**Speech to Text**

**Libraries Used**

For installing the speech\_recognition library: pip install SpeechRecognition.

For installing pyaudio library: pip install pyaudio (Supports microphone inputs).

**What is Energy Threshold**

recognizer.energy\_threshold

We humans are able to distinguish speech from noise naturally, for a computer program they are just audio levels. It needs to know which levels should be considered speech (which it needs to process in order to recognize what's being said), and which levels should be considered silence or background noise. So, libraries like the SpeechRecognition has an energy threshold set which defines what audio level and above should be considered speech.

We should adjust the energy threshold to properly distinguish the speech from noise.

Value is set to 300 by default. Under 'ideal' conditions (such as in a quiet room), values between 0 and 100 are considered silent or ambient, and values 300 to about 3500 are considered speech.

We can use “**adjust\_for\_ambient\_noise**” function. Program will listen for some time to adjust the value: r.adjust\_for\_ambient\_noise(source, duration=5)

Since ambience levels of our environment are keep changing. “adjust\_for\_ambient\_noise” function is not a dynamic function.

So, we use **“dynamic\_energy\_threshold”** function:

r = sr.Recognizer()

r.energy\_threshold = 4000

r.dynamic\_energy\_threshold = True

**The best way to step up energy level** is: first using “**adjust\_for\_ambient\_noise**” then, we use “**adjust\_for\_ambient\_noise**”.

r.adjust\_for\_ambient\_noise(source, duration=5)

r.dynamic\_energy\_threshold = True

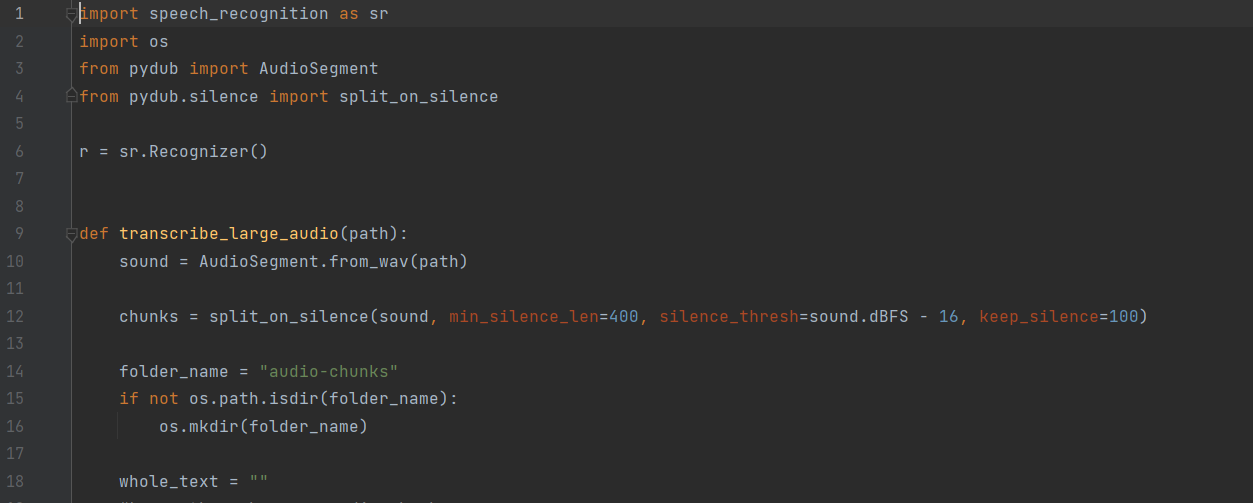
**Split on Silence**

Function splits the audio according to silence in it.

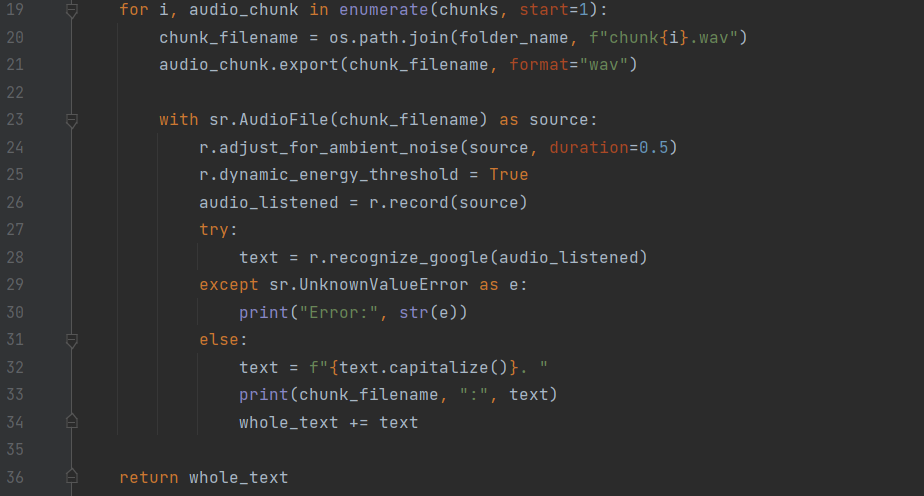
Function takes some parameters:

* Sound: Our .wav audio.
* Min\_silence\_len: The minimum length for silent sections is in milliseconds. By default it is 1000ms. If it is greater than audio length, it returns empty list. Ideal value is 500ms.
* Silence\_tresh: Upper bound for how quiet is silent in dBFS. By default it is -16. Ideal value is around -40.
* Keep\_silence: Leaves some silence at the beginning and end of the chunks. Keeps the sound from sounding like it is abruptly cut off. When the length of the silence is less than the keep\_silence duration it is split evenly between the preceding and following non-silent segments. If True is specified, all the silence is kept, if False none is kept. default: 100ms.
* Seek\_step: Step size for interating over the segment in ms.

**Explanation of The Function**



* Function takes path as parameter value.
* **Line-6:** We created a recognizer object.
* **Line-12:** Pydub library splits the audio to the chunks. The chunks are divided by minimum silence length, silence threshold, keep silence and seek step.
* **Line-15:** Controls if a folder existence with specific name.
* **Line-16:** Used to create a directory. If file exist (we control this with if statement in line 15), it gaves us an error at line 16. Thats why we use if statement.



**Line-19:** Loops through every audio chunk.

**Line-20:** Export chunks and save it into a folder.

**Line-24:** This function listens through the first 0.5 seconds of the chunk. Then, it decides on a energy threshold value based on that.

**Line-25:** This function dynamically changes the previous energy threshold value.

**Line-28:** Converts chunks to text via API.

**Line-36:** Returns text for all of the chunks.

As an audio, finishing lines of the movie called “American Psycho” is taken.

Result of the function has been written to a text file named “result.txt”.

**References**

Energy threshold: <https://www.codesofinterest.com/2017/04/energy-threshold-calibration-in-speech-recognition.html>

Split on silence: https://www.codespeedy.com/split-audio-files-using-silence-detection-in-python/