



# Emotion Classification from Audio using SVM

---

Kyrillos Ayamn Shenouda Aziz :2305221

Youssef Hamdy Ibrahim :2305576

Abdelrahman Hesham Salama: 2305239

---

## 1. Introduction

This report explains an audio emotion recognition system using machine learning. The system processes audio recordings of speech, extracts features from them, and then trains a Support Vector Machine (SVM) model to classify the emotional state of the speaker into one of eight categories:

- Neutral
- Calm
- Happy
- Sad
- Angry
- Fearful
- Disgust
- Surprised

---

## 2. Data Loading

The audio dataset is organized in folders, each containing multiple audio files. The `load_data` function:

- Reads all audio files in the dataset folder
- Extracts emotion labels encoded in the filename

- Stores the file paths and corresponding emotion labels
- 

### 3. Audio Preprocessing and Augmentation

To improve model generalization, the audio data undergoes several preprocessing steps:

- **Add Noise:** Random noise is added to simulate real-world variations.
- **Pitch Shift:** The audio pitch is shifted slightly to augment the data.
- **Time Stretch:** The audio speed is changed without affecting pitch.
- **Standard:** The original audio without changes.

These augmentations help the model learn robust features.

---

### 4. Feature Extraction: MFCC

- Mel-Frequency Cepstral Coefficients (MFCCs) are extracted from audio signals.
  - MFCCs effectively represent the audio's timbral texture.
  - For each audio clip, MFCC features are computed and later flattened into 1D arrays to be fed into the model.
- 

### 5. Data Preparation

- All extracted MFCC feature arrays are collected into a single dataset.
  - The labels are one-hot encoded for eight emotion classes.
  - The feature arrays are reshaped to two-dimensional form (samples × features).
  - The dataset is split into training and testing sets (80% train, 20% test), stratified to maintain class distribution.
- 

### 6. Model Training

- A Support Vector Machine (SVM) with a linear kernel is used.
  - The model is trained on the training data.
  - The trained model predicts emotion labels on the test set.
- 

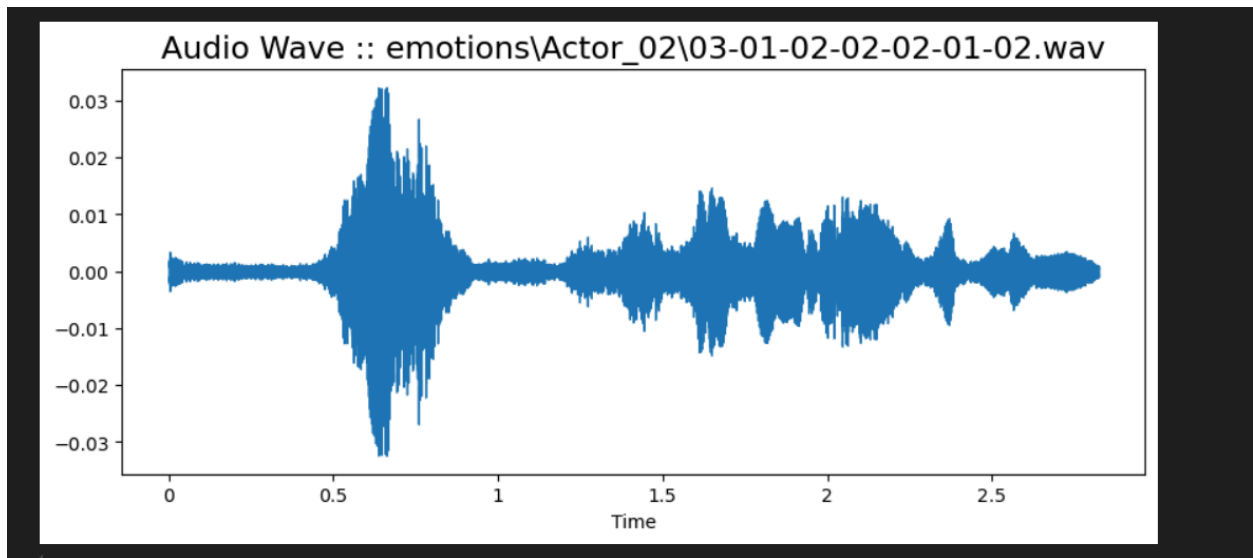
## 7. Evaluation

- Test accuracy is computed to evaluate overall performance.
  - A confusion matrix is plotted using seaborn to visualize per-class performance.
  - The confusion matrix shows how many samples were correctly or incorrectly classified for each emotion.
- 

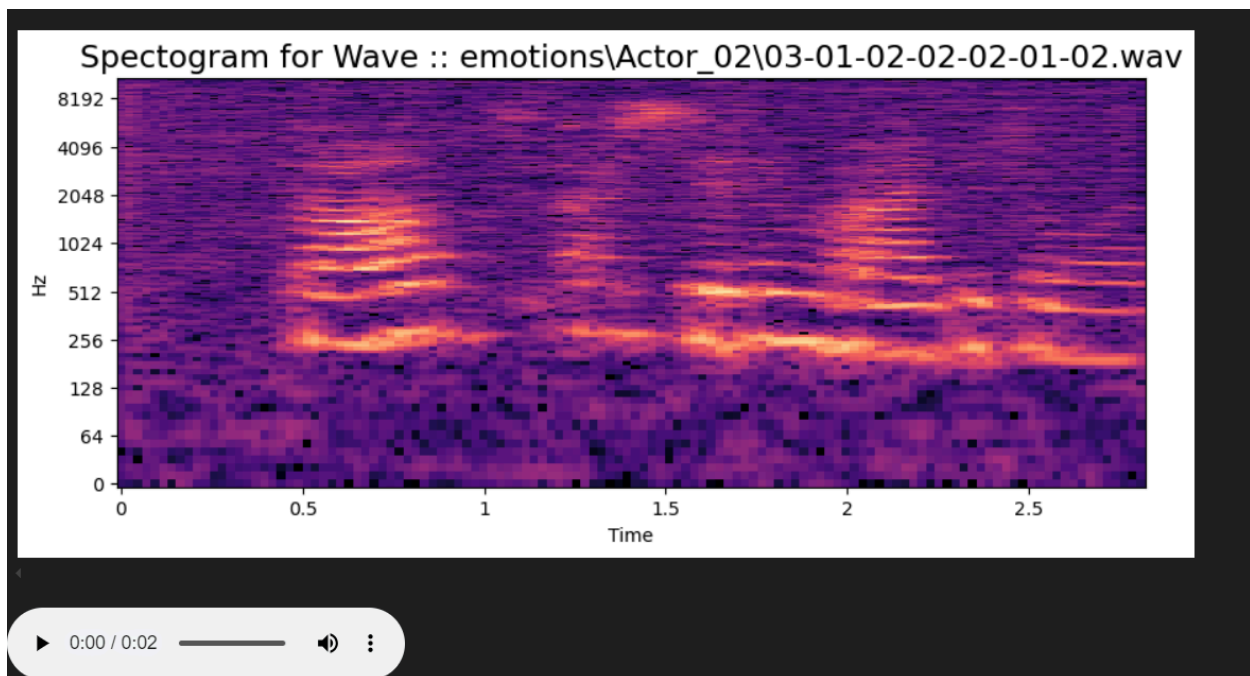
## 8. Visualization

Here are sample visualizations from the dataset:

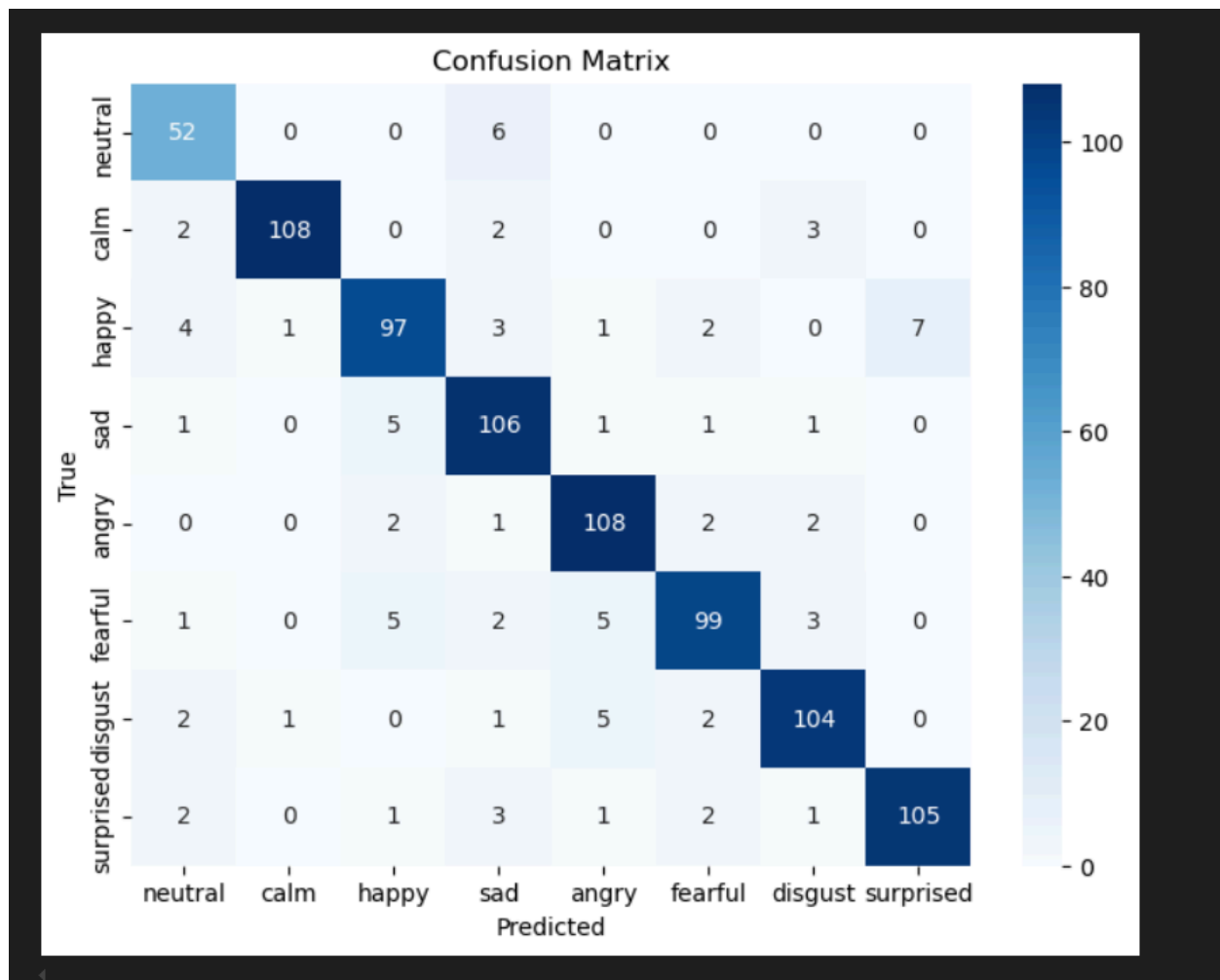
### Waveform



### Spectrogram



Confusion Matrix



## 9. Results

- Test accuracy: **90.27%**
- Confusion matrix reveals strong classification across most classes, with minor confusions mainly between similar emotions.

## 10. Conclusion

This system demonstrates effective audio emotion recognition using MFCC features and an SVM classifier, with augmentation techniques to improve robustness. Future work can include testing deeper models or adding more complex features.

# Appendix: Key Code Sections Explained

## 1. Data Loading

```
python
CopyEdit
def load_data(path):
    f_emotions = []
    f_pathes = []
    folders = os.listdir(path)
    for folder in folders:
        files = os.listdir(path + folder)
        for file in files:
            step = file.split('.')[0]
            step = (step.split('-')[2]) # Emotion label index
            f_emotions.append(int(step))
            f_pathes.append(path + folder + os.sep + file)
    return (f_emotions, f_pathes)
```

- **Purpose:** This function loads the audio dataset.
- It navigates through folders inside the given path.
- For each audio file, it extracts the emotion label from the filename by splitting the string based on delimiters (e.g., ).
- The function returns two lists: one with emotion labels and one with full file paths to the audio files.

## 2. Audio Augmentation Functions

```
python
CopyEdit
def add_noise(data, sr):
    noise = 0.035 * np.random.uniform() * np.amax(data)
    data += noise * np.random.normal(size=data.shape[0])
```

```
return data, sr
```

```
def pitch(data, sr, factor=0.7):  
    pitched = librosa.effects.pitch_shift(y=data, sr=sr, n_steps=factor)  
    return pitched, sr
```

- **add\_noise:** Adds random noise to the audio signal to simulate real-world disturbances and make the model more robust.
- **pitch:** Changes the pitch of the audio clip without changing its speed. This helps augment the data by slightly altering the audio characteristics.

---

### 3. Feature Extraction: MFCC

```
python  
CopyEdit  
def feature_extraction(data, sr):  
    mfcc = librosa.feature.mfcc(y=data, sr=sr)  
    return mfcc
```

- This function extracts **Mel-Frequency Cepstral Coefficients (MFCCs)** from the audio signal.
- MFCCs capture important characteristics of the audio such as tone and timbre, which are useful for distinguishing emotions.
- The extracted MFCCs are used as input features for the machine learning model.

---

### 4. Model Training & Evaluation

```
python  
CopyEdit  
model = SVC(kernel='linear', probability=True)  
model.fit(x_train, y_train)
```

```
y_pred = model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Test Accuracy: {accuracy * 100:.2f}%")
```

- **SVC:** An SVM classifier with a linear kernel is created.
- **Training:** The model learns patterns from the training features ( `x_train` ) and their corresponding labels ( `y_train` ).
- **Prediction:** The model predicts the emotions on the unseen test data ( `x_test` ).
- **Accuracy:** The accuracy metric calculates how many predictions matched the true labels.

# "Comprehensive Analysis and Enhancement of Voice Emotion Recognition System Including User-Driven Model Training"

## Overview

This project implements a **Voice Emotion Recognition** web application using **Streamlit**. The app allows users to record their voice (3 seconds), then extracts audio features and predicts the speaker's emotion using a pre-trained machine learning model.

## Key Components

### 1. Audio Recording

- The app records audio directly from the user's microphone for 3 seconds at a sampling rate of 22050 Hz using the `sounddevice` library.
- The audio is saved temporarily in-memory using `BytesIO` to avoid file system locks.



- The recorded audio is then loaded with `librosa` for further processing.

## 2. Feature Extraction

- The app extracts **Mel-Frequency Cepstral Coefficients (MFCCs)** from the audio signal using `librosa.feature.mfcc`.
- Only feature arrays of specific dimensions ( $20 \times 104$ ) are accepted, matching the shape used during model training.
- The MFCC features are flattened into a 1D array for compatibility with the model.

## 3. Emotion Prediction

- A Support Vector Machine (SVM) classifier, trained externally and saved as `trained_model.pkl`, is loaded using `joblib`.
- The model predicts the emotion label from the extracted MFCC features.
- The app maps the predicted emotion number to a human-readable label such as *happy*, *sad*, *angry*, etc.
- The predicted emotion is displayed alongside a corresponding emoji for better user experience.

## 4. User Interface

- Streamlit provides a simple, interactive UI with a "Record Voice" button.
- After recording, the app plays back the recorded audio and shows the detected emotion.
- A sidebar guides users on how to use the app and lists the supported emotions.

---

## User Training and Customization

- The app architecture allows users to **train the emotion recognition model themselves** if they have access to labeled audio datasets.
- Users can extract MFCC features from their dataset, train an SVM (or any classifier), and save the model as `trained_model.pkl`.

- By replacing the existing model file, the app will use the new model for emotion prediction.
- This flexibility makes the app suitable for research, experimentation, or personalized emotion recognition.


Deploy ⋮

**How to use:**

1. Click the 'Record Voice' button
2. Speak clearly for 3 seconds
3. Wait for the emotion analysis

**Supported Emotions:**

- 😊 Happy
- 😞 Sad
- 😡 Angry
- 😱 Fearful
- 🤢 Disgust
- 😲 Surprised
- 😐 Neutral
- 😌 Calm

 **Voice Emotion Recognition**

Click the button below to record your voice (3 seconds)

🔊 Record Voice

Recording... Speak now!

Recording complete!

▶ 0:03 / 0:03 🔊 ⋮

Analyzing emotion...

Detected Emotion: **SURPRISED**

😲 **SURPRISED**