Title

Hello, my name is Janggun Lee. I am working with Taeyoung Kim and Geon Park in this group project, and here to present our Project pitch.

Slide 1 : Table of Contents

In this presentation, first I will shortly introduce the Gaussian Mixture Model, which is a key part of the model we wish to use in this project, Mixture of Experts. After introducing the models, I will talk about how we will conduct our experiments, and the benefits of using probabilistic programming in our project.

Slide 2 : GMM-1

I will begin by first introducing GMM.
Gaussian Mixture Model, or GMM, is a well-known generative model, where it assumes that the data is obtained from a weighted sum of multivariate Normal distributions. Now, while the model itself is reasonable and simple, the learning step is not as easy.

Slide 3 : GMM-2

Learning is done by an Expectation Maximization algorithm, as shown in the slide. The E step computes the probability that the ith input belongs to the kth cluster, the M step updates the parameters accordingly, and these steps are repeated until convergence. For GMM, the equations for EM is nice, but for other models it may be much more complicated.

Slide 4 : GMM-3

This image shows the result of a GMM. The center of the clusters are the mean, and the shapes correspond to the covariance matrix.

Slide 5 : GMM-4

The following Anglican code is an implementation of GMM, taken from the Anglican homepage.
One interesting fact is the hyperparameters and parameters are chosen to be conjugate priors. Conjugate priors are pairs prior, posterior distributions that transform into each other by Bayes' Rule. The ones given here are pi, which is sampled from the Dirichlet distribution, and the indicator variable z, which is sampled from the categorical distribution using pi. When we have such conjugate pair, computations are much simpler and results can be expected to be 'nice'.

Slide 6 : MoE-1

The purpose of introducing GMM is to use in Mixture of Experts or MoE, which works as follows: After classifying the input into clusters by mixture models such as GMM, each cluster is further trained by so called expert models. As the mixture model divides the original input, it is called the gating model. Examples of gating models are GMM and Neural Networks, and examples of expert models are Linear classifiers, Neural Networks, etc. An example of a MoE we have already seen in class is the piecewise linear function in lecture 1. The gating model is deciding the intervals to approximate linearly, and the experts are the linear pieces.

Slide 7 : MoE-2

A possible benefit of using MoE is application to inverse problems. If f is not invertible , inverse problems are difficult to learn with a single function, as we are then trying to arroximate something that is not a function using a function. Using MoE, f can be split into parts that are one-to-one, and experts used on each parts. An example is shown in the left, where a GMM splits the data set, representing f into three parts, and each part is approximated linearly, representing the experts.

Slide 8 : MoE-3

A key point of MoE is that a MoE can also be an expert, so we can have a hierarchical structure for MoE. A model where we can see this type of structure are decision trees, where every internal node is a classification function, which is essentially a gating model, and another decision tree.

Slide 9 : Experiment

In terms of the actual experiment, the model we want to use in this project is this hierarchical MoE, where the gating models are taken as GMMs. Experts are not decided yet as they will differ by our choice of dataset. We wish to tackle inverse problems using our model, with the dataset most likely from image processing.
The figure describes how our model will work in action. Data is separated by GMM generator, and for each cluster it is decided by the expanding decider to either go into another round of the GMM generator, or be modeled by the expert generator.

Slide 10 : Advantage-1

I will now describe some advantages in using a probabilistic programming for our project.
If you have derived the EM algorithm of GMM by hand, you will know that it is complex due to matrix derivatives and inverses. At least for GMM the result was simple, but for MoE the EM algorithm is very complex and the answer may not be achievable in a compact form. As probabilistic programming does automatic inference to find parameters, EM is not needed at all. It is still prone to local minima, which we are not yet sure how to avoid.

Slide 11 : Advantage-2

Hyperparameters are usually hard to tune in many ML models. In the case of GMM, the number of clusters K needs to be tuned by running the model multiple times with various K. This is time consuming and can also be hard to automate. In probabilistic programming, it is possible to introduce hyperparameters as distributions, which will help simplify the tuning. The diagram represents the structure of variables for GMM with this idea, where the primed variables represent the latent variables for the hyperparameters.

Slide 12 : Advantage-3

A major incentive for using a probabilistic programming for MoE is that the splitting condition can also be included in inference. So unlike other models such as CHAID and CART where we need to fully expand the tree then prune, the size of tree can be trained simultaneously.
The benefits so far are ways how probabilistic programming can simplify the process of machine learning. The last benefit is how the probabilistic programming simplifies our model's structure compared to a mathematical way.
As Hierarchical MoE has structure of random tree, it is difficult to express mathematically in a compact form, so hard to implement in more classical languages. Anglican has the query structure which can express such form. compactly.

Slide 13: Finale

These are the references for outside material in our presentation.
This is the end of our presentation. If you have any questions, doesn't hesitate to ask.