correctly. The overall performance of the classifier was also evaluated using ROC (Reciever Operating Characteristics) curve. An ideal ROC curve should drastically rise to the top left corner. The area under the ROC curve (AUC) gave us an insight into the classifier's performance. For example, by looking at the ROC -AUC graph and AUC score below, we can conclude that our Naive Bayes classifier

performed below average when classifying positive classes in a binary dataset.

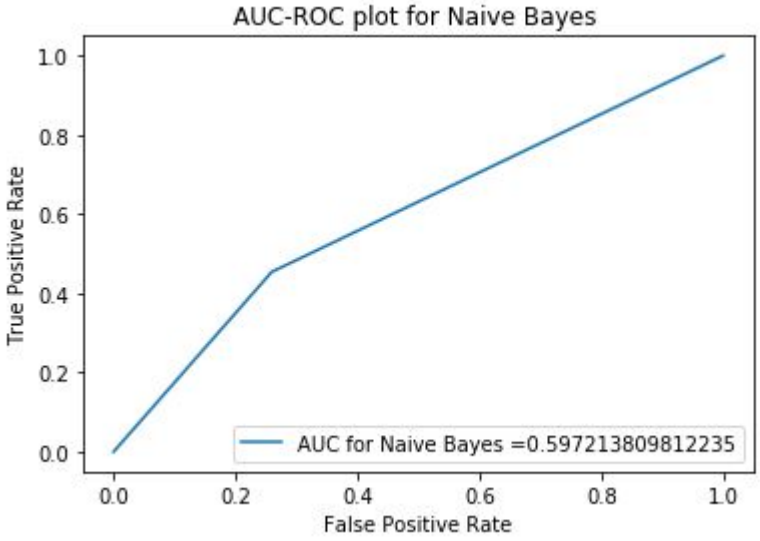


Figure 1 : AUC-ROC plot for MB Naive Bayes

2.3.3 In domain Analysis : BoW (Wikipedia & Stack Exchange)

BoW models were built using ConoKit's inbuilt transformer. BoW model is one of the simplest way to represent text data in a machine-readable format. Any dependencies or the structure is disregarded and only the multiplicity (term frequency) of the words are stored in a matrix format. The BoW Transformer by default removed all the stop words used in an utterance. Wikipedia Politeness dataset had a total of 563 unique words in its vocabulary while Stack Exchange Politeness dataset had 717 unique words in its vocabulary. The unique words formed the feature set for all the models and the Binary variable formed the outcome set. The models were trained on 90% of the total observations while the remaining 10% of observations were used for testing. The BoW models for Stanford Politeness Corpus (Wikipedia) performed extremely well when compared to baseline models in Table 2. The BoW models outperformed each of its baseline models with exception of KNN and SVC(with polynomial kernel). There was a decrease in F1 and accuracy scores for both models. However, the performance of the Perceptron model when compared to baseline (Table 2) increased drastically. From a baseline accuracy of 52.6% and F1 score of 8.2% (Impolite class) and 68.0%, BoW Perceptron model had an accuracy of 80.3% and F1 scores of 78.4% (Impolite) and 81.9%. Similarly BoW model for Stack Exchange Corpus outperforms its baseline model for Logistic Regression, Naive Bayes, SVC(RBF), Decision Tree, MLP(tanh). The other models however showed a decrease in performance and had lower F1 and accuracy scores

when compared to the baseline(Table 3). The following table summarizes the results obtained:

Model Name	Class Label	Precision	Recall	F1 Score	Accuracy
Logistic Regression	Impolite	79.1%	82.1%	80.6%	80.70%
	Polite	82.4%	79.5%	80.9%	
Multinomial NB	Impolite	79.2%	79.2%	79.2%	79.8%
	Polite	80.4%	80.4%	80.4%	
SVC RBF	Impolite	78.1%	84.0%	80.9%	80.7%
	Polite	83.7%	77.7%	80.6%	
SVC Poly	Impolite	60.8%	90.6%	72.7%	67.0%
	Polite	83.3%	44.6%	58.1%	
KNN	Impolite	51.1%	87.7%	64.6%	53.2%
	Polite	63.9%	20.5%	31.1%	
Decision Tree	Impolite	71.7%	71.7%	71.7%	72.5%
	Polite	73.2%	73.2%	73.2%	
MLP (Relu)	Impolite	82.5%	80.2%	81.3%	82.1%
	Polite	81.7%	83.9%	82.8%	
MLP (tanh)	Impolite	78.0%	80.2%	79.1%	79.4%
	Polite	80.7%	78.6%	79.6%	
Perceptron	Impolite	83.9%	73.6%	78.4%	80.3%
	Polite	77.6%	86.6%	81.9%	

Table 5a : Summary of Metrics for BoW models (Wikipedia)

Model Name	Class Label	Precision	Recall	F1 Score	Accuracy
Logistic Regression	Impolite	58.1%	61.0%	59.5%	60.1%
	Polite	62.2%	59.3%	60.7%	
Multinomial NB	Impolite	59.0%	61.6%	60.3%	61.0%
	Polite	63.0%	60.5%	61.7%	
SVC RBF	Impolite	59.5%	61.0%	60.2%	61.3%
	Polite	63.1%	61.6%	62.4%	
SVC Poly	Impolite	51.5%	88.1%	65.0%	54.4%
	Polite	67.8%	23.3%	34.6%	
KNN	Impolite	50.6%	82.4%	62.7%	52.9%
	Polite	61.1%	25.6%	36.1%	
Decision Tree	Impolite	51.4%	56.6%	53.9%	53.5%
	Polite	55.8%	50.6%	53.0%	
MLP (Relu)	Impolite	54.2%	56.6%	55.4%	56.2%
	Polite	58.2%	55.8%	57.0%	
MLP (tanh)	Impolite	54.0%	55.3%	54.7%	55.9%
	Polite	57.7%	56.4%	57.1%	
Perceptron	Impolite	53.3%	50.9%	52.1%	55.0%
	Polite	56.4%	58.7%	57.5%	

Table 5b : Summary of Metrics for BoW models (Stack Exchange)

2.3.4 In domain Analysis: BoW with Politeness Markers (Wikipedia & Stack Exchange)

To further evaluate our model, the unigram features obtained from BoW models were combined with the 21 linguistic Markers obtained earlier in the experiment. These new features were used to perform classification tasks using Supervised ML algorithms. For Wikipedia Politeness, the resultant models on an average had an increased performance of 2-3% when compared to baselines which were built using only Linguistic markers or BoW as features. For Stack Exhange, the new model performed very similar to baseline (Table 3) and BoW models(Table 5b.). The new models built using both BoW(unigram features) and Linguistic Markers together, failed to outperform our baselines. The values of accuracy and F1 scores obtained were similar to those in Table 3 and Table 5b. with no significant changes. The following table provides a summary of models with only their accuracy scores :

Model Name	Corpus	Accuracy
Logistic Regression	Wiki	82.1%
	SE	59.5%
Multinomial NB	Wiki	80.7%
	SE	60.4%
SVC RBF	Wiki	82.6%
	SE	65.0%
SVC Poly	Wiki	72.9%
	SE	56.5%
KNN	Wiki	63.3%
	SE	51.7%
Decision Tree	Wiki	73.9%
	SE	56.8%
MLP (Relu)	Wiki	81.2%
	SE	59.2%
MLP (tanh)	Wiki	77.5%
	SE	54.7%
Perceptron	Wiki	79.4%
	SE	57.7%

Table 6 : Summary of Metrics for BoW Models + Linguistic Markers (Stack Exchange and Wikipedia)

The following figure shows the AUC-ROC plot for one of the models. The AUC value of the model was around 82.1%

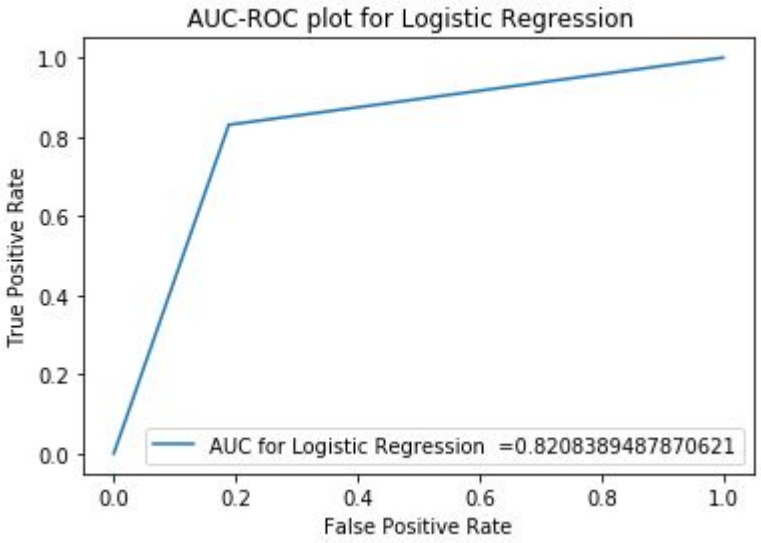


Figure 2 : AUC-ROC plot for Unigram features with Linguistic Markers (Logistic Regression - Wikipedia)

2.3.5 In domain Analysis: TF-IDF (Term Frequency - Inverse Document Frequency) (Wikipedia & Stack Exchange)

TF-IDF was implemented using ConvoKit's Column normalized TF-IDF Transformer. The transformer by default normalizes the words by their corresponding columns. The model was built without removing any stop words. Removing stop words can sometimes remove salient features that might be extremely essential. The unigram models were built on utterances for both corpora using TF-IDF transformers. The TF-IDF feature values were used as train and test data. For classification models built using TF-IDF (Wikipedia), no significant improvement in F1 scores or accuracy scores were seen in comparison to baseline models (Linguistic Marker and BoW). Except for Logistic Regression, all our classification models failed to outperform their baseline accuracy and F1 scores. Also, the MLP classifier (with hyperbolic tangent as activation function) and Perceptron classifier, outperformed their corresponding BoW baseline accuracy scores. The following table summarizes the results :

Model Name	Class Label	F1 Score	Accuracy	Baseline Accuracy
Logistic Regression	Impolite	82.1%	81.7%	Ling. 73.4%
	Polite	81.1%		BoW 80.70%
Multinomial NB	Impolite	68.5%	73.4%	Ling. 73.9%
	Polite	77.0%		BoW 79.8%
SVC RBF	Impolite	70.9%	72.9%	Ling. 74.8%
	Polite	74.7%		BoW 80.7%
SVC Poly	Impolite	0.0%	51.4%	Ling. 74.8%
	Polite	67.9%		BoW 67.0%
KNN	Impolite	7.3%	53.2%	Ling. 67.0%
	Polite	68.7%		BoW 53.2%
Decision Tree	Impolite	70.0%	69.3%	Ling. 71.6%
	Polite	68.5%		BoW 72.5%
MLP (Relu)	Impolite	80.0%	79.4%	Ling. 74.8%
	Polite	78.7%		BoW 82.1%
MLP (tanh)	Impolite	80.4%	79.8%	Ling. 75.2%
	Polite	79.2%		BoW 79.4%
Perceptron	Impolite	72.0%	76.1%	Ling. 63.3%
	Polite	79.2%		BoW 80.3%

Table 7a : Summary of Metrics for TF-IDF models (Wikipedia) with Baseline Scores (From Table 2 and 5a)

Model Name	Class Label	F1 Score	Accuracy	Baseline Accuracy
Logistic Regression	Impolite	58.6%	55.6%	Ling. 58.6%
	Polite	52.1%		BoW 60.1%
Multinomial NB	Impolite	53.3%	52.3%	Ling. 57.7%
	Polite	51.2%		BoW 61.0%
SVC RBF	Impolite	53.8%	51.7%	Ling. 58.9%
	Polite	49.4%		BoW 61.3%
SVC Poly	Impolite	64.9%	48.0%	Ling. 56.5%
	Polite	0.0%		BoW 54.4%
KNN	Impolite	64.9%	48.0%	Ling. 59.2%
	Polite	0.0%		BoW 52.9%
Decision Tree	Impolite	52.1%	52.3%	Ling. 54.7%
	Polite	52.4%		BoW 53.5%
MLP (Relu)	Impolite	53.7%	52.6%	Ling. 58.6%
	Polite	51.4%		BoW 56.2%
MLP (tanh)	Impolite	55.7%	53.5%	Ling. 57.7%
	Polite	51.0%		BoW 55.9%
Perceptron	Impolite	47.3%	56.2%	Ling. 52.6%
	Polite	62.5%		BoW 55.0%

Table 7b : Summary of Metrics(Stack Exchange) Baseline (Table 3 & 5b)

3 Group Affect and Performance (GAP) Corpus

3.1 Introduction

The dataset for GAP Corpus was collected at University of The Fraser Valley(UFV). GAP corpus contains data from 28 small group meetings. The GAP Corpus dataset has various features such as decision-making performance, Group Member performance, Individual Performance, Group Member Influence, etc which can help us to analyze group performance, individual interactions, and other computational metrics pertaining to small group tasks. The dataset is based on WST (Winter Survival Task) exercise where all the participants rank the 15 items according to given a hypothetical plane crash scenario [11].

The dataset has Speaker, Utterance, and Conversation Level Information which have various features associated with them. Speaker Level Information has a total of 84 observations with features such as Time Expectations, Worked Well Together, Year at UFV, Gender, etc. Utterance level information has a total of 8009 observations with features such as utterance text, utterance duration, corresponding speaker, etc. Conversation Level Information has a total of 28 observations with features such as group number, meeting length, Average Group Scores(AGS), Group Time Expectation(TE), and Group Effectiveness. For additional details regarding the dataset refer [7,11].

3.2 Methodology

As the GAP corpus is a small corpus with a small number of participants, techniques such as EDA(Exploratory Data Analysis), Data Visualization (using matplotlib and seaborn), and Unsupervised Learning ML algorithms (Clustering and PCA) were used. For EDA, various plots such as Count plots, Category plots, Box plots, Relational Plots, Histograms, and Pie charts were used. Clustering and PCA were performed on all Utterance, Speaker, and Conversation Level Information. Clustering Algorithms such as K-Means Clustering (Hard Clustering), C-Means Clustering(Overlapping Clustering), Hierarchical Clustering, and PCA were performed also performed on Group Level and Speaker Level data.

For Clustering Group Level Information, 3 types of clustering techniques were used. K-Means or hard clustering, C-Means Clustering for Fuzzy Clustering, and Hierarchical Clustering for Agglomerative Clustering. To evaluate group-level clustering models, only 8 simple features were used. The features for clustering were Meeting length, Absolute Group Score(AGS), Group Time Expectations(TE), Group Worked Well Together(WW), Group Worked Time Management(TM), Group Eff, Group Overall Quality of Work(QW) and Group Satisfaction(Sat). To reduce the curse of dimensionality Principal Component Analysis was used on the dataset. The components obtained after performing PCA were used as ML features for all the clustering methods mentioned above.

For Clustering Speaker Level Information instead of using a naive approach of using features such as AIS, AII, Ind TE, Ind WW, Ind TM, etc. An

average TF-IDF was calculated using utterance level information for each of the participants. Cosine similarity between each of the participants was calculated. The computed pairwise cosine similarities were used as features for various clustering models

3.3 Experiments

3.3.1 Descriptive Statistics

The figures below summarize basic descriptive statistics such as mean, median, standard deviation, min, max, etc, and Correlation values between values for Participants (Speaker level information) and Groups (Conversation level information) that were obtained. The heatmaps in the figure below show correlation values between variables. For instance, in figure 5 AIS (Absolute Individual Score) and AII (Absolute Individual Influence) had a negative correlation value of -0.15 while Individual Work Well Together (Ind WW) and Individual Overall Satisfaction(Ind Sat) had a strong positive correlation value of 0.68.

	meta.AIS	meta.AII	meta.Ind_TE	meta.Ind_WW	meta.Ind_TM	meta.Ind_Eff	meta.Ind_QW	meta.Ind_Sat	meta.Ind_Lead
count	84.000000	84.000000	84.000000	84.000000	84.000000	84.000000	84.000000	84.000000	84.000000
mean	76.638571	43.867500	3.059524	4.398810	4.422619	4.351190	4.315476	4.109524	3.452381
std	11.194425	16.416838	1.434104	0.684509	0.778065	0.981085	0.751624	0.591021	0.974429
min	48.000000	12.000000	1.000000	1.000000	2.000000	1.000000	1.000000	1.400000	1.000000
25%	68.000000	32.000000	2.000000	4.000000	4.000000	4.000000	4.000000	3.800000	3.000000
50%	78.000000	41.500000	3.000000	4.000000	5.000000	5.000000	4.000000	4.200000	3.000000
75%	84.000000	56.000000	4.000000	5.000000	5.000000	5.000000	5.000000	4.600000	4.000000
max	102.000000	78.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000

Figure 3 : Descriptive Statistics for Individual-level Participants (Speaker)

	meta.Meeting Length in Minutes	meta.AGS	meta.Group_TE	meta.Group_WW	meta.Group_TM	meta.Group_Eff	meta.Group_QW	meta.Group_Sat
count	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000
mean	9.182500	74.607143	3.089643	4.393214	4.423214	4.330714	4.311786	4.108929
std	3.474581	12.679348	1.199273	0.468196	0.477315	0.823020	0.473068	0.456641
min	3.880000	47.000000	1.500000	2.830000	2.800000	1.500000	2.500000	2.930000
25%	6.507500	66.000000	2.000000	4.000000	4.287500	4.187500	4.000000	3.875000
50%	8.825000	76.000000	2.835000	4.500000	4.500000	4.550000	4.330000	4.060000
75%	12.390000	83.250000	4.270000	4.670000	4.690000	5.000000	4.542500	4.447500
max	15.580000	100.000000	5.000000	5.000000	5.000000	5.000000	5.000000	4.870000

Figure 4 : Descriptive Statistics for Group-Level Participants (Conversation)

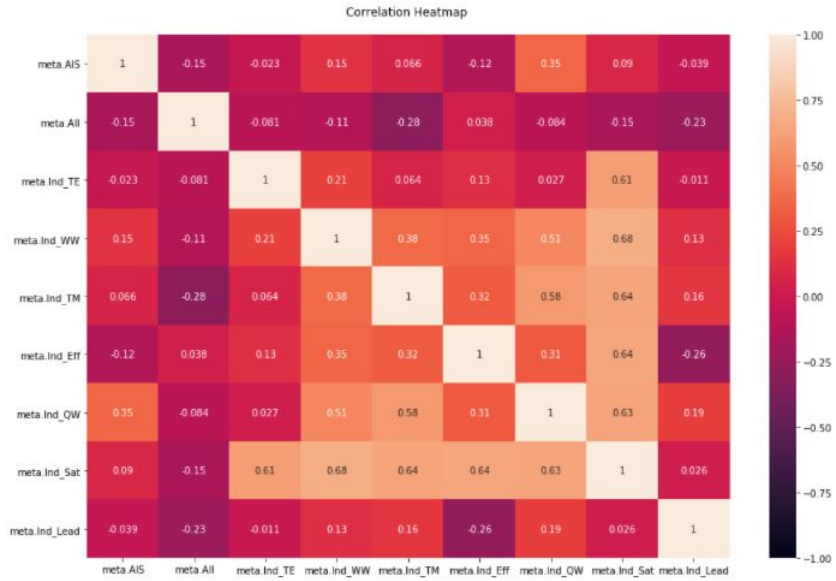


Figure 5 : Correlation heatmap for Speaker-level data

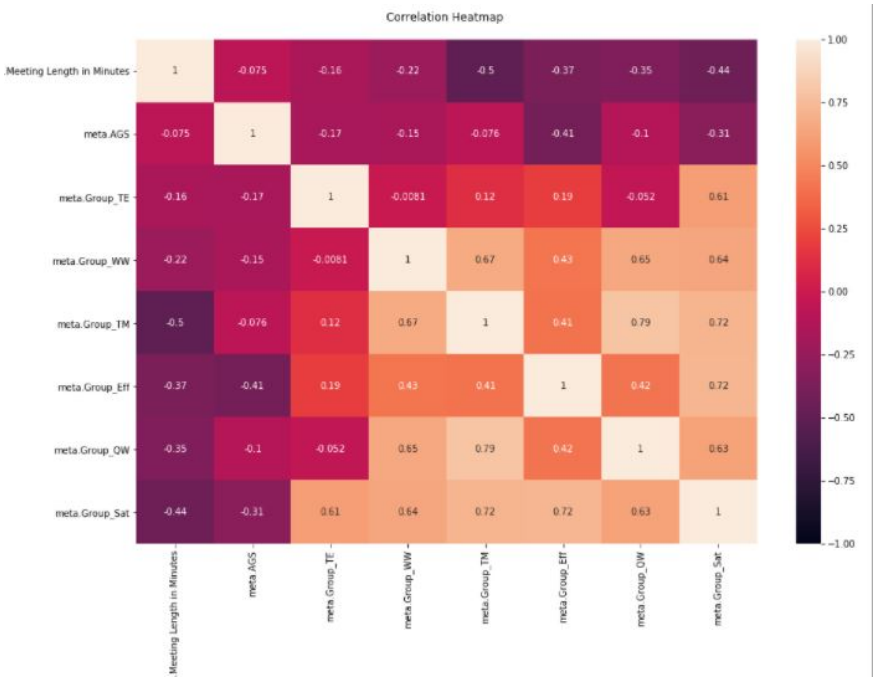
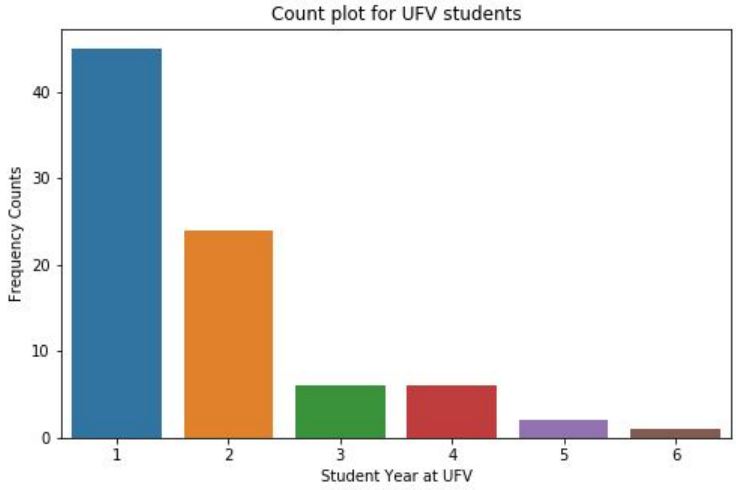


Figure 6 : Correlation heatmap for Group Level data.

.

3.3.2 EDA (Exploratory Data Analysis)

- Count Plots
  To visualize the speaker or participant diversity and visualize other observations present in a particular categorical variable various count plots were used. Some count plots are given below:




```
1    45
2    24
4     6
3     6
5     2
6     1
Name: meta.Year at UFV, dtype: int64
```

Figure 7 : Count plot showing Participant's Year at UFV

The above Figure 7 shows participant diversity with respect to their corresponding year at UFV. The counts show that majority of the students or participants of the task were students from 1st year and 2nd second. Only 15 participants had spent more than 2 years at UFV.

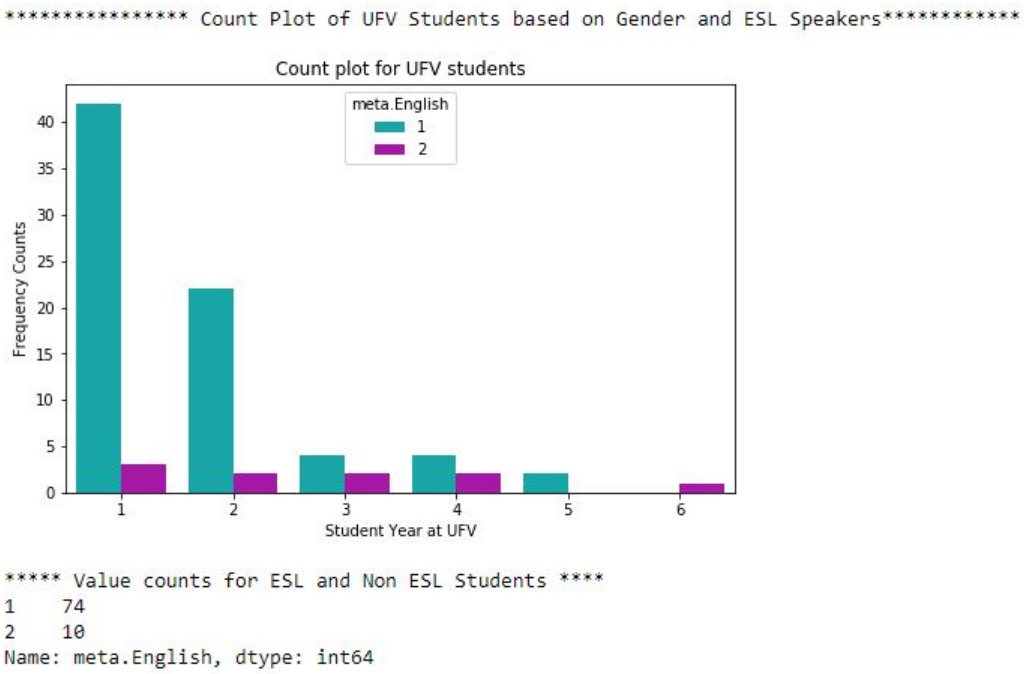


Figure 8 : Count plot showing Participant's Year at UFV based on Native-Non Native Speakers

Similarly, to analyze participant diversity Frequency plot based on ESL and Non-ESL speakers was graphed. The above Figure (fig 8) shows us the result obtained. The majority of the participants/speakers that were a part of the task were Native speakers (74 in total). Only 10 Non-ESL Speakers formed a part of the Winter Survival Task(WST) activity.

Count plots for other variables such as Individual Effectiveness, Individual Quality of work. Most of the values for this data fell in a range of 4-5 (about 88% of total participants).

Apart from participant level information, conversation data which included Group Score, Meeting times, Meeting size, etc were also analyzed using countplot. The following countplot (Figure 9) shows that approximately 16 groups had 3 participants while the remaining 12 groups had 2 or 3 participants.

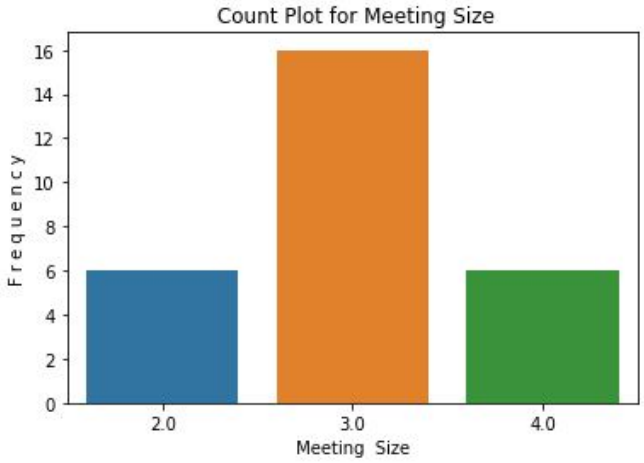


Figure 9 : Count plot showing Group Size

- Categorical Plots : As most of the data comprised of a mix of categorical and numeric variables, categorical plots were used to observe data distribution. Categorical Plots such as swarm plots were used to represent the distribution of data with non overlapping points. For example, a categorical plot was used to visualize the distribution of Absolute Individual Influence(AII) based on a participant's year at UFV. AII a numeric variable was plotted on the y-axis while the participant's year, a categorical variable was plotted on the x-axis.

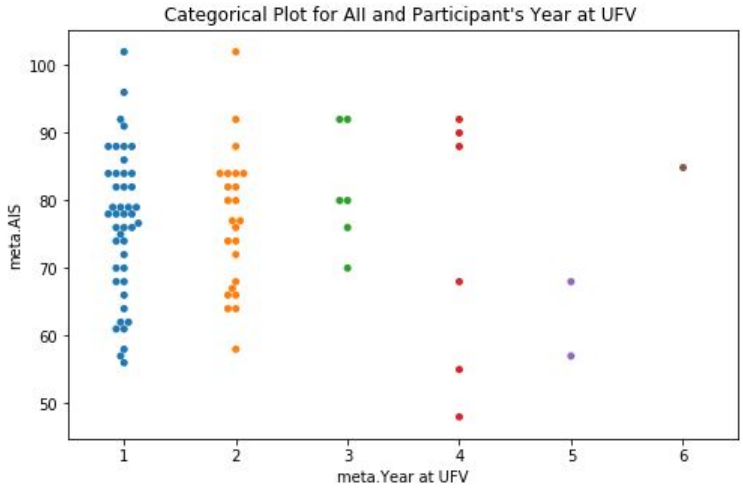


Figure 10a. : Swarm plot for AII and Participant's Year

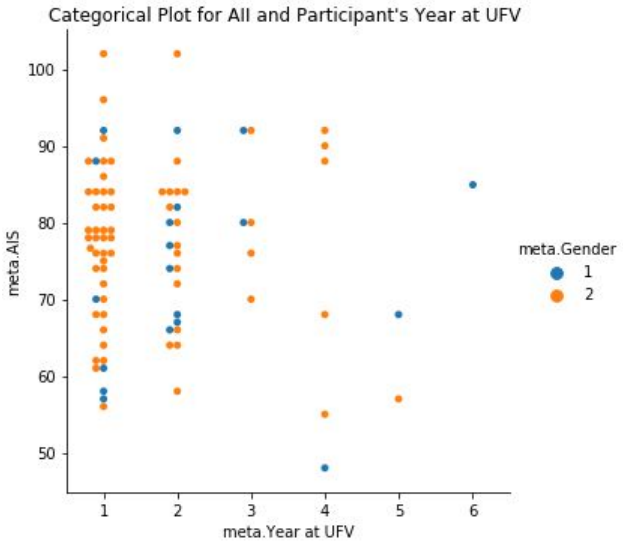


Figure 10b. : Swarm plot for AII and Participant's Year based on Gender

Similarly, Box plots were used to visualize Absolute Individual Score(AGS) and participant's year at UFV. Boxplots can be extremely useful for checking outliers or skewness in data. As GAP corpus is a small data, the chances of finding an outlier were small. The following figures show the results obtained:

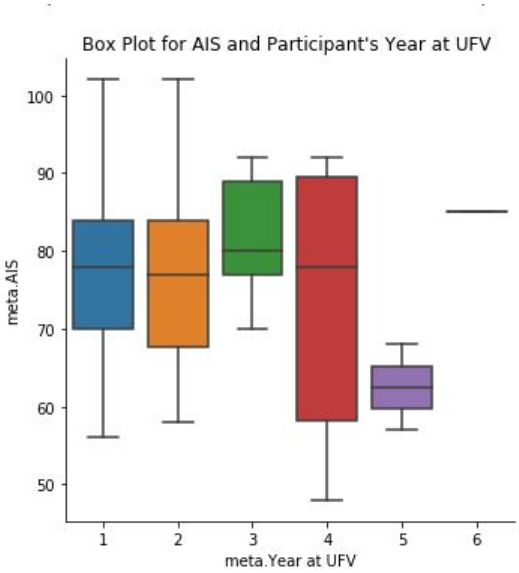


Figure 11a. : Box and Whisker plot for AIS and Participant's Year

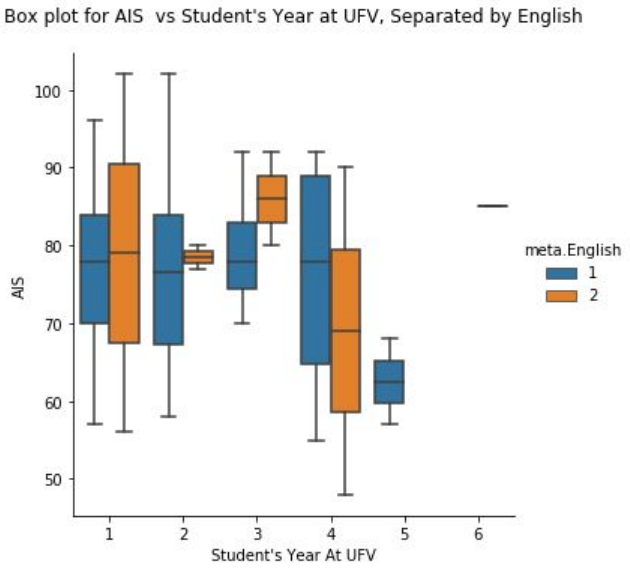


Figure 11b. : Box and Whisker plot for AIS and Participant's Year Grouped by ESL and NON-ESL participants (1 corresponds to NON-ESL and 2 corresponds to ESL)

- Joint Plots and Scatter plots: To analyze the distribution and observe an underlying relation between given variables scatter chart and relation plot was used. Scatter and Joint plots are shown below (Fig. 12) showed no relation between any of its pairs.

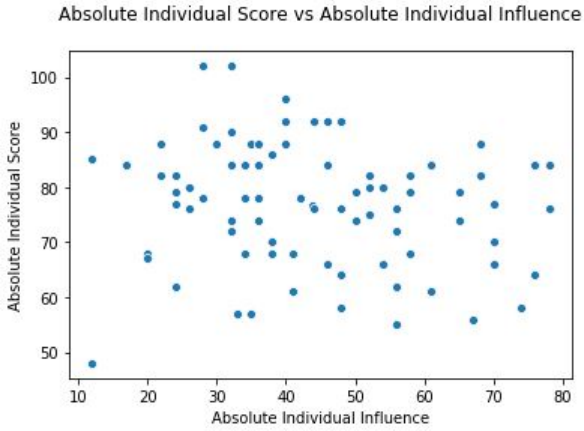


Figure 12a. : Scatter Plot for AII VS AIS

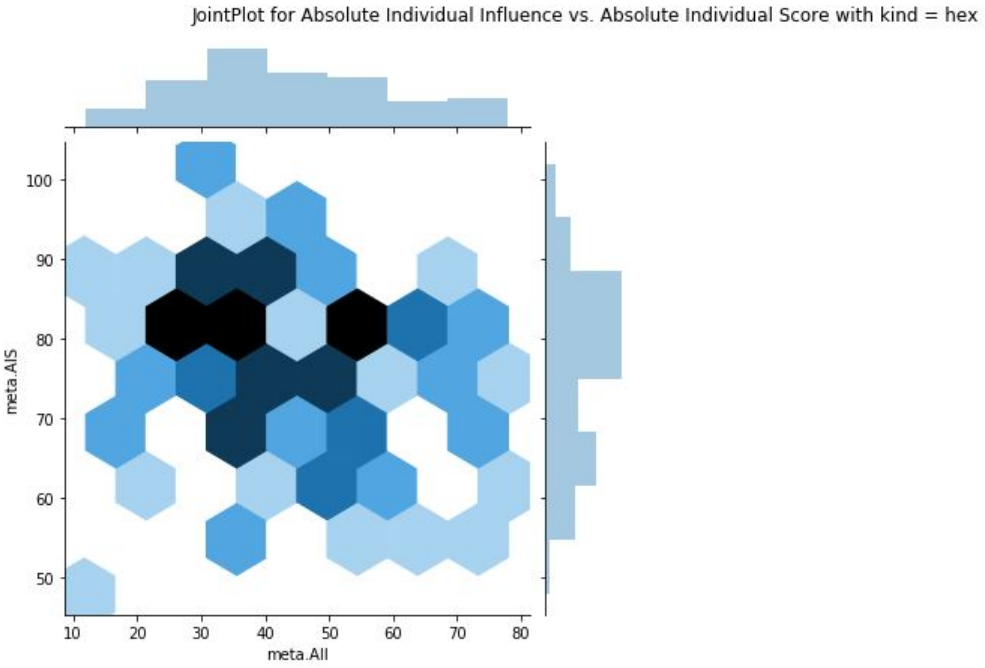


Figure 12b. : Joint Plot for AII VS AIS

- Histogram and Distribution Plot : To visualize AII's and AIG's data distribution histogram were plotted. Histograms and Distribution plots showed various characteristic such as skewness, mean, mode of the data. As observed in the figure below (Figure 13a.), the distribution plot for AIS is little right skewed(positive skewed) while the plot of AII is little left skewed(negative skewed). Similarly in Figure 13b. shows Histogram and Distribution plots of Absolute Group Score and Meeting times. The distribution plot for AGS showed that the values were normally distributed.

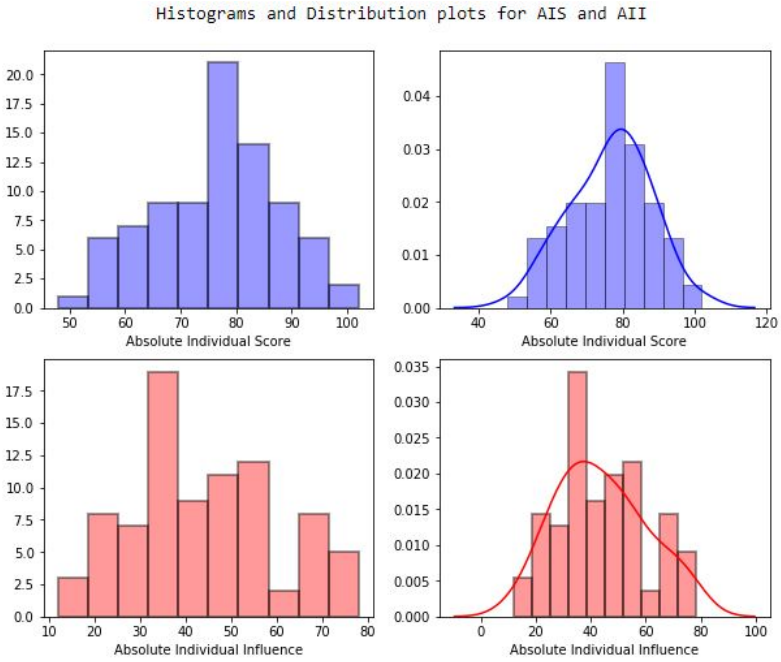


Figure 13a. : Histograms and Distribution plots for AIS and AII

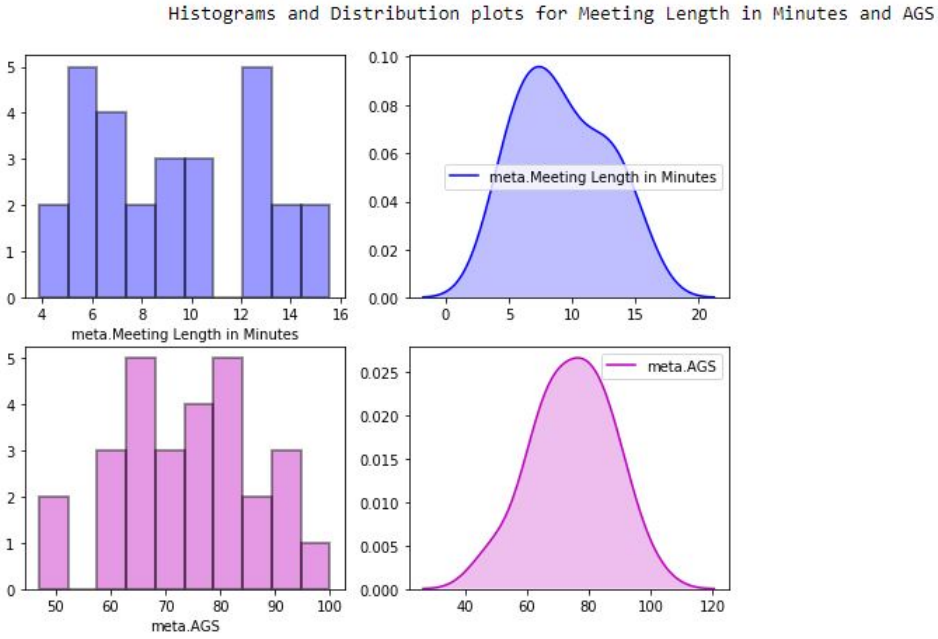


Figure 13b. : Histograms and Distribution plots for Meeting Length in Minutes and AGS

3.3.3 Clustering on Conversational Data (Group Level Information)

- K-Means Clustering : To cluster conversational data via K-Means the data features needed to be reduced. This was achieved by removing similar or unnecessary features and using PCA to reduce the number of dimensions. After performing PCA, a new set of variables was generated. To maximize explained variance, the first three components of the transformed dataset were used. The first three components consisted of about 80% of the total variance of the dataset. .

	Component 0	Component 1	Component 2	Component 3	Component 4
1.Pink.1	-2.741798	-0.084602	-0.383955	1.596541	-0.209844
10.Orange.1	-0.838581	-1.685319	-1.700899	-0.967307	0.142320
11.Pink.1	3.146702	1.378457	-0.343839	1.518624	1.959076
12.Blue.1	-1.889511	0.510418	-1.642653	0.477164	-0.352864
13.Yellow.1	0.416884	-1.265654	0.137690	0.051090	0.403291

Figure 14: New Features for Conversational Level Information after using PCA

To estimate the optimal number of clusters, Within Cluster Sum Squared(WCSS) method was used. The graph obtained using the WCSS method (Figure 15) was used to compute the optimal cluster value of 4. Note k cluster value of 5 can also be used as it is also an inflection point.

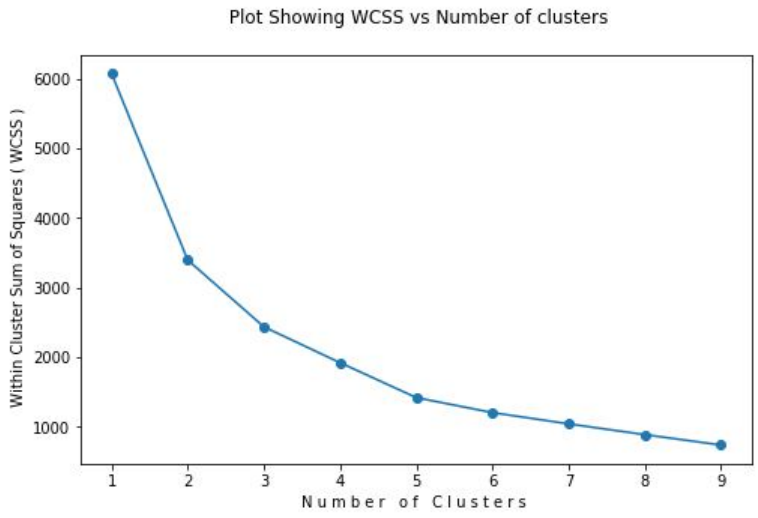


Figure 15: WCSS VS Number of Clusters

With a value of clusters set to 4, and the dataset comprising 3 features and 28 observations (28 groups), each observation was assigned to a unique cluster-based on the observation's distance from the centroid vector. After KMeans Clustering was performed Frequency of observations in each of the clusters is summarized in Table 8 below :

Cluster Number	Frequency
Cluster 0	10
Cluster 1	14
Cluster 2	3
Cluster 3	1

Table 8 : Cluster Number and Frequency Counts

The result obtained was visualized using a scatter plot using 3d and 2d scatter plots. As the new dataset obtained after performing PCA comprised of observation/vectors which are 3-dimensional, 3d plots were used. However, 3d plots are difficult to visualize. For sake of simplicity 2d plots were also used and were considered as a baseline for any comparisons.

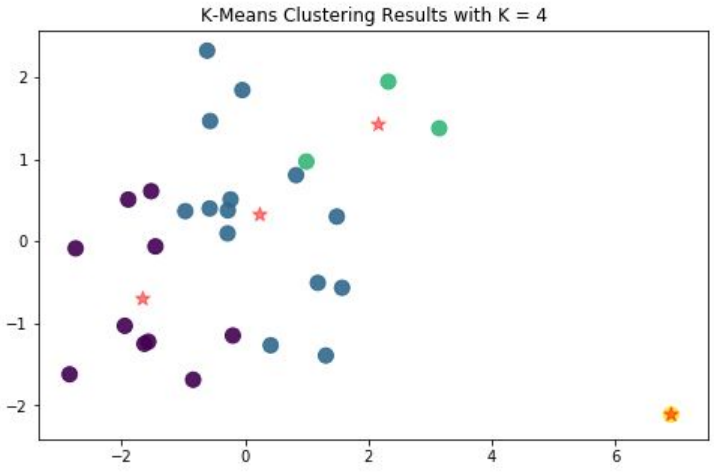


Figure 14a : K Means Plot (2-D) (Group Level Data)

The above plot obtained represents the four clusters. For example, purple dots represent Cluster 0, yellow dots represent Cluster 3, and red stars represent the centroid value. Observations which belonged to a particular cluster were essentially similar in nature.

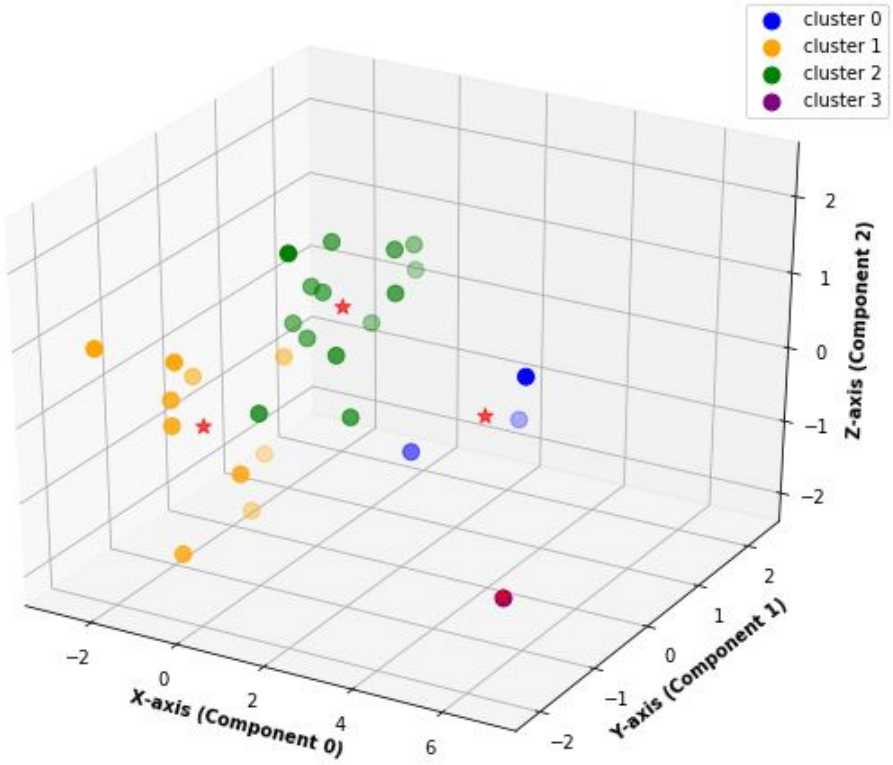


Figure 14b : K Means Plot (3-D) (Group Level Data)

- C-Means Clustering : Instead of using Hard Clustering approaches, fuzzy clustering was used to observe any differences in performance between the two clustering approaches. C Means Clustering weighs each point against every cluster i.e each observation has a probability of belonging to each cluster, rather than completely belonging to just one cluster. In C-Means, an observation can belong to multiple clusters. After C-Means Clustering was performed Frequency of observations in each of the clusters is summarized in Table 8 below :

Cluster Number	Frequency
Cluster 0	4
Cluster 1	5
Cluster 2	9
Cluster 3	10

Table 8 : Cluster Number and Frequency Counts

The clusters obtained Via C-Means were different when compared to KMeans as none of the cluster values were even close to being similar. As C-Means Clustering is a more flexible form of clustering, overlaps were quite predominate between the observations. The centroid value of a given cluster was also different from centroid cluster values obtained using K-Means (Ref Figure 14, 15).

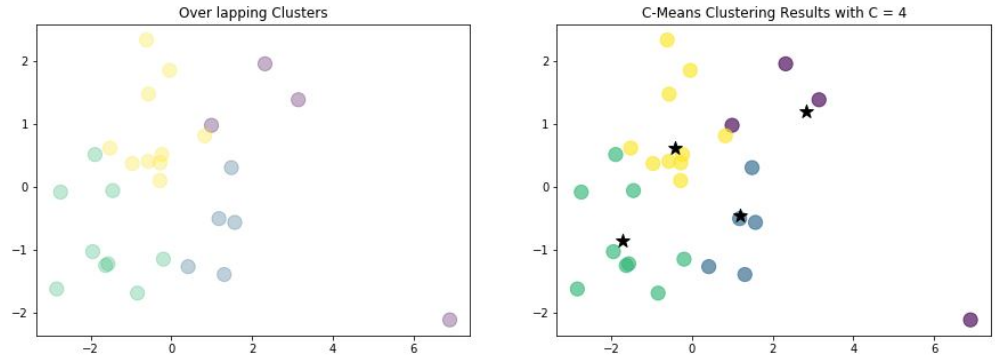


Figure 15a : C Means Plot (2-D) (Group Level Data)

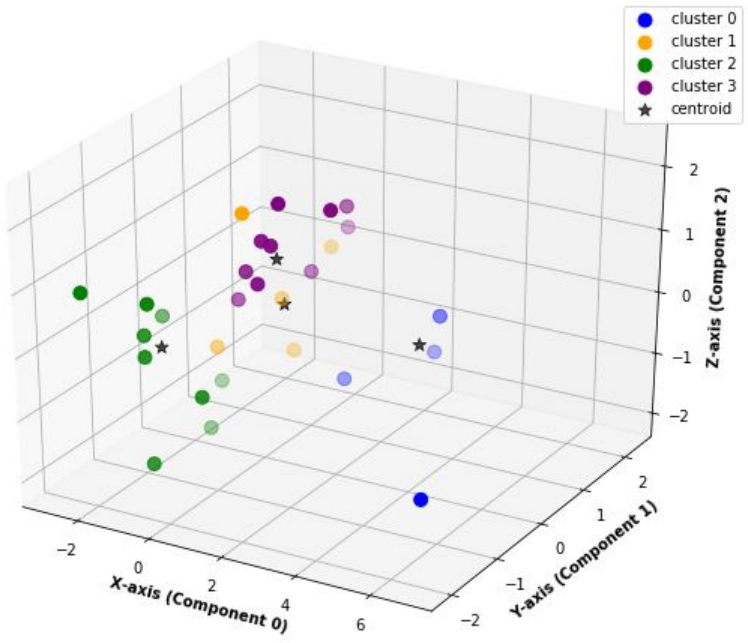


Figure 15b: C Means Plot (3-D) (Group Level Data)

- Hierarchical Clustering : To analyze the similarity between groups Hierarchical clustering was also used. The aim was to make clusters that can segment similar groups together. As Hierarchical Clustering computes the similarity between each of the clusters and joins the two most similar clusters at every step, the objective to segment similar groups was achieved. Using dendrograms with different linkage criteria such as average and single, hierarchical relationships were analyzed.

  Dendrogram can also be used to identify and outlier. In Figure 16a. Group 25 was an outlier (simplicifolious or single leaf'd).

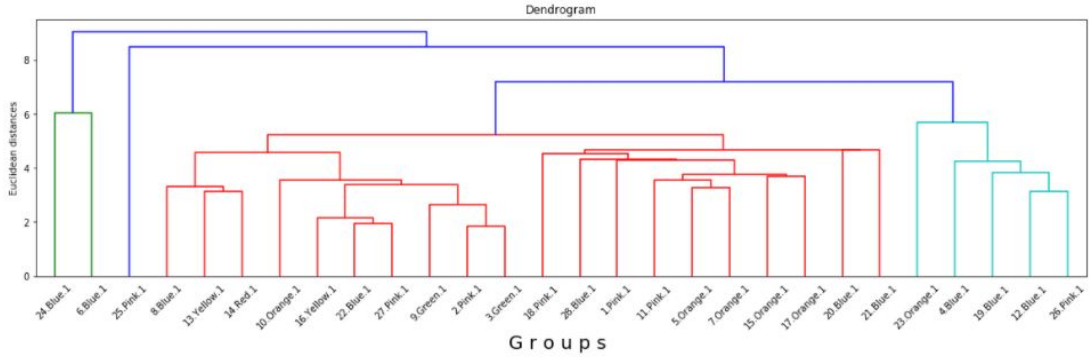


Figure 16a : Dendrogram Representing Nested Clusters (Linkage = Single)

In Figure 16b. Group 1. Group 20, and Group 21 formed one cluster which was merged with two other sub-clusters at a distance value of approximately 8 into one giant cluster.

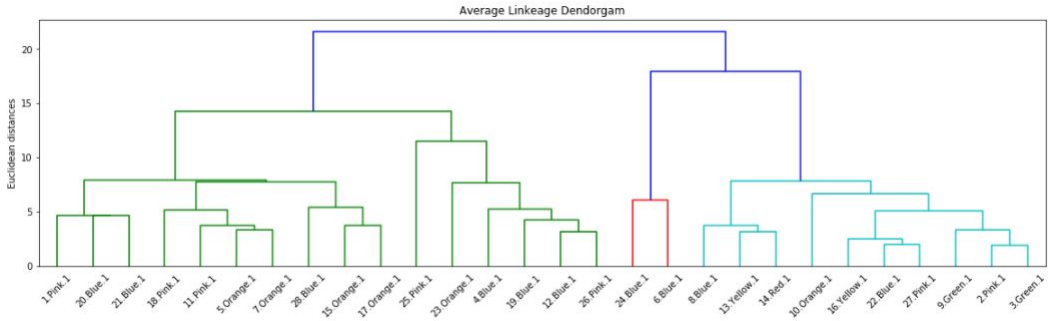


Figure 16b : Dendrogram Representing Nested Clusters (Linkage = Average)

3.3.4 Clustering of Participant Data (Speaker Level Information) using Average TF-IDF

To cluster participants, an average TF-IDF score of each participant was calculated using utterance level information. Pair-wise cosine similarity was computed for each of the participants which were used as Machine Learning features. A total of 84 observations and 84 features comprised our new dataset.

To reduce the number of features, PCA was performed. After performing PCA, a new set of variables was generated. To maximize explained variance, the first three components of the transformed dataset were used. The first three components consisted of about 80% of the total variance of the dataset. From the graph obtained using WCSS method the optimal cluster value of 4 was computed.

Similar to clustering Conversational level data, K-Means and C-Means Clustering were used. The results obtained by these clustering algorithms were extremely different. K-Means clustering failed to identify any distinct clusters or groups of participants which were similar to each other. Using K-Means clustering didn't produce a reliable model for analysis(Ref. Figure 17).

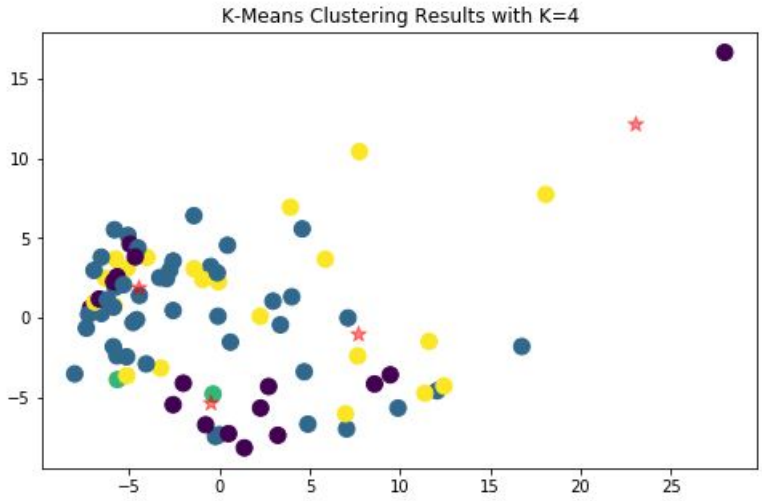


Figure 17 : K-Means for Participant Vector

C-Means Clustering on the other hand produced distinct clusters for participants based on their corresponding similarity scores. The following figure shows the results obtained using C-Means clustering :

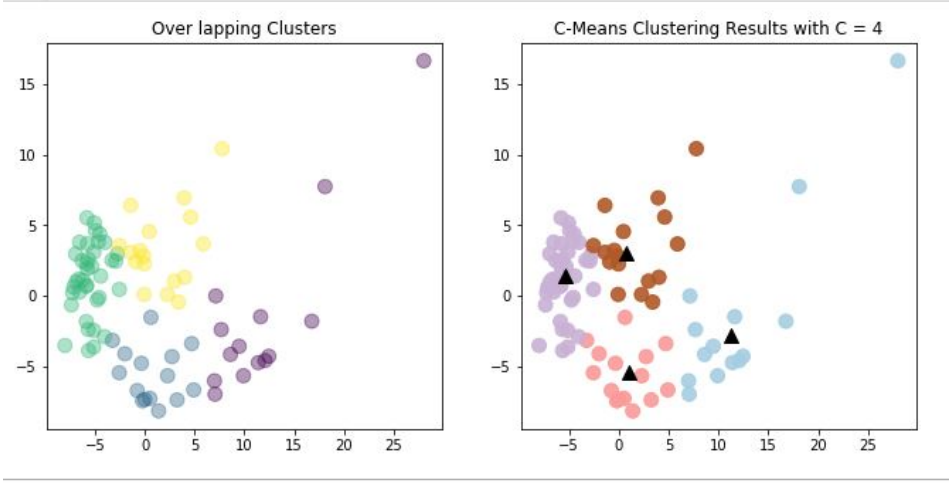


Figure 18 : C-Means for Participant Vector

4 Conclusion and Future works

In this project variety of experiments were performed on conversational data. Conversational data (Stanford Politeness Corpus(Stack Exchange and Wiki)) were analyzed using Politeness Strategies and methods described in [2]. The results presented by Table 4 in [2] were somewhat replicated by the experiments conducted. Most of our models outperformed our baselines models built using Linguistic Markers. Moreover results obtained from other models such as BoW, and TF-IDF outperformed our baseline models for Wiki Politeness Corpus. However, for Stack Exchange, these models performed averagely when compared to baseline models.

The politeness models obtained can be extremely useful as they can be used to annotate conversational data and predict whether a conversation is polite or not. As proposed in [2] models can be built to annotate data which can reduce the pain stack effort to annotate a massive amount of corpus. In fact, to annotate conversations Wikipedia Talk Corpus, a Corpus with about 400,000 has a similar structure to those of the Stanford Corpora. With the use of Politeness models created and Semi-Supervised Learning, utterances or conversations can be classified as Polite or Impolite. Using Semi-Supervised Learning the performance of the model can be evaluated. To increase performances models from both Stack Exchange and Wikipedia Politeness can be combined to predict politeness labels for Wikipedia Talk Pages Utterances.

For GAP Corpus using EDA and different clustering techniques gave us information such as participant's similarity, group similarity, etc. For future works the calculated average TF-IDF aggregated over utterances, combined with other Machine Learning models can be used to predict a group's performance.

As the GAP corpus consists of multiple speakers, Coordination scores can also be incorporated to improve Machine Learning models.

References

[1] Ajitesh Kumar.2021. Python – Text Classification using Bag-of-words Model. *Data Analytics Data, Data Science, Machine Learning, AI* (August 2021).

[2] Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. *The Association for Computer Linguistics* (2013), 250–259. DOI: http://dx.doi.org/10.48550/arXiv.1306.6078

[3] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning: With Applications in R*, New York, NY: R. Springer Publishing Company, Incorporated.

[4] Jason Brownlee. 2017. How to Encode Text Data for Machine Learning with scikit-learn. *Machine Learning Mastery* (September 2017).

[5] Jonathan P. Chang, Caleb Chiam, Liye Fu, Andrew Z. Wang, Justine Zhang, and Cristian Danescu-Niculescu-Mizil. 2020. ConvoKit: A Toolkit for the Analysis of Conversations. (2020). DOI:http://dx.doi.org/10.48550/ARXIV.2005.04246

[6] Jurij Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2015. *Mining of Massive Datasets* 2nd ed., Cambridge: Cambridge Univ. Press.

[7] McKenzie Braley and Gabriel Murray. 2018. The Group Affect and Performance (GAP) Corpus. *Proceedings of the Group Interaction Frontiers in Technology (GIFT'18)* (October 2018). DOI: http://dx.doi.org/10.1145/3279981.3279985

[8] Muller Andreas C. and Sarah Guido. 2018.*Introduction to machine learning with python: A guide for data scientists* First., Sebastopol, CA: O'Reilly Media, Inc.

[9] Penelope Brown and Stephen C. Levinson. 1994. *Politeness: Some Universals in Language Usage*, Trowbridge, Wiltshire: Cambridge Univ. Press.

[10] Rohit Madan. 2019. TF-IDF/Term Frequency Technique: Easiest explanation for Text classification in NLP using Python (Chatbot training on words). *Medium* (May 2019).

[11] Steven Bird, Edward Loper, and Ewan Klein. 2016. *Natural Language Processing With Python* 3rd ed., Sebastopol, CA: O'Reilly Media.

[12] Uliyana Kubasova, Gabriel Murray, and McKenzie Braley. 2019. Analyzing Verbal and Nonverbal Features for Predicting Group Performance. (June 2019). DOI: http://dx.doi.org/10.48550/arXiv.1907.01369