2024

Speaker Age Prediction using Regression Model

Introduction

The Age prediction system objective is to efficiently predict the age of speakers from audio recordings. This capability is valuable for applications such as personalized voice interfaces, targeted advertising, and forensic analysis. The dataset provided consists of audio recordings with labels indicating the speaker's age, gender, and accent.

System Overview

The system is built using Python language and leverages various libraries to streamline the process of audio data handling, feature extraction, and model development.

1. **Librosa**

Purpose: Specializes in audio signal processing, used here for extracting features such as pitch and intensity from audio recordings.

Benefits: Facilitates efficient and effective audio analysis, crucial for feature extraction.

2. **NumPy**

Purpose: Handles numerical operations, used extensively for matrix operations and data manipulations required in custom regression algorithms.

Benefits: Speeds up data processing with its powerful array operations, enhancing performance.

3. **Pandas**

Purpose: Manages data in DataFrame format, used for data loading, cleaning, and preparation.

Benefits: Streamlines data manipulation tasks, making data handling simpler and more intuitive.

4. **Matplotlib**

Purpose: Provides plotting capabilities, used to visualize data distributions and results.

Benefits: Enables clear and customizable visual representation of complex data insights.

5. **Pickle**

Purpose: Serializes Python objects for persistence, used to save and reload the trained model.

Benefits: Allows efficient reuse of the model without retraining, saving time and resources.

Implementation Details

Step 1: Feature Extraction

def extract_features(file_path, age):

Librosa Library: Used for extracting acoustic features from audio data.

Pitch and formant frequencies provide insights into the anatomical characteristics of the speaker's vocal tract, which are indicative of age. Intensity and duration offer behavioral cues that also correlate with age. These features were extracted using Librosa because it provides robust methods for audio analysis which are well-documented and widely used in the industry for similar tasks.

The features extracted include:

- Pitch (F0): Fundamental frequency, associated with the perceived pitch of the voice.
- Formant Frequencies: Resonances of the vocal tract which characterize vowel sounds.
- Intensity: The energy or loudness of the voice, which can vary with age.
- Duration: Length of speech segments, varying with factors like speaking rate.
- Spectral Features: Derived from the Fourier transform or spectrogram.

```python
# Function to extract features
def extract_features(file_path, age):
    # Load audio file
    audio, sr = librosa.load(file_path, sr=None)
    # Duration
    duration = librosa.get_duration(y=audio, sr=sr)
    # Pitch
    pitches, magnitudes = librosa.piptrack(y=audio, sr=sr)
    pitch = np.median(pitches[pitches > 0])  # Median pitch excluding zero values
    # Energy
    intensity = np.mean(librosa.feature.rms(y=audio))
    # MFCCs
    mfccs = librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=13) #spectral_features
    mfccs_mean = np.mean(mfccs, axis=1)

    # Assuming the first two formant frequencies are required
    # Get Spectrogram
    D = np.abs(librosa.stft(audio))
    # Find peaks in the first (lower) part of the spectrum
    peaks, _ = find_peaks(D[:50].mean(axis=1), height=0)
    formant_frequencies = peaks[:2] * sr / 1024  # Convert bins to Hz

    return [age, duration, pitch, intensity, *formant_frequencies, *mfccs_mean]
```

Step 2: Data Preprocessing

Imputation: Missing values in the dataset were dropped.

Rationale: Proper data preprocessing ensures that the machine learning model trains effectively without bias or skew from the input data format or scale.

```python
# Load the dataset
df = pd.read_csv('truncated_train.csv')

# Drop rows where age, gender, or accent is null
df = df.dropna(subset=['age', 'gender', 'accent'])

# Function to assign a random age based on the category
def assign_random_age(age_group):
    if age_group == 'teens':
        return random.randint(10, 18)
    elif age_group == 'twenties':
        return random.randint(19, 29)
    elif age_group == 'thirties':
        return random.randint(30, 39)
    elif age_group == 'fourties':
        return random.randint(40, 49)
    elif age_group == 'fifties':
        return random.randint(50, 59)
    elif age_group == 'sixties':
        return random.randint(60, 69)
    elif age_group == 'seventies':
        return random.randint(70, 79)
    elif age_group == 'eighties':
        return random.randint(80, 89)
    elif age_group == 'nineties':
        return random.randint(90, 99)
    else:
        return np.nan  # Return NaN if the age_group is not recogni
```

```python
# Apply the function to the 'age' column
df['age'] = df['age'].apply(assign_random_age)

# Save the cleaned data to a new CSV file
df.to_csv('cleaned_truncated_train.csv', index=False)

print("Dataset cleaned and saved to 'cleaned_truncated_train.csv'.")
```

Step 3: Model Training

Techniques Used:

- **Custom Linear Regression Model:** Developed a linear regression model using the normal equation for learning the relationship between features and age.

Rationale: The linear regression algorithm was chosen for its simplicity and interpretability, which is crucial for understanding feature influences on predictions. Given the project constraints against using built-in models, a custom implementation was developed to provide hands-on control over the training process and adapt it specifically to our needs.

```
def custom_linear_regression(X, y):
    # Add a column of ones to X for the intercept term
    X = np.c_[np.ones((X.shape[0], 1)), X]

    # Calculate the coefficients using the normal equation
    coefficients = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(y)

    # Return the coefficients
    return coefficients

def predict(X, coefficients):
    # Add a column of ones to X for the intercept term
    X = np.c_[np.ones((X.shape[0], 1)), X]

    # Return the predictions
    return X.dot(coefficients)
```

Step 4: Model Evaluation

Techniques Used:

- MSE and MAE: Evaluated the model's accuracy using Mean Squared Error and Mean Absolute Error.
- R^2 Coefficient: Used to measure how well the variations in age are explained by the model.

Rationale: MSE and MAE provide direct metrics to quantify the error in predictions, while R^2 provides an indication of the quality of the model fit. Evaluating the model across different age groups helps understand its effectiveness and potential biases.

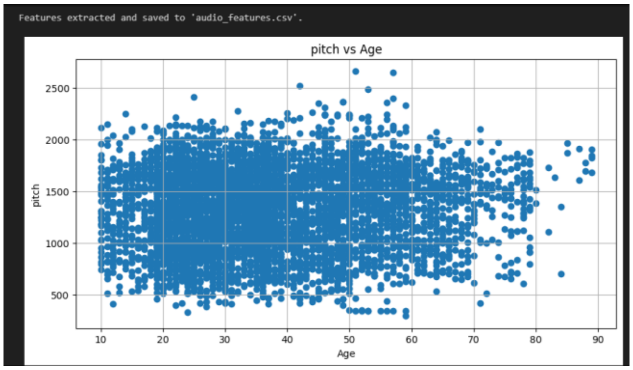
```
# Calculate MAE, and R²
mae = mean_absolute_error(features_df['age'], features_df['predicted_age'])
r2 = r2_score(features_df['age'], features_df['predicted_age'])

print(f"Mean Absolute Error (MAE): {mae}")
print(f"R-squared (R²): {r2}")

✓ 0.0s

Mean Absolute Error (MAE): 12.673216764537418
R-squared (R²): 0.036696709342453339
```

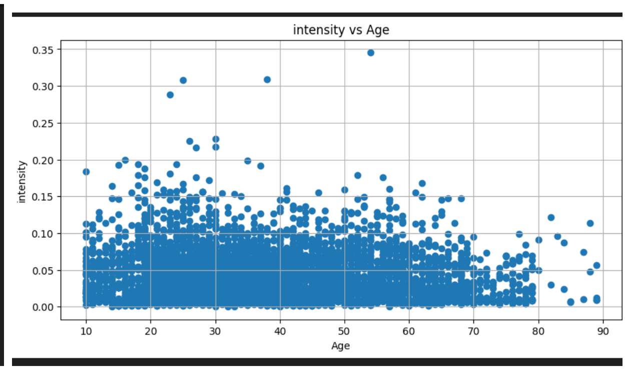
Conclusion:

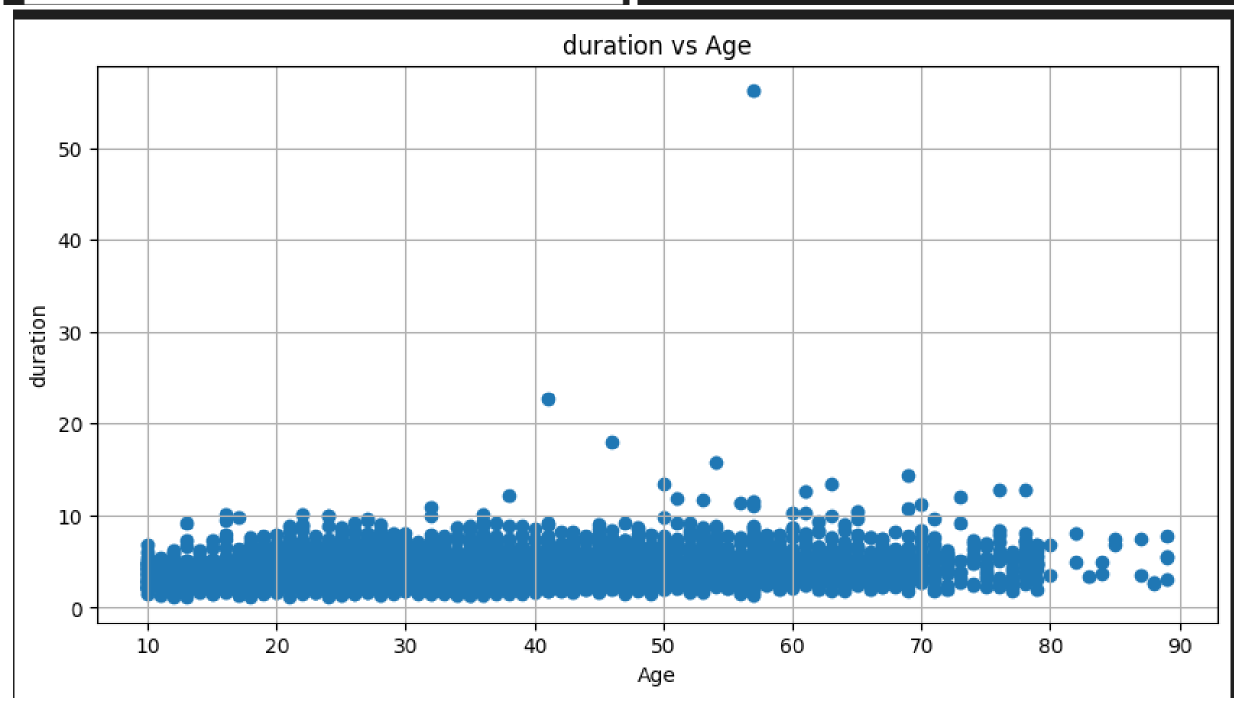
This project demonstrates the feasibility of predicting speaker age from audio recordings using extracted features and a custom linear regression model. The evaluation shows promising results but also highlights areas for future improvement, such as model complexity and data diversity.

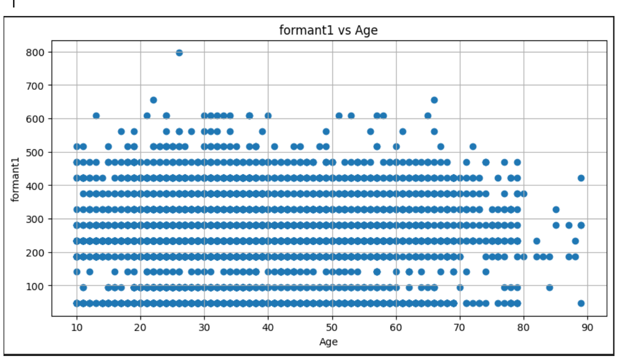
Screenshots of the output:

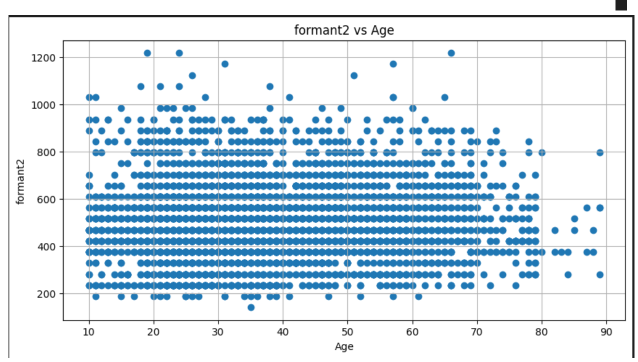
1. Feature Extraction:











2. Model Training:

	filename	pitch	intensity	duration	formant1	formant2	predicted_age	age
0	cv-valid-train/sample-000005.mp3	706.46216	0.006549	5.832	375.000	796.875	40.851103	20
1	cv-valid-train/sample-000008.mp3	1478.27920	0.013888	1.728	421.875	609.375	36.932751	71
2	cv-valid-train/sample-000013.mp3	1714.86770	0.035668	4.224	375.000	750.000	40.449537	32
3	cv-valid-train/sample-000014.mp3	532.87090	0.031007	5.376	46.875	328.125	36.954784	64
4	cv-valid-train/sample-000019.mp3	692.22130	0.043531	3.720	187.500	421.875	35.376621	51
...	...	...	...	...	...	...	...	...
4817	cv-valid-train/sample-014993.mp3	743.70200	0.013427	7.464	281.250	609.375	42.289732	43
4818	cv-valid-train/sample-014994.mp3	1426.64950	0.022652	6.696	328.125	609.375	43.007962	39
4819	cv-valid-train/sample-014995.mp3	1657.86320	0.072386	3.024	609.375	890.625	38.513974	33
4820	cv-valid-train/sample-014998.mp3	764.28010	0.069570	4.704	234.375	468.750	36.339851	46
4821	cv-valid-train/sample-015000.mp3	1305.58740	0.073601	3.864	234.375	421.875	36.482812	19

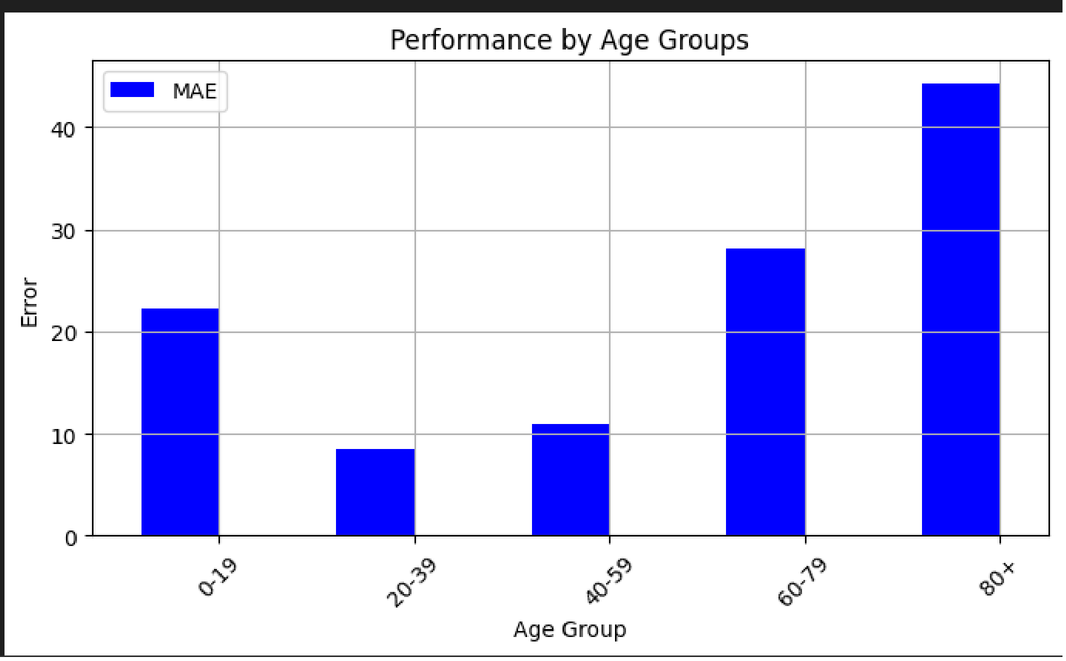
4822 rows × 8 columns

```
Accuracy of Training Model: 63.83%
```

3. Training Model Evaluation

```
...  Mean Absolute Error (MAE): 12.673216764537418
     R-squared (R²): 0.03669670934245339
```

```
age_group
0-19     22.235891
20-39     8.592534
40-59    10.909296
60-79    28.079186
80+      44.264724
dtype: float64
```



4. Testing Model

	filename	pitch	intensity	duration	formant1	formant2	predicted_age	age
0	cv-valid-test/sample-000003.mp3	706.46216	0.006549	5.832	375.000	796.875	41.178539	24
1	cv-valid-test/sample-000005.mp3	1478.27920	0.013888	1.728	421.875	609.375	36.993732	73
2	cv-valid-test/sample-000008.mp3	1714.86770	0.035668	4.224	375.000	750.000	40.743800	32
3	cv-valid-test/sample-000009.mp3	532.87090	0.031007	5.376	46.875	328.125	36.826229	61
4	cv-valid-test/sample-000014.mp3	692.22130	0.043531	3.720	187.500	421.875	35.356824	59
...	...	...	...	...	...	...	...	...
1322	cv-valid-test/sample-003971.mp3	336.43503	0.020357	1.776	234.375	421.875	32.594688	28
1323	cv-valid-test/sample-003975.mp3	799.96630	0.027563	4.560	234.375	421.875	37.357313	78
1324	cv-valid-test/sample-003976.mp3	1692.40480	0.009603	2.400	187.500	328.125	36.998604	23
1325	cv-valid-test/sample-003980.mp3	1093.68680	0.061802	5.496	234.375	468.750	38.559205	42
1326	cv-valid-test/sample-003989.mp3	1147.12320	0.009859	2.784	375.000	703.125	37.866560	53

1327 rows × 8 columns

```
Accuracy of Testing Model: 62.40%
```

5. Testing Model Evaluation → Accuracy Function

```
... Mean Absolute Error (MAE): 12.595109108239173
    R-squared (R²): 0.048573168941934310
```

```
MAE by Age Group:
age_group
0-19     22.207632
20-39     8.648322
40-59    10.525210
60-79    28.015921
80+      44.383153
dtype: float64
```