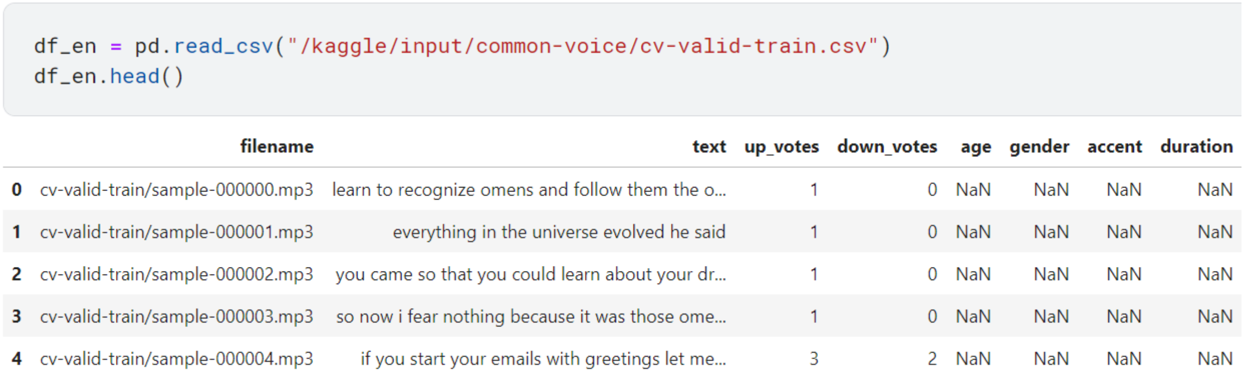
Spoken Language Identification (SLID)

Goal: Given an English sentence spoken by the user, the program will be able to identify the gender and potentially the accent of the person speaking.

Common Voice

Data preprocessing



```
df_en = pd.read_csv("/kaggle/input/common-voice/cv-valid-train.csv")
df_en.head()
```

	filename	text	up_votes	down_votes	age	gender	accent	duration
0	cv-valid-train/sample-000000.mp3	learn to recognize omens and follow them the o...	1	0	NaN	NaN	NaN	NaN
1	cv-valid-train/sample-000001.mp3	everything in the universe evolved he said	1	0	NaN	NaN	NaN	NaN
2	cv-valid-train/sample-000002.mp3	you came so that you could learn about your dr...	1	0	NaN	NaN	NaN	NaN
3	cv-valid-train/sample-000003.mp3	so now i fear nothing because it was those ome...	1	0	NaN	NaN	NaN	NaN
4	cv-valid-train/sample-000004.mp3	if you start your emails with greetings let me...	3	2	NaN	NaN	NaN	NaN

For the training dataset, after removing all the data that did not contain a label for gender, we are left with 74059 samples. The associated mp3 files are in another folder. We also dropped the up_votes, down_votes and age columns as they are not useful for the remainder of the project.

Machine Learning Model

We used a subset of Pytorch called Torchaudio for most of the data preprocessing (particularly to turn the audio files into MFCC spectrograms and to do the feature extraction). The Convoluted Neural Network is also implemented using Pytorch. So far, we only have a basic model of a CNN that will then be modified to ameliorate its performance. It takes as input a RGB image and has two convolutional layers. The first layer applies 5x5 filters to the input images and outputs 6 channels. Then it goes through a pooling layer which serves to reduce the dimensions of the feature map by 2 to decrease the computational burden. The second layer applies 5x5 filters to the output of the first layer then outputs 16 channels which go through a pooling layer again. The output of the second pooling layer is flattened and passed through three fully connected layers with 120, 84, and 10 output units, respectively. It looks something like this:

input image -> conv1 -> pool -> conv2 -> pool -> fc1 -> fc2 -> fc3 -> output

Each filter produces a feature map that highlights certain aspects of the MFCC spectrogram we fed it.
The model hasn't been trained on the dataset yet as we are still working on a way to optimize transforming each audio file into MFCC spectrograms.