Emotion Recognition - Emotion recognition is the process of identifying human emotion. People vary widely in their accuracy at recognizing the emotions of others. Use of technology to help people with emotion recognition is a relatively nascent research area. Generally, the technology works best if it uses multiple modalities in context. To date, the most work has been conducted on automating the recognition of facial expressions from video, spoken expressions from audio, written expressions from text, and physiology as measured by wearables.

A voice-user interface (VUI) makes spoken human interaction with computers possible, using speech recognition to understand spoken commands and answer questions, and typically text to speech to play a reply. A voice command device (VCD) is a device controlled with a voice user interface.

The importance of emotion recognition is getting popular with improving user experience and the engagement of Voice User Interfaces (VUIs). Developing emotion recognition systems that are based on speech has practical application benefits. However, these benefits are somewhat negated by the real-world background noise impairing speech-based emotion recognition performance when the system is employed in practical applications

Speech Emotion Recognition (SER) is one of the most challenging tasks in speech signal analysis domain, it is a research area problem which tries to infer the emotion from the speech signals.

**Where can we use it ?**

Even though it isn't that popular, SER has entered so many areas these years, including:

**The medical field:** In the world of telemedicine where patients are evaluated over mobile platforms, the ability for a medical professional to discern what the patient is actually feeling can be useful in the healing process.

**Customer service:** In call center conversation may be used to analyze behavioral study of call attendants with the customers which helps to improve the quality of service.

**Recommender systems**: Can be useful to recommend products to customers based on their emotion towards that product.

**Required Dependencies:**

**Librosa**

**Numpy**

**Soundfile**

**Scikit-learn**

**PyAudio**

**The whole pipeline is as follows (as same as any machine learning pipeline):**

**Preparing the Dataset:** Here, we download and convert the dataset to be suited for extraction.

**Loading the Dataset:** This process is about loading the dataset in Python which involves extracting audio features, such as obtaining different features such as power, pitch and vocal tract configuration from the speech signal, we will use librosa library to do that.

**Training the Model:** After we prepare and load the dataset, we simply train it on a suited sklearn model.

**Testing the Model:** Measuring how good our model is doing.

We will use [MFCC](https://en.wikipedia.org/wiki/Mel-frequency_cepstrum" \o "MFCC Definition on Wikipedia" \t "https://www.thepythoncode.com/article/_blank), [Chroma](https://en.wikipedia.org/wiki/Chroma_feature" \o "Chroma Feature definition on Wikipedia" \t "https://www.thepythoncode.com/article/_blank) and [Mel Frequency Cepstrum](https://en.wikipedia.org/wiki/Mel-frequency_cepstrum" \o "Mel Frequency Cepstrum Wikipedia" \t "https://www.thepythoncode.com/article/_blank) as speech features rather than raw waveform which may contain unnecessary information that doesn't help on the classification.

Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC They are derived from a type of [cepstral](https://en.wikipedia.org/wiki/Cepstrum" \o "Cepstrum) representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the [cepstrum](https://en.wikipedia.org/wiki/Cepstrum" \o "Cepstrum) and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in [audio compression](https://en.wikipedia.org/wiki/Data_compression" \l "Audio" \o "Data compression).

In [music](https://en.wikipedia.org/wiki/Music" \o "Music), the term chroma feature or chromagram closely relates to the twelve different [pitch classes](https://en.wikipedia.org/wiki/Pitch_classes" \o "Pitch classes). Chroma-based features, which are also referred to as "[pitch class profiles](https://en.wikipedia.org/wiki/Harmonic_pitch_class_profiles" \o "Harmonic pitch class profiles)", are a powerful tool for analyzing music whose pitches can be meaningfully categorized (often into twelve categories) and whose tuning approximates to the [equal-tempered scale](https://en.wikipedia.org/wiki/Equal_temperament" \o "Equal temperament). One main property of chroma features is that they capture harmonic and melodic characteristics of music, while being robust to changes in timbre and instrumentation.

Then we perform a [grid search](https://en.wikipedia.org/wiki/Hyperparameter_optimization" \l "Grid_search" \o "Grid search " \t "https://www.thepythoncode.com/article/_blank) on MLPClassifier to get the best possible hyper parameters.

**Grid search**

The traditional way of performing hyperparameter optimization has been grid search, or a parameter sweep, which is simply an exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. A grid search algorithm must be guided by some performance metric, typically measured by cross-validation on the training set[3] or evaluation on a held-out validation set.[4]

Since the parameter space of a machine learner may include real-valued or unbounded value spaces for certain parameters, manually set bounds and discretization may be necessary before applying grid search.

For example, a typical soft-margin SVM classifier equipped with an RBF kernel has at least two hyperparameters that need to be tuned for good performance on unseen data: a regularization constant C and a kernel hyperparameter γ. Both parameters are continuous, so to perform grid search, one selects a finite set of "reasonable" values for each, say

{\displaystyle C\in \{10,100,1000\}}C\in \{10,100,1000\}

{\displaystyle \gamma \in \{0.1,0.2,0.5,1.0\}}\gamma \in \{0.1,0.2,0.5,1.0\}

Grid search then trains an SVM with each pair (C, γ) in the Cartesian product of these two sets and evaluates their performance on a held-out validation set (or by internal cross-validation on the training set, in which case multiple SVMs are trained per pair). Finally, the grid search algorithm outputs the settings that achieved the highest score in the validation procedure.

Grid search suffers from the curse of dimensionality, but is often embarrassingly parallel because the hyperparameter settings it evaluates are typically independent of each other.

**Flow Chart:** Structure of Speech Emotion Recognition System

Speech Input

Recognized Emotion

Classifier

Feature

Selection

Feature Extraction