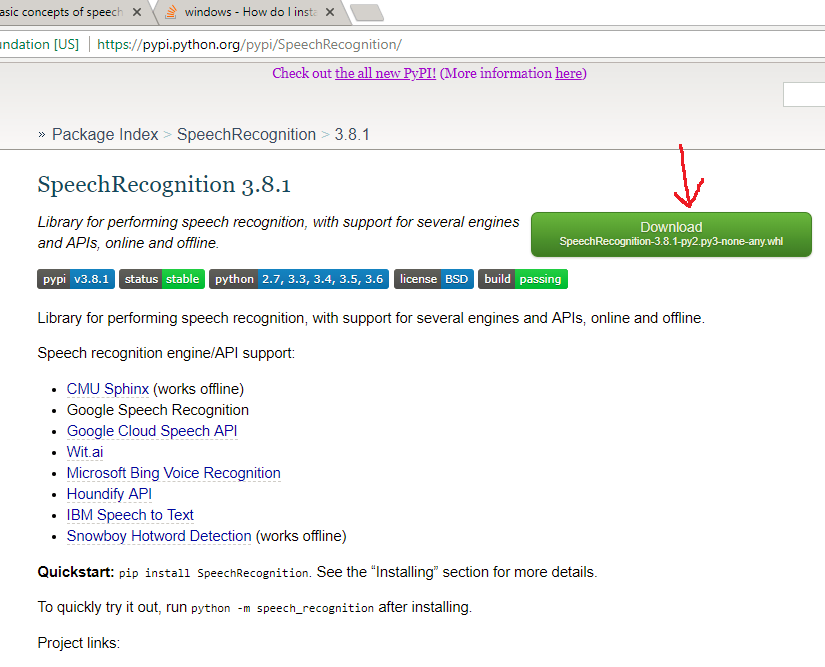
There are two toolkits that got my attention, one is HTK toolkit, and the other is CMU Sphinx.

Now, I started looking in the tutorial of HTK, and the whole thing is huge. So it might require a lot of time. In the meantime, Googled Python speech recognition module, got me to the site

<https://pypi.python.org/pypi/SpeechRecognition/>

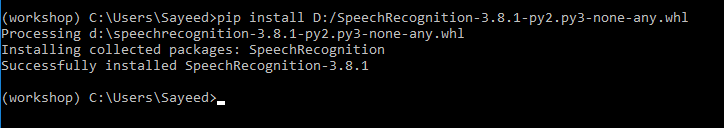
so from there, downloaded the speechrecognition toolkit, a file named

SpeechRecognition-3.8.1-py2.py3-none-any.whl

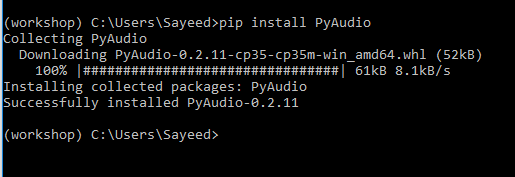


After downloading, went to the windows command line, activated my anaconda workshop environment, and installed the module.

* activate workshop
* pip install D:/ SpeechRecognition-3.8.1-py2.py3-none-any.whl

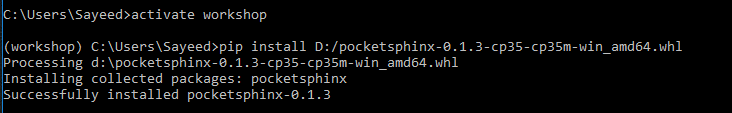


* pip install PyAudio



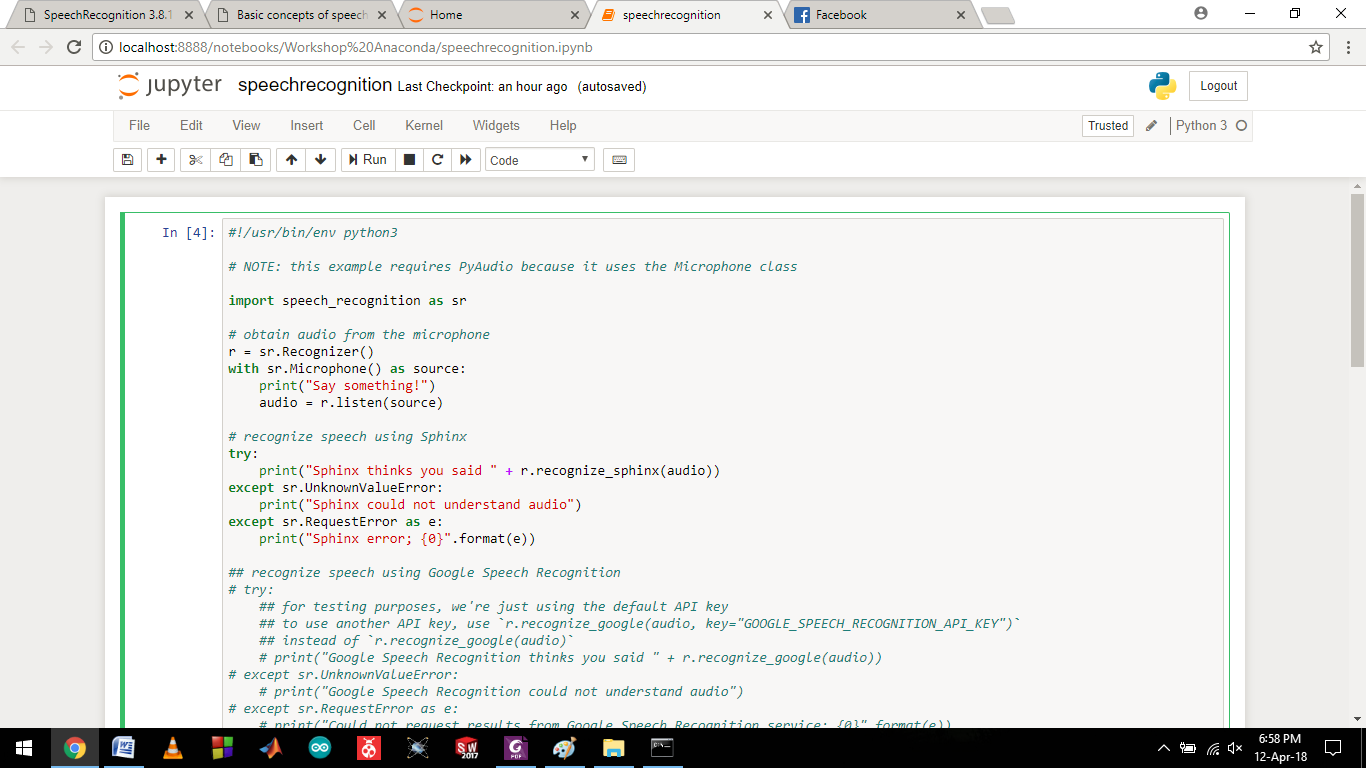
Download the file

<https://github.com/Uberi/speech_recognition/blob/master/third-party/pocketsphinx-0.1.3-cp35-cp35m-win_amd64.whl>



<https://github.com/Uberi/speech_recognition/blob/master/examples/microphone_recognition.py>

ran this example while commenting out the other engines and APIs except cmusphinx



Running the module, you have to wait for a while when the program is looking for speech, and then it generates results.



I actually said, ‘Hello speech recognition say something’

But this is something to work on, so I think we have a stepping stone in our progression now.

CMUSphinx has a full documentation and tutorial, although we are using PocketSphinx, I think the tutorials are worth looking into.

<https://cmusphinx.github.io/wiki/>

Maybe we should look into the dictionary used by sphinx, and make oun dictionary with necessary training data.

Interfacting with matlab

The *system()* function in matlab can directly execute system command and return result. So we can use this to call the python interpreter during runtime of any matlab script.

Example:

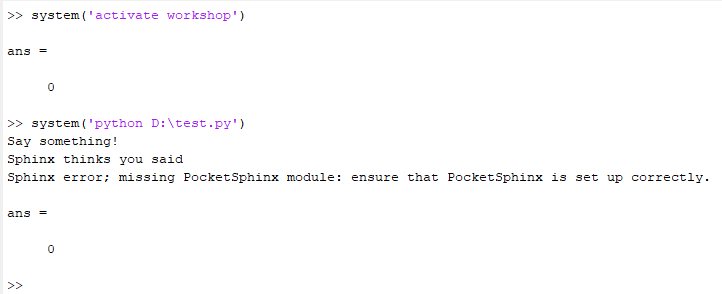
* system(‘python somecode.py’);

This works almost the same as running python from windows command line.

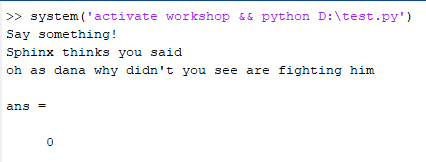
But in the previous steps we have set up the entire speech recognition module inside the environment ‘workshop’. So somehow either we have to set up the environment inside matlab and run our voice processing code inside that environment, or we have to set up the speech recognition module inside the base environment of python.

For some reason, I could not install the pocketsphinx library inside the base environment.

And when used in two different lines, the environment setup doesn’t hold, for example:



So we just have to pipe the two commands so that the python script can work inside the environment.



Clarifying, again, it isn’t working, I said “Hello testing one two three four five” and this is what it thinks.

So adjusting the speech recognition part is later, and to make things work, meaning that for using whatever text that the toolkit is showing us, we somehow have to read those lines from the matlab command window and feed it into our script for further processing. The specific code that ran in the previous line is as follows:

D:\test.py

import speech\_recognition as sr

# obtain audio from the microphone

r = sr.Recognizer()

with sr.Microphone() as source:

print("Say something!")

audio = r.listen(source)

# recognize speech using Sphinx

try:

print("Sphinx thinks you said ")

print(r.recognize\_sphinx(audio))

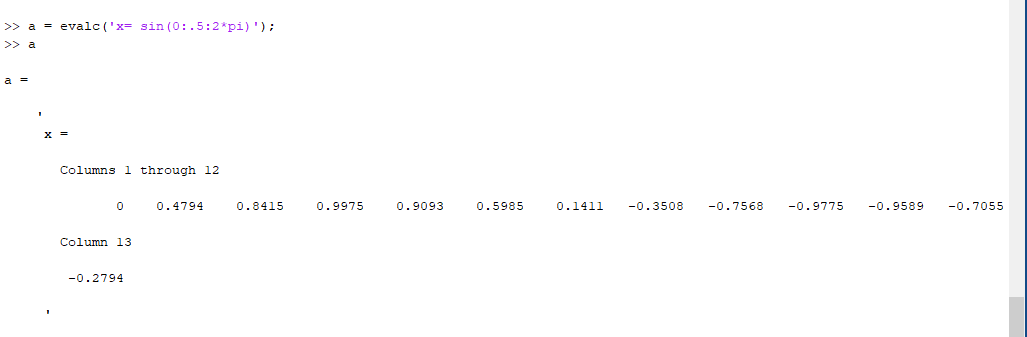
except sr.UnknownValueError:

print("Sphinx could not understand audio")

except sr.RequestError as e:

print("Sphinx error; {0}".format(e))

Found a function that can nest commands inside it and after execution of that command, saves whaever output is shown in the command window as a string, which can be used later for other processes.



See how the supposed output of the command x = sin(0:.5:2\*pi) was not shown in execution, but the whole thing has been saved as a string inside a. We can use this process to convert whatever output the python program shows in the command window and extract our required voice data from that string.

clear all;

close all;

clc;

a=evalc('system(''activate workshop && python D:\test.py'')');

a=compose(a);

a=char(a);

C = strsplit(a,'\n');

disp(C(1));

now C(1) contains voice processed string output.

Google speech API

pip install google-api-python-client

Add the following part to our previous test.py code to add google voice recognition api.

# recognize speech using Google Speech Recognition

try:

# for testing purposes, we're just using the default API key

# to use another API key, use `r.recognize\_google(audio, key="GOOGLE\_SPEECH\_RECOGNITION\_API\_KEY")`

# instead of `r.recognize\_google(audio)`

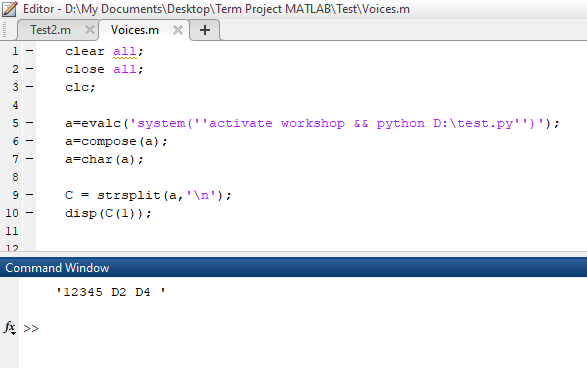
print("Google Speech Recognition thinks you said " + r.recognize\_google(audio))

except sr.UnknownValueError:

print("Google Speech Recognition could not understand audio")

except sr.RequestError as e:

print("Could not request results from Google Speech Recognition service; {0}".format(e))



I said “Hello Testing 1 2 3 4 5 D2 D4”

For some reason Hello Testing went missing

But Google Speech API works far better than pocketsphinx