A Comparative Study of Machine Learning Classifiers for Speaker's Accent Recognition

Hasibul Hasan Sabuj[1], Paul Richie Gomes[2], Kazi Ehsanul Mubin[3], Akram Hossain[4],
Samin Yeasar Seaum[5], Md Tanzim Reza[6] and Md. Golam Rabiul Alam[7]
[(1,2,3,4,5,6,7)]Department of Computer Science and Engineering, BRAC University, 66 Mohakhali, Dhaka 1212, Bangladesh.
Email: [1]sabuj.hh@gmail.com, [2]paulrichiegms@gmail.com, [3]kazimubin.46@gmail.com,
[4]akram.nayem01@gmail.com, [4]Seaum333@gmail.com, [6]rezatanzim@gmail.com and [7]rabiul.alam@bracu.ac.bd

2023 IEEE World AI IoT Congress (AIIoT) | 979-8-3503-3761-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/AIIoT58121.2023.10174511

***Abstract*—Recognizing a speaker's accent is crucial in speech technology, natural language processing, and forensic linguistics, especially in English. With the English language having a diverse range of accents and being the most widely spoken language globally, it is vital to develop accurate accent recognition systems for speech recognition and language learning platforms. Our dataset comprises six different accents, including French English, Spanish English, German English, Italian English, UK English, and US English. Our proposed model utilizes Mel-Frequency Cepstral Coefficients (MFCCs) to generate numerical data from raw audio and predicts the speaker's accent from our classification categories. This paper presents a comparative study of machine learning classifiers for speakers' accent recognition. We conducted multi-class classification on our dataset and implemented necessary data preprocessing techniques to address the class imbalance and improve data quality. Eight different machine learning models, including Support Vector Machine, Random Forest, and XGBoost algorithm, were trained and compared for their performance. Random Forest achieved the best performance with an accuracy of 95.28%, and we provided a detailed analysis of each model's strengths and weaknesses and their performance in each accent class.**

***Index Terms*—Speaker's Accent Recognition, Machine Learning Algorithms, Random Forest, Multiclass Classification Problem**

I. Introduction

The audience must acquire perfect insight into the language being used to recognize a speech being delivered by someone. Without clear language, communication is difficult and may be impossible if the meaning on both ends is not the same. According to [1], accents typically consist of recurrent word mispronunciations, misspellings, and minor grammatical errors. The manner in which a word is pronounced can reveal a speaker's linguistic, social, or cultural background [4]. The current methods for determining a speaker's accent may call for specialized linguistic knowledge, an analysis of the specific speech contrasts, and frequently a great deal of pre-processing.

In linguistics and phonetics, an accent can refer to any audible variation from a language's standard pronunciation or to the pronunciation characteristics of a particular group of people. More than three hundred million people speak English as their first language, of the billions who use it as their second language. English has had a significant influence on other parts of the world over the past few hundred years, and many words have been adapted from other languages, causing it to be quite variant. It has a very long history and is one of the world's three most widely spoken languages. Effective human communication, speech technologies like voice recognition software, and accent recognition, are often hindered by the variability of the speaker's accent. [5] advances our understanding of accents that differ among speakers from a cognitive standpoint. Machine Learning techniques can be used to examine and validate speech and the speaker. Certainly, it is possible to train a program to recognize speakers' accents with adequate data. Our study is useful for researchers and practitioners working on speaker accent recognition, as well as those developing speech recognition systems or language learning platforms that require the recognition of different language accents.

A. Motivation

Accent differences among speakers of the same language frequently make communication difficult. Therefore, accurate accent recognition is essential, and numerous tests have been done on it recently. Accent recognition can also be considered crucial in cases like security features, remote shopping, online banking, etc. The process of speaker recognition can identify the speaker's accent from the recorded voice signal, [3]. In previously studied approaches, the concept was to determine only the US-English accent and Non-US Accents and it did not solve that problem completely. So, we tried a multi-class classification to specify all the accents-Spanish, French, German, Italian, UK-English, and US-English accordingly so that all the accents could be differentiated.

B. Contributions

In our approach, we conduct experimentation on an audio dataset using eight different and unique machine learning algorithms to classify speech into multiclass classification, recognizing all the individual accents present there in the dataset. We conduct feature extraction, finding MFCCs on the dataset, followed by an upsampling procedure using SMOTE (Synthetic Minority Over-Sampling Technique), encoding the data set as well as scaling it, to split it for training and testing. Then train and test the machine learning algorithms to find the best one. The key contributions of this research are summarized below-

979-8-3503-3761-7/23/$31.00 ©2023 IEEE

- We introduced a multiclass classification approach in this paper, which is completely a new approach for accent recognition, compared to previously attempted binary classification approaches.
- We specified each accent individually. Until our work, all approaches showed only US accent and non-US accent classification. Whereas, we specified all the non-US accents as Spanish-English, French-English, German-English, Italian-English and UK-English.
- Our study provides a detailed explanation of the performance of the implemented classification Algorithms and the factors that influenced their accuracy on our dataset.

II. Related Works

The goal for [1] was to identify the region of the spoken accent. Using the same dataset as ours, they proposed a binary classification of the speech from the audio dataset converted into MFCC. K-NN or K-Nearest Neighbor was the only ML algorithm used in order to classify if the English accent belonged to the US or Non-US. There arose a few complications including, the accuracy results from one model were insufficient for any comparison. One complication that needed to be considered was, the data number was low for some attributes and it needed scaling. In our procedure, we performed a multiclass classification differentiating and recognizing every accent individually. Also, considered the mentioned complication and scaled the data accordingly.

With the application of five classifiers, [2]. determined and extracted signal features in order to perform speaker accent identification addressing MFCCs. They compared the classifiers based on computation time and accuracy. The classifiers used are, LDA, QDA, SVM(RBF), SVM(PLY), and K-NN which did a binary classification to determine the US and Non-US accents. The comparison resulted in K-NN being the best algorithm based on the accuracy of the classification. Meanwhile, in our experimentation, we considered the multiclass classification of all the accents present in the same dataset and used eight unique ML algorithms in terms of performing the classification. We compared the accuracy of the algorithms which resulted in finding the best algorithm to be the Random Forest model.

Seven Machine Learning algorithms are employed in [3], to recognize speaker accents. A binary classification was done in this paper, to identify the accents of English of the people from various countries. The one belonging to the US was considered a US accent and the rest were considered Non-US accents. The seven classifiers they used are, MLP or Multilayer perceptron algorithm, Random Forest algorithm, Radial Basis Function or RBF algorithm, Decision Tree algorithm, K-NN algorithm, Naive Bayes algorithm, and Logistic Model Tree or LMT algorithm. Meanwhile, we used eight different models in our experiment to perform a multiclass classification of the accents. The dataset they used for MFCC is on the UCI ML Repository. The performance metrics they used are Kappa statistics, Mean Absolute Error, Root Mean Squared Error, i.e. the error metrics which we did not use. We determined the accuracy of the classifier and visualized it through Confusion Matrix and ROC-AUC.

III. Methodology

Firstly, the raw dataset containing audio files was taken from an open-source repository. The MFCCs of the dataset got used in order to proceed with the experimentation. Taking the numeric data found in the feature extraction part, the data preprocessing was initiated. In the data preprocessing section, the data upsampling process was done starting with SMOTE, then feature encoding, and feature scaling, and the process ended with the data being split into train and test portions. The preprocessed data that was kept to train, was then passed through the classifiers.

We used eight machine learning algorithms as classifiers for the multi-class classification of the data. The algorithms used are- K-NN, Decision Tree, SVC, Random Forest, Gradient Boost, Ada-boost, XG-boost, and Linear Regression. We evaluate the classification through comparison first, then with the confusion matrix and ROC-AUC.

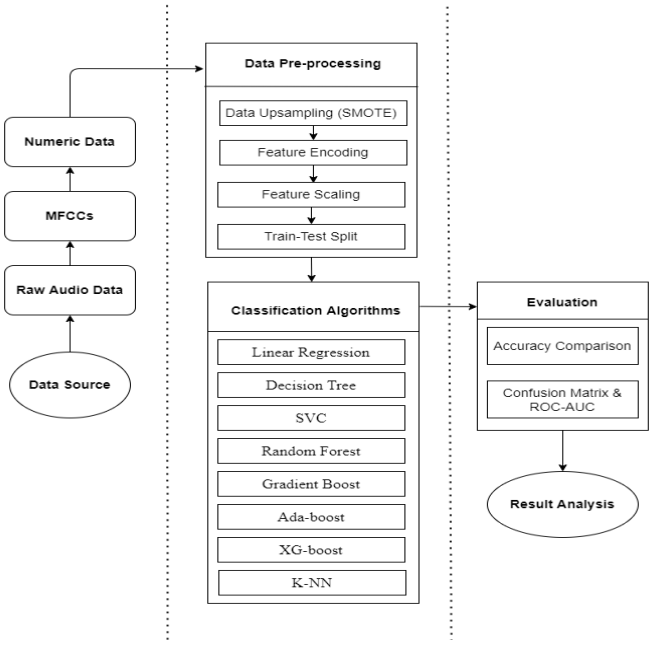

Fig. 1. Top-level overview of the proposed model

In fine, we determined the best algorithm for conducting this experiment. The workflow diagram given in Figure 1 guides the steps to complete the entire experiment step by step through the explained procedure.

A. Dataset description

Our chosen dataset contains 165 recordings of English speakers with American accents and 164 recordings of English speakers with accents from other locations. 22 speakers provided 329 audio data in total. Without any background noise, 11 of these lectures are by men and 11 by women. The dataset has 12 attributes that can be used for categorization. Each voice was allocated a total of 15 words. In terms of

accent, this is a balanced design, but not according to gender. We may solely concentrate on accent identification in this situation. Each soundtrack vector has over 30,000 entries on the time domain, despite the fact that the soundtracks only last for 1 second at a sample rate of 44,100 Hz. Other than the American accent, separate voices for each word were recorded from individuals of Spanish, French, German, Italian, and English ethnicity. The MFCC or Mel-Frequency Cepstral Coefficients approach has been used previously to extract these properties from voice data.

B. Data preprocessing

To fit the data into our classifiers, we need to process the data further, so the preprocessing steps are as follows:

1) Feature Extraction: According to studies, the MFCC algorithm is effective for extracting features from voice signals. The speech signal in the time domain, which is only a time series of the voice's amplitude, easily produces a huge number of variables when taken into account from the dataset description. However, feature extraction reduces high dimensionality. Because we want the method to lower the dimensionality and also preserve the features of the unique voice as much as possible. This algorithm should differ from popular techniques like principal component analysis in terms of speech signals. [2] First, pre-emphasis filtering is conducted, followed by an absolute value finding process through windowing, then wrapping to the auditory frequency scale right before discrete cosine transformation is done, and lastly, the first q MFCCs are returned to be used in the experimentations. The formula determining the MFCCs for M filter outputs is-

$$c[q] = \sum_{m=0}^{M-1} \left[S[m] \cos \left[\frac{rq\left(m - \frac{1}{2}\right)}{M} \right] \right], 0 < m \leq M \quad (1)$$

The primary concept of MFCC is to represent the speech signal in a compact form and map the transformed signal in hertz onto the Mel-scale.

2) Data Upsampling: Upsampling involves injecting artificially produced data points into the dataset. The label counts are nearly equal after this procedure. Due to this equalization process, the model will not favor the class with the majority of members.

Here, the dataset before and after the upsampling is graphed in Figures 2 and 3 with the accent languages on the x-axis and frequency along the y-axis. We upsampled the data using SMOTE. The Synthetic Minority Over-Sampling Technique is referred to as SMOTE in [6]. SMOTE provides new observations in the form of a randomly selected point and its closest neighbors rather than just replicating existing ones, which is the same basic task as basic resampling by generating new data points for the minority class.

3) Feature encoding: We must have all of our data in numeric format since machine learning models can only learn from numeric data. Therefore, we must encode the categorical information in our dataset as follows: 'ES', 'FR',

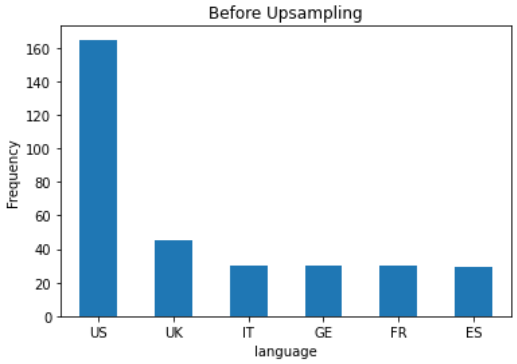

Fig. 2. Before Data Upsampling

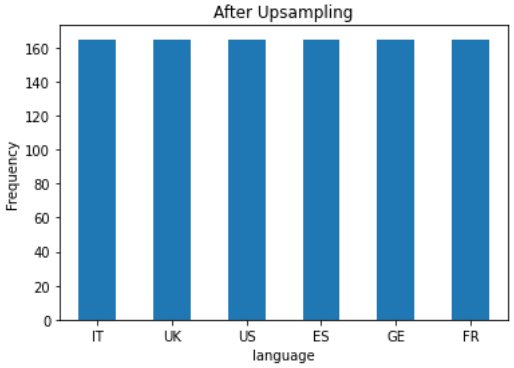

Fig. 3. After Data Upsampling

'GE', 'IT', 'UK', and 'US' which are Spanish, French, German, Italian, UK-English, and US-English respectively, into numeric representations. Now, all of the provided categorical data are converted to numbers and each is assigned a value. To turn those changeable inputs into numbers, we used a label encoder. The scikit-learn label encoder [7] that we utilized operates by using the steps shown below. In other words, it switches out the strings for numerals that range from 0 to (number of classes -1).

4) Feature Scaling: Data standardization heavily relies on this scaling method. Smaller standard deviations make the effects of the outliers less noticeable because the data is scaled within a predetermined range. The importance of machine learning to those who estimate is well-known. We utilized the Standard scalar from scikit-learn as our feature scalar, which calculates a sample's standard score using the following formula:

$$z = \frac{x - u}{s} \quad (2)$$

5) Train-Test Split: We adopted a training-then-testing strategy in our approach. In order for a machine-learning model to become flawless, we must train it on a large amount of data than testing data. We used 70% of the total 990 data or instances for training, while the remaining 30% were used for testing. In other words, 693 data are used for training, and the remaining 297 examples are to test the model.

C. Model Specification:

This section will provide a detailed description of the machine-learning algorithms that were utilized during the research. We will discuss the advantages and limitations of each model to provide a better understanding of its suitability for the task at hand.

1) K-Nearest Neighbor Algorithm(KNN): The KNN algorithm is a machine learning technique that is non-parametric and is often applied to create both binary and multi-class classifiers. The foundational idea of KNN is to find the closest data point or clustered neighbor class for any data point to predict its class. To calculate the closest distance, KNN uses Euclidean distance as mentioned in [8]. The formula for this operation is given below:

$$(x_i, x_j) = \sqrt{\sum_{r=1}^{n} w_r \left(a_r\left(x_i\right) - a_r\left(x_j\right)\right)^2} \tag{3}$$

K is a hyperparameter that can be set by the user. For large datasets, KNN can be computationally expensive. It also requires the data to be normalized and scaled properly since KNN is sensitive to differences in the scale of the input features.

2) Decision Tree Algorithm: The decision tree algorithm represents decisions and their possible consequences in a tree-like structure. Each of the nodes in the tree corresponds to a decision based on the features or attributes while each branch represents the outcome of that decision [9]. Based on the salient traits, the algorithm divides the dataset into subsets until it reaches a stopping criterion. In the end, each leaf node represents a predicted outcome or class as a label.

3) Support Vector Machine(SVM): SVM is a machine learning algorithm that locates the hyperplane in multidimensional space that best divides the data points of various classes. SVM is exclusive to finding classification for high dimensional datasets, where other machine learning models might struggle to find the boundary line of the classes. For non-linear classification, it uses kernel trick to make data linearly separable in a higher dimension [10]. which is presented by the following equation:

$$f(x) = \sum_{x_j \in S} \alpha_j y_j K\left(x_j, x\right) + b \tag{4}$$

SVM models are less vulnerable to overfitting compared to other machine learning models. However, it is often quite computationally expensive for large-scale datasets.

4) Random Forest Algorithm: Random Forest Algorithm is an ensemble method that accumulates multiple decision trees to improve the generalization of the model as well as reduce the variance [11]. It can handle both categorical and continuous data and can provide metrics of feature relevance. It uses the Gini index as a measure of impurity. On the other hand, it is computationally expensive and can be tough to get a good result from a highly imbalanced dataset.

5) Gradient Boost Algorithm: Gradient Boosting is an ML technique that uses a sequential approach to accumulate several ineffective models to create a strong predictive model. The algorithm creates a weighted sum of functions that approximates the target function F(x) in an additive manner [13]. Which in turn, results in a model which is highly accurate and less prone to overfitting. F(x) can be determined by the following formula:

$$F_m(x) = F_{m-1}(x) + \rho_m h_m(x) \tag{5}$$

6) Ada-boost Algorithm: Ada-boost, an ensembling machine learning classifier, can result in computing multi-class classification problems by combining multiple weak classifiers. In recent decades, ada-boost shows great promise in classification problems considering it can boost the accuracy of weak classifiers which in return can improve the performance of many complex classification problems [14]. It is sensitive to noisy and distorted data, so it is paramount that data cleaning is properly performed before feeding into the algorithm.

7) XG-boost Algorithm: XGBoost is an advanced implementation of the gradient boosting algorithm proposed in 2016 [15]. XGBoost is known for its efficiency, speed, and accuracy, and is widely used in both industry and academia. The algorithm is designed to handle large datasets and can be parallelized across multiple CPUs or GPUs to further improve performance [13].

8) Linear Regression Algorithm: Linear regression is commonly used for modeling the relationship between a dependent and one or multiple independent variables. The goal is to apply a linear function to the data and filter it in a way that reduces the total amount of error between the predicted and actual values of the dependent variable [16]. The key algorithm for determining linear regression is:

$$\lambda(t) = \mathbf{Y}(t)\boldsymbol{\alpha}(t) \tag{6}$$

D. Evaluation Matrices:

This subsection provides the evaluation metrics that have been used to compare the findings of our implementation.

1) Confusion Matrix: The confusion matrix is a very useful tool to visualize the performance of multi-class classifiers. The confusion matrix compares the predicted results with the ground truth and illustrates the accuracy for each class in the classifier. The confusion matrix is given in figure 5, where X-label is marked with the predicted results and Y-label is marked with the ground truth.

2) Precision, Recall and f1-score: The precision score is the measurement of the accuracy of positive prediction in a classifier. A greater accuracy score denotes a more accurate positive categorization and fewer false positives. It is equated by the following formula in equation 8 where TP is the True Positive and TN is the True Negative. Recall is the ability of a classification model to correctly identify all instances of a class of interest. Equation 9 shows the calculating method of recall score. Finally, if we combine both precision and recall

we can calculate the f1-score which is the overall assessment of the model's performance. The value of the f1-score ranges between 0 and 1. This equation is denoted in equation 8.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{9}$$

3) Reciever Operator Characteristics Curve(ROC): The ROC curve shows how a classifier model's diagnostic capacity changes as its discrimination threshold. A one-vs-all or one-vs-one strategy can be used to build a ROC curve for each class in a multiclass model. The one-vs-one strategy trains a binary classifier for each potential pair of classes, whereas the one-vs-all approach treats each class as positive and combines the remaining classes as negative classes.

4) AUC: The area under the ROC curve is represented by the scalar value AUC. It speaks for the overall performance of the classifier system based on the ROC curve. The value ranges from 0 to 1, with 0 denoting a subpar classifier that incorrectly classifies all negatives as positives and 1 denoting a flawless classifier that consistently predicts the right outcomes. However, a perfect AUC score on the training data could be a sign of overfitting.

IV. Result and Analysis

After the successful execution of all our selected models, we can conclude that the Random Forest algorithm performs best in our selected dataset with an accuracy of 95.28% shown in the following table I. However, it is crucial that we must consider all the algorithms before coming to a conclusion. The comparison chart in figure 4 illustrates how each algorithm performs in our classifier. For numerical comparison, we can view table I Here, it is evident that all but three models managed to reach over 90% accuracy. These models are KNN, Decision Tree, Random Forest, AdaBoost, and XGBoost.

TABLE I
Algorithm Accuracy Table

Algorithms	Accuracy (in percent)
K Nearest Neighbors	91.91%
Support Vector Machine	85.85%
Decision Tree	90.23%
Random Forest	95.28%
Adaboost	92.25%
Gradient Boost	40.40%
XGBoost	91.91%
Logistic Regression	83.50%

Considering that our classifier model is sophisticated to some extent, Random Forest performed the best in our implementation since it is an ensemble learning designed to avoid overfitting and handle complex data. The same reasoning can be used for the models that perform the worst. Gradient Boost, for example, frequently fail to produce the expected results when there is a lack of heterogeneity in the dataset. The same can also be addressed in the case of the Support Vector Machine. Our primary dataset comprised classifications that were imbalanced. To limit the imbalance, we employed an upsampling model, SMOTE. Given this scenario, we may conclude that neither model was able to manage the complexity of our dataset. Furthermore, Logistic regression is typically utilized in binary classification. Given the complex multi-class classification conducted in our dataset, logistic regression may also prove ineffective in terms of providing accurate findings. Our implementation produced the more or less desired results for all of the other models.

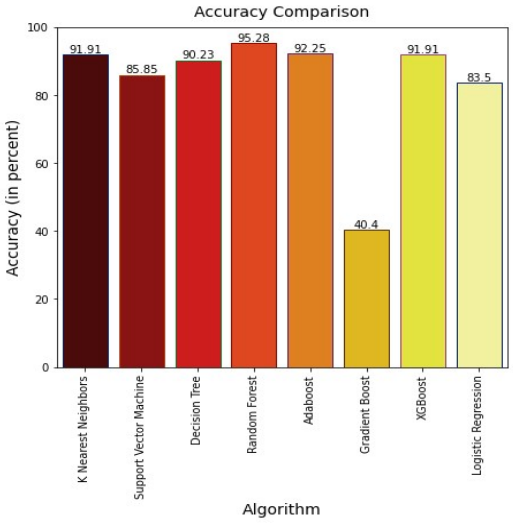

Fig. 4. Accuracy Comparison of Algorithms

After evaluating the best-performing algorithm, we measure the f1 score for the Random Forest algorithm. The following table II shows the performance of our classifier for each class. Considering the lack of heterogeneousness of the original dataset, our final result shows no visible discrepancy and biases for any class. On average the f1-score, precision, and recall of 0.95 are maintained in our implementation. Moreover the confusion matrix in figure 5 shows the comparison between correct and incorrect prediction for all of our classes. The X-axis is the predicted results and the Y-axis is the ground truth marked by a label.

The performance of a classifier algorithm can also be evaluated from the ROC curve and AOC from our experimentation. In this study, the ROC curve for each class was plotted using a one-versus-rest approach, resulting in a curve for each class. The ROC curve for the model in this study is shown in Figure 6, which illustrates how the predictive value of the model changes as the AUC value increases or decreases.

V. Conclusion

In conclusion, our study focused on using machine learning algorithms to recognize the speaker's accent. We utilized

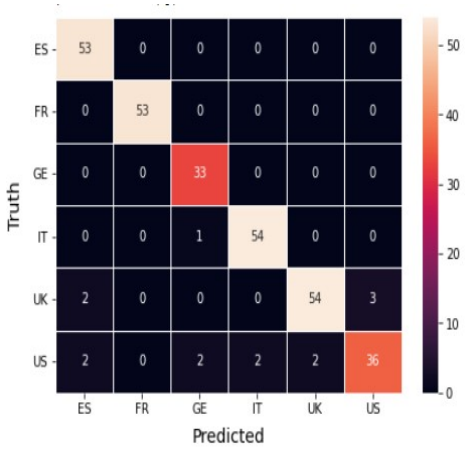

Fig. 5. Confusion Matrix

TABLE II
CLASSIFICATION REPORT

	Precision	Recall	F1-Score	Support
Spanish-English	0.93	1.00	0.96	53
French-English	1.00	1.00	1.00	53
German-English	0.92	1.00	0.96	33
Italian-English	0.96	0.98	0.97	55
UK-English	0.96	0.92	0.94	59
US-English	0.92	0.82	0.87	44
Accuracy			0.95	297
Macro avg	0.95	0.95	0.95	297
Weighted avg	0.95	0.95	0.95	297

the MFCC feature extraction technique and applied feature engineering and upsampling using SMOTE to handle the class imbalance and heterogeneity of the data. We trained eight models, including K-NN, Decision Tree, SVM, Random Forest, Gradient Boost, Ada-boost, XG-boost, and Linear Regression, and the result shows that the Random Forest algorithm performed the best in our dataset. Our study also demonstrated the effectiveness of using SMOTE to handle imbalanced data in machine learning applications. Our study

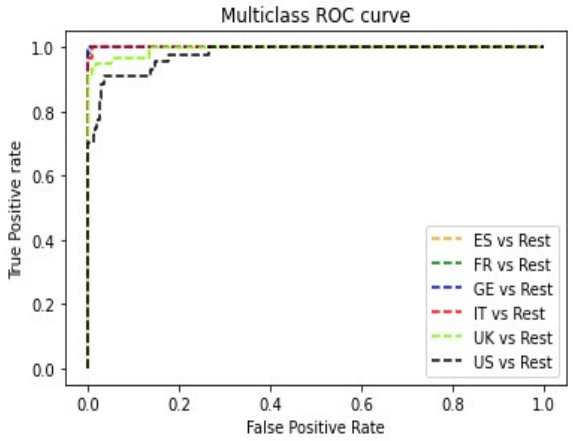

Fig. 6. Multiclass ROC curve

produced some extremely encouraging findings. Yet, it is not at its most optimal state. There is scope for adding other English language accents from various geographic locations to further improve the performance of the machine learning algorithms in recognizing a speaker's accent. Additionally, future research can focus on combining deep neural network classifiers with machine learning algorithms. Overall, our study contributes to the growing body of research on speakers' accent recognition and has potential applications in areas such as speech recognition, natural language processing, and communication technology.

REFERENCES

[1] Taspinar, Y. S., Koklu, M., Altin, M. (2020). Identification of the English Accent Spoken in Different Countries by the k-Nearest Neighbor Method. International Journal of Intelligent Systems and Applications in Engineering, 8(4), 191-194.

[2] Ma, Z., Fokoué, E. (2015). A comparison of classifiers in performing speaker accent recognition using MFCCs. arXiv preprint arXiv:1501.07866.

[3] AYRANCI, A. A., Sergen, A. T. A. Y., YILDIRIM, T. (2021). Speaker Accent Recognition Using MFCC Feature Extraction and Machine Learning Algorithms. International Journal of Advances in Engineering and Pure Sciences, 33, 17-27.

[4] Pedersen, C. and J. Diederich. Accent classification using support vector machines. in 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007). 2007. IEEE.

[5] keno, A. and J.H. Hansen, The effect of listener accent background on accent perception and comprehension. EURASIP Journal on Audio, Speech, and Music Processing, 2007. 2007(1): p. 076030.

[6] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

[7] Bisong, E. (2019). Introduction to Scikit-learn. In: Building Machine Learning and Deep Learning Models on Google Cloud Platform. Apress, Berkeley, CA

[8] S. Sun and R. Huang, "An adaptive k-nearest neighbor algorithm," 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, China, 2010, pp. 91-94, doi: 10.1109/FSKD.2010.5569740.

[9] Charbuty, Bahzad, and Adnan Abdulazeez. "Classification based on decision tree algorithms for machine learning." Journal of Applied Science and Technology Trends 2.01 (2021): 20-28.

[10] S. V. M. Vishwanathan and M. Narasimha Murty, "SSVM: a simple SVM algorithm," Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290), Honolulu, HI, USA, 2002, pp. 2393-2398 vol.3, doi: 10.1109/IJCNN.2002.1007516.

[11] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[12] Bikmukhametov, T., Jäschke, J. (2019). Oil production monitoring using gradient boosting machine learning algorithm. Ifac-Papersonline, 52(1), 514-519.

[13] Bentéjac, C., Csörgő, A., Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review, 54(3), 1937-1967.

[14] CAO, Ying; MIAO, Qi-Guang; LIU, Jia-Chen; GAO, Lin (2013). Advance and Prospects of AdaBoost Algorithm. Acta Automatica Sinica, 39(6), 745–758. doi:10.1016/S1874-1029(13)60052-X

[15] Chen, T., Guestrin, C. (2016, August). Xgboost: A scalable tree-boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

[16] Odd O. Aalen (1989). A linear regression model for the analysis of life times. , 8(8), 907–925. doi:10.1002/sim.4780080803