Part 2 Software Familiarization

From online search, we found that hmmlearn also implements the Hidden Markov Models. We learned how to use it and the differences between our implements.

1. How to use

First, build the model with using

model = hmm.MultinomialHMM(n\_components=n\_states)

which n\_components means how many states in each timestamp.

Then we could set the start probability, the transfer matrix and emission probability with using below code.

model.startprob\_= start\_probability

model.transmat\_= transition\_probability

model.emissionprob\_= emission\_probability

Since we have built the model, we could use it to get the most possible hidden layer given the observation series.

Model.predict(observationlist)

1. Differences
2. Our model only available with the discrete observation data.

But in that library, it includes GaussianHMM and GMMHMM which could be used for continuous observation data.

1. To the HMMs, there are three fundamental problems:

* Given the model parameters and observed data, estimate the optimal sequence of hidden states.
* Given the model parameters and observed data, calculate the likelihood of the data.
* Given just the observed data, estimate the model parameters.

Our model only resolved the first problems, but the library resolved all these

problems.

1. Our model could not save and load hmm model, however the library could do it with function dump and load. I think we could implement this function in the future.
2. Improvements
3. Extend our code so that it could support more functions like training the continuous observation data and saving the trained model.