

Voice-Print Recognition System Using Python And Machine Learning With IBM Watson

Pritam Ahire¹, Hariharan Achary², Sourabh Shirke³, Pratik Kalaskar⁴

Professor, Computer Science, Dr D.Y Patil Institute of Engineering and Technology, Pune, India¹

Student, Computer Science, Dr D.Y Patil Institute of Engineering and Technology, Pune, India²

Student, Computer Science, Dr D.Y Patil Institute of Engineering and Technology, Pune, India³

Student, Computer Science, Dr D.Y Patil Institute of Engineering and Technology, Pune, India⁴

Abstract: With the popularity of the various security system, identify the authentication of an exact user has been a key challenge for each system. The voiceprint recognition technology has some merit over another security system with flexible, economical terms. Therefore using voice as a key security tool will be useful for the ecosystem. In this system, we are going to use some machine learning concept SVM (Support Vector Machine). The SVC will be useful to differentiate the dataset and find out the actual required result so that user can get authentication to access the machine (Files, Folder, Documents), to make the system fully workable it is supported by Python on the frontend, with the help of the python SVC implement of it is successful and from the backend part, we are going to use IBM Watson which will provide actual power to the system. Using the IBM server APIs would be extra key to success to make the system more efficiently get the result as solving machine learning process on the client is hectic and hard to perform, that's why we are using the server of IBM which provides some free tools which are easily accessible and taking advantage of these tools we can easily achieve the required security and result for the system.

Keywords: Machine Learning, Python, IBM Watson Text to Speech, Fuzzy Wuzzy, Skitlearn, Support Vector Machine (SVM), Mel-Frequency Cepstral Coefficients.

I. INTRODUCTION

In this advanced era, when people leverage social networking, online shopping, and online financial transactions without the need of being physically present at places. As a result, identity authentication has become the most critical security activity in the online world. The traditional solution uses a password or a private key (which is encrypted). In fact of this encryption, many people use easy passcode to remember "123456" to crack this kind of code are very easy for the intruder or any 3rd party person to hack out the system. The traditional solutions are a risky affair as the passwords are forgotten or lost and are also prone to hacker attacks.

The best example would be the router password i.e admin and admin which can easily be attacked and all the connections to that device can be hacked out easily within that network.

To overcome these drawbacks, we have decided voice security system. In the system, we have a key advantage that the person needs to be physically present to unlock the system and gain access. But a common question may arise that "I can record the person's voice and play it unlock the device, how it is secured?". The simple solution to its there is a difference playing sound and actual speaking which can be recognized with the SVC (Support Vector Classifier) algorithm and Machine learning which we are using SVM (Support Vector Machine).

With the help of the python framework and using the machine learning concept of SVC (Support Vector Classifier) which will help us to distinguish the dataset and show us the required result with a higher accuracy rate. We have used MFCC (Mel-frequency cepstral coefficients), which plays a key role. Handling this huge dataset on the user side would be a hectic task, therefore to resolve this problem we are using the IBM Watson server. Using their APIs and processing the task on a higher powerful server would be an advantage for the system to work more efficiently. Combining both tools Python and IBM server we acquired a higher accuracy result.

II. ALGORITHM

i. SVC (Support Vector Machine)

SVC algorithm is a supervised machine learning algorithm that can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.

The implementation is based on libsvm. The fit time scales at least quadratically with the number of samples and maybe impractical beyond tens of thousands of samples. Considering large datasets using sklearn SVM LinearSVC or sklearn linear_model SGD Classifier.

$$A_{ij} = \begin{cases} 1, & \text{if } f(x) > 0 \text{ for all } x \text{ on the line segment connecting } x_i \text{ and } x_j \\ 0 & \text{otherwise.} \end{cases}$$

The further explanation of the SVC algorithm can be easily understood with the help of the diagram and its structure.

The following figure will explain the actual machine learning SVC used in the voice-print security system.

Condition 1:

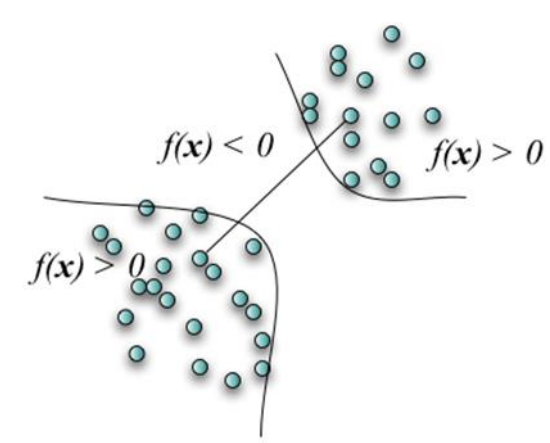


Fig (1): The line segment that connects points in different clusters has to go through a low-density region in data space where the SVDD returns a negative value.

Left $f(x) > 0$ is a dataset and Right $f(x) < 0$ is another dataset in which are easily identical and can be separate out with $f(x) < 0$. This is the most simple and easy test case we can get the result at a single glance. Not much cores or process is not required to figure out the solutions, but it won't be a similar case.

Condition 2:

When the mixed dataset occurs the complexity of the problem increases, eg of mixed data is shown in Fig (2). The machine needs more cores and more data to analyse it and figure out the accurate result [15]. To solve such kind of complex problem we can increase the number of hidden layers in SVM (Scaler Vector Machine).

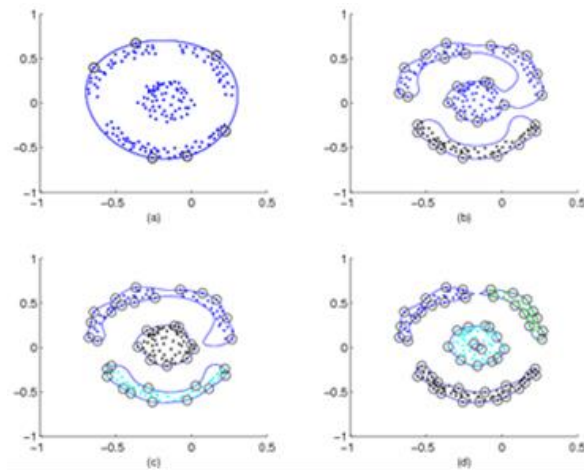


Fig (2) Contours generated by SVDD as γ is increased.

We can use 2 hidden layers, which 1st layer will have 5 neurons present in it and the 2nd layer will have 2 neurons which will help out the output to varying weights and differentiate the datasets.

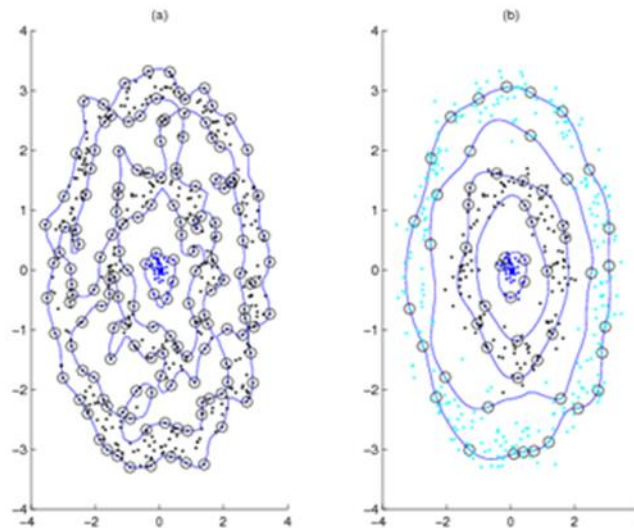


Fig (3) Allowing for outliers allows SVC to separate noisy data

The above Fig (3) shows the result when the dataset is differentiated with the help of the SVC algorithm.

ii. MLP Classifier

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(X)=R^m \rightarrow R^0$ by work through data training, where m is the input value is and the 0 is output size. Given a set of features $X=x_1, x_2, \dots, x_m$ and references γ , it can read a non-linear function scale with a split or a reversal. It differs from ordering in order, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers [17]. Figure 1 shows one hidden layer of MLP by scale extraction.

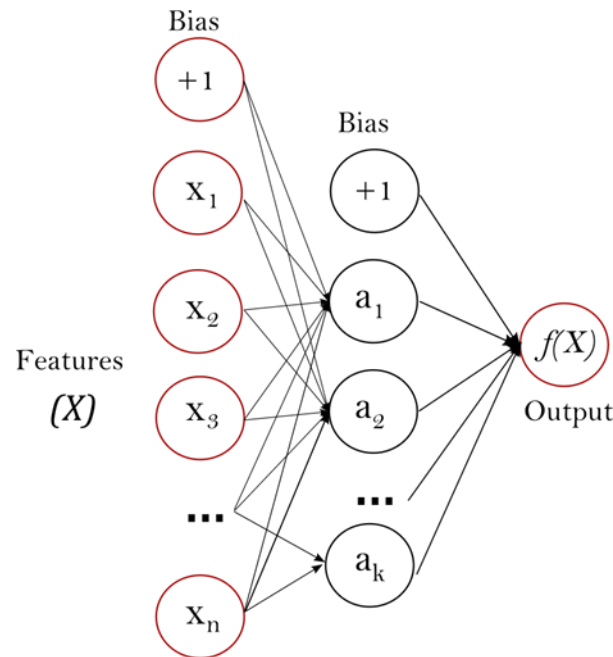


Fig (4) Hidden Layer of MLP

The left layer, known as the input layer, contains a set of neurons $\{ | x_i | | x_1, x_2, \dots, x_m \}$ representing the input features. Each neuron in the hidden layer converts values from the previous layer by a cut line $\omega_1 x_1 + \omega_2 x_2 + \dots + \omega_m x_m$, followed by indirect activation of $g(\cdot): \mathbb{R} \rightarrow \mathbb{R}$ as a hyperbolic tan function. The output layer detects values from the last hidden layer and converts them into output values.

The module contains social attributes breaks coefficients & intercepts. Coefficients list of weight matrices, where the weight matrix i index represents the weights between layer i and layer $i + 1$. Intercepts list of bias vectors, where the vector in the code represents the bias values included in the $i + 1$.

The body of the paper consists of numbered sections that present the main findings. These sections should be organized to best present the material.

It is often important to refer back (or forward) to specific sections. Such references are made by indicating the section number, for example, “In Sec. 2 we showed...” or “Section 2.1 contained a description...” If the word Section, Reference, Equation, or Figure starts a sentence, it is spelled out. When occurring in the middle of a sentence, these words are abbreviated Sec., Ref., Eq., and Fig.

At the first occurrence of an acronym, spell it out followed by the acronym in parentheses, e.g., charge-coupled diode (CCD).

III. ARCHITECTURE

The MFCC feature extraction technique includes windowing the signal, applying the DFT, taking the log of the magnitude, and then warping the frequencies on a Mel scale, followed by applying the inverse DCT [18]. The detailed description of various steps involved in the MFCC feature extraction is explained below.

3.1 MFCC Feature

3.1.1 Pre-emphasis: Pre-emphasis refers to filtering that emphasizes the higher frequencies. Its purpose is to balance the spectrum of voiced sounds that have a steep roll-off in the high-frequency region [13]. For voiced sounds, the glottal source has an approximately -12 dB/octave slope. However, when the acoustic energy radiates from the lips, this causes a roughly $+6$ dB/octave boost to the spectrum [1]. As a result, a speech signal when recorded with a

microphone from a distance has approximately a -6 dB/octave slope downward compared to the true spectrum of the vocal tract. Therefore, pre-emphasis removes some of the glottal effects from the vocal tract parameters. The most commonly used pre-emphasis filter is given by the following transfer function

$$H(z) = 1 - bz^{-1}$$

Where the value of b controls the slope of the filter and is usually between 0.4 and 1.0.

3.1.2 Frame blocking and windowing: The speech signal is a slowly time-varying signal. For stable acoustic characteristics, speech needs to be examined over a sufficiently short time. Therefore, speech analysis must always be carried out on short segments across which the speech signal is assumed to be stationary. Short-term spectral measurements are typically carried out over 20ms windows, and advanced every 10ms [2],[3]. Advancing the time window every 10ms enables the temporal characteristics of individual speech sound to be tracked, and the 20ms analysis window is usually sufficient to provide good Spectral resolution of these sounds and at the same time short enough to resolve significant temporal characteristics. The purpose of the overlapping analysis is that each speech sound of the input sequence would be approximately centred at some frame. On each frame, a window is applied to taper the signal towards the frame boundaries. Generally, Hanning or Hamming windows are used. This is done to enhance the harmonics, smooth the edges, and reduce the edge effect while taking the DFT on the signal [1]. For the continuity of the start and the end, the Hamming window as in the equation is used.

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N+1}\right) \quad 0 \leq n \leq N$$

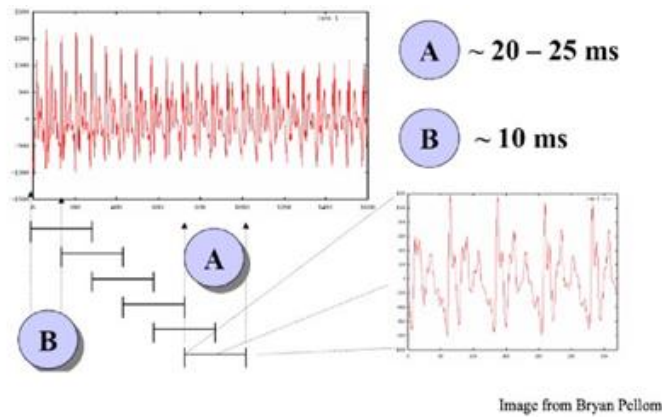


Fig (5) Framing & Windowing

3.1.3 DFT spectrum: Each windowed frame is converted into a magnitude spectrum by applying DFT.

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi nk/N}$$

$$0 \leq k \leq N - 1$$

3.1.4 Mel spectrum: Mel spectrum is computed by passing the Fourier transformed signal through a set of band-pass filters known as Mel-filter bank. A Mel is a unit of measure based on the human ear's perceived frequency. It does not correspond linearly to the physical frequency of the tone, as the human auditory system does not perceive pitch linearly. The Mel scale is approximately a linear frequency spacing below 1 kHz and a logarithmic spacing above 1 kHz [4], the approximation of Mel from physical frequency can be expressed as

$$f_{Mel} = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$

Where f denotes the physical frequency in Hz, and f_{Mel} denotes the perceived frequency [2].

Filter banks can be implemented in both the time domain and frequency domain. For MFCC computation, filter banks are generally implemented in the frequency domain. The center frequencies of the filters are normally evenly spaced on the frequency axis [16]. However, to mimic the human ears perception, the warped axis, according to the nonlinear function given in Equation, is implemented. The most commonly used filter shaper is triangular, and in some cases, the Hanning filter can be found [1]. The triangular filter banks with Mel frequency warping are given in Fig (7)

The Mel spectrum of the magnitude spectrum $X(k)$ is computed by multiplying the magnitude spectrum by each of the triangular Mel weighting filters [5],[6].

$$s(m) = \sum_{k=0}^{N-1} [|X(k)|^2 H_m(k)];$$

$$0 \leq m \leq M - 1$$

Where M is the total number of triangular Mel weighting filters. $H_m(k)$ is the weight given to the k th energy spectrum bin contributing to the m th output band and is expressed as:

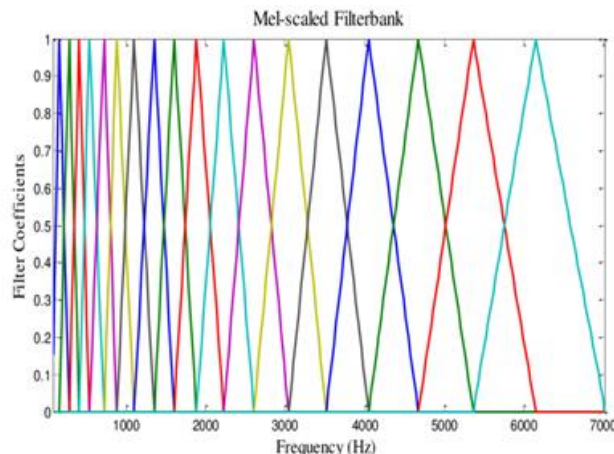


Fig (6) Mel Filter Bank

$$H(m) = \begin{cases} 0, & k < f(m-1) \\ \frac{2(k - f(m-1))}{f(m) - f(m-1)}, & f(m-1) \leq k \leq f(m) \\ \frac{2(f(m+1) - k)}{f(m+1) - f(m)}, & f(m) < k \leq f(m+1) \\ 0, & k > f(m+1) \end{cases}$$

With m ranging from 0 to $M-1$.

3.1.5 Discrete Cosine Transform (DCT): Since the vocal tract is smooth, the energy levels in adjacent bands tend to be correlated. The DCT is applied to the transformed Mel frequency coefficients produces a set of cepstral coefficients. Before computing DCT, the Mel spectrum is usually represented on a log scale. This results in a signal in the cepstral domain with a quefrequency peak corresponding to the pitch of the signal and several formants representing low quefrequency peaks [1]. Since most of the signal information is represented by the first few MFCC coefficients, the system can be made robust by extracting only those coefficients ignoring or truncating higher order DCT components. Finally, MFCC is calculated as [1],[13]

$$c(n) = \sum_{m=0}^{M-1} \log_{10}(s(m)) \cos\left(\frac{\pi n(m - 0.5)}{M}\right);$$

$$n = 0, 1, 2, \dots, C - 1$$

Where $c(n)$ is the cepstral coefficients, and C is the number of MFCCs. Traditional MFCC systems use only 8–13 cepstral coefficients. The zeroth coefficient is often excluded since it represents the average log-energy of the input signal, which only carries little speaker-specific information.

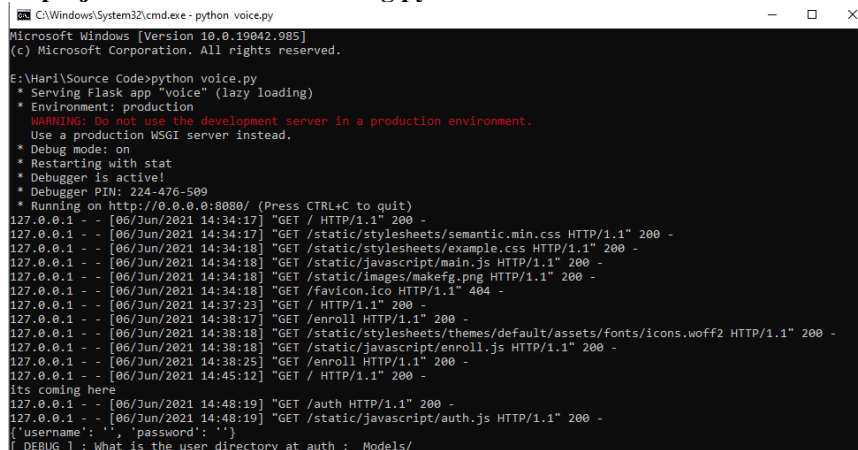
3.1.6 Dynamic MFCC features: The cepstral coefficients are usually referred to as static features since they only contain information from a given frame. The extra information about the temporal dynamics of the signal is obtained by computing the first and second derivatives of cepstral coefficients [7],[8],[9]. The first-order derivative is called delta coefficients, and the second-order derivative is called delta-delta coefficients. Delta coefficients tell about the speech rate, and delta-delta coefficients provide information similar to the acceleration of speech. The commonly used definition for computing dynamic parameter is [7]

$$\Delta C_m(n) = \frac{\sum_{i=-T}^T ki C_m(n + i)}{\sum_{i=-T}^T |i|}$$

Where $cm(n)$ denotes the m th feature for the n th time frame, ki is the i th weight, and T is the number of successive frames used for computation. Generally, T is taken as 2. The delta-delta coefficients are computed by taking the first-order derivative of the delta coefficients.

IV. DEVELOPMENT

Step1: Running the project on localhost:8080 using python

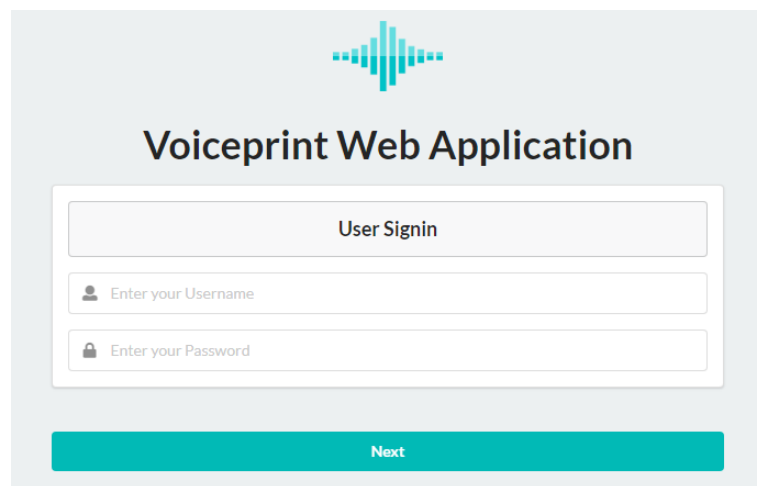
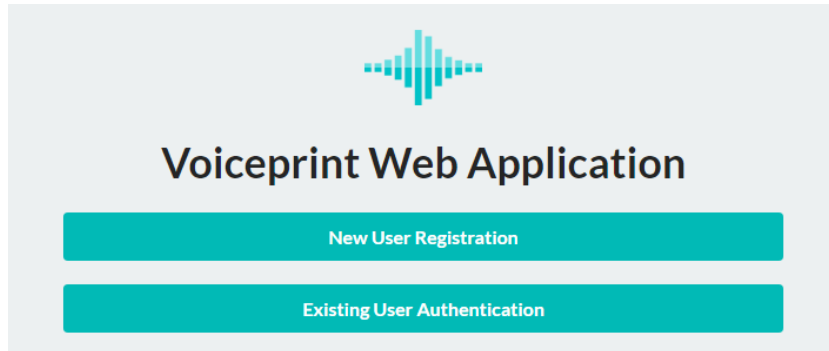


```

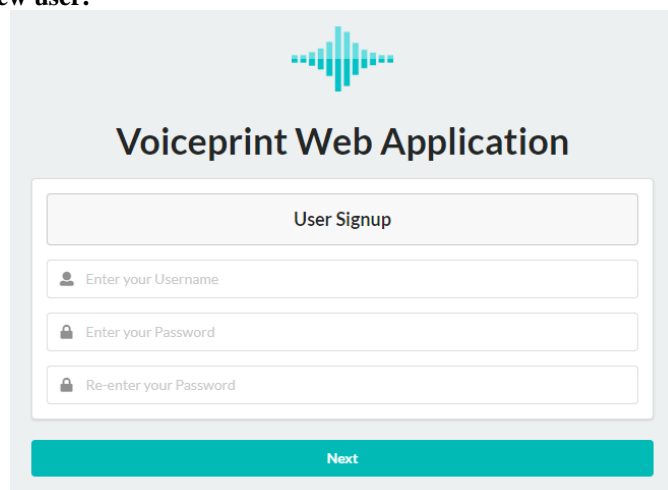
C:\Windows\System32\cmd.exe - python voice.py
Microsoft Windows [Version 10.0.19042.985]
(c) Microsoft Corporation. All rights reserved.

E:\Hari\Source Code>python voice.py
* Serving Flask app "voice" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 224-476-509
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
127.0.0.1 - - [06/Jun/2021 14:34:17] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:34:17] "GET /static/stylesheets/semantic.min.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:34:18] "GET /static/stylesheets/example.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:34:18] "GET /static/javascript/main.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:34:18] "GET /static/images/makefg.png HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:34:18] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [06/Jun/2021 14:37:23] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:38:17] "GET /enroll HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:38:18] "GET /static/stylesheets/themes/default/assets/fonts/icons.woff2 HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:38:18] "GET /static/javascript/enroll.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:38:25] "GET /enroll HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:45:12] "GET / HTTP/1.1" 200 -
Its coming here
127.0.0.1 - - [06/Jun/2021 14:48:19] "GET /auth HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2021 14:48:19] "GET /static/javascript/auth.js HTTP/1.1" 200 -
{"username": "", "password": ""}
[DEBUG] : What is the user directory at auth : Models/
  
```

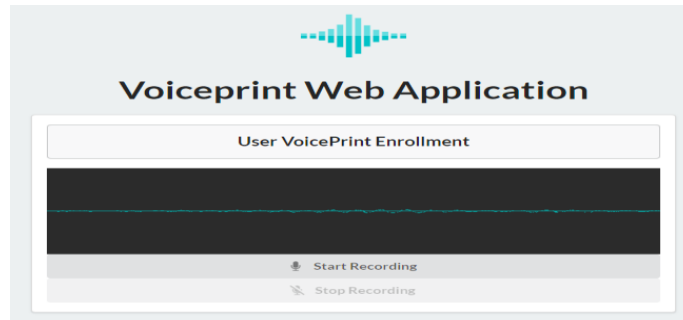
Step 2: Registration of new user or existing user sign in.



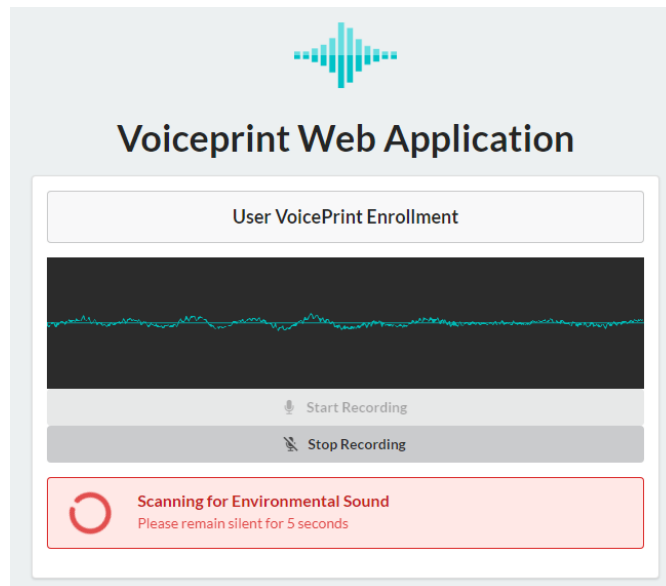
Step 3: Registration of new user.



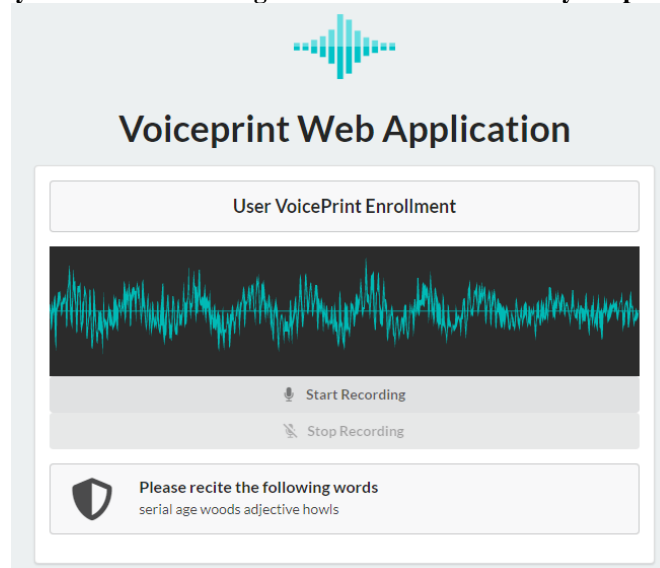
Step 4: Voice Print enrollment



Step 5: Scanning of Environmental Sound before you start recording the voice.



Step 6: After recording of your voice recite the given words to understand your pronunciation.



V. CONCLUSION

In this paper, we try to present a security system using voice biometrics with the help of machine learning and python on the frontend side. The trained data with SVC (Support Vector Classifier). As we all voice security has a low

accuracy rate but using this machine learning technique we can increase this accuracy rate. The main idea of implementing this system is to provide more security to the system. With the help of SVC, MFCC, and MLP Classifier with powered backend IBM Watson. The background noise can be removed with python packages. This package will help us to have a clear output that can be compared with the existing dataset and if the output and present data are matched then we can get access to the system.

However in the case of complex background noise, the method of voiceprint identification will be affected a little bit, but in the future, this complicated situation can be studied and solved with more advanced machine learning concepts.

VI. ACKNOWLEDGEMENT

This paper was successful with the contribution of many people who were directly or indirectly connected to this paper as some concepts can understand with help of the internet and other authors, who had broken-down the concept which made it easy to this paper. I would like to express our gratitude to Prof. Pritam Ahire for their help and advice. He provided many favourable suggestions for the grammar and experimental parts of machine learning concepts.

REFERENCES

- [1] J.W. Picone, Signal modeling techniques in speech recognition. Proc. IEEE 81, 1215–1247 (1993)
- [2] J.R. Deller, J.H. Hansen, J.G. Proakis, Discrete-Time Processing of Speech Signals (Prentice Hall, NJ, 1993)
- [3] J. Benesty, M.M. Sondhi, Y.A. Huang, Handbook of Speech Processing (Springer, New York, 2008)
- [4] J. Volkmann, S. Stevens, E. Newman, A scale for the measurement of the psychological magnitude pitch. J. Acoust. Soc. Am. 8, 185–190 (1937)
- [5] Z. Fang, Z. Guoliang, S. Zhanjiang, Comparison of different implementations of MFCC. J. Comput.Sci. Technol. 16, 582–589 (2000)
- [6] G.K.T. Ganchev, N. Fakotakis, Comparative evaluation of various MFCC implementations on the speaker verification task, in Proceedings of International Conference on Speech and Computer(SPECOM) (2005), pp. 191–194
- [7] L. Rabiner, B.-H. Juang, B. Yegnanarayana, Fundamentals of Speech Recognition (Pearson Education, London, 2008)
- [8] S. Furui, Comparison of speaker recognition methods using statistical features and dynamic features IEEE Trans. Acoust. Speech Sig. Proc. 29, 342–350 (1981)
- [9] J.S. Mason, X. Zhang, Velocity and acceleration features in speaker recognition, in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (1991), pp. 3673–3676