# Mixture Density Networks

**K M Arefeen Sultan**
u1419693
Kahlert School of Computing
University of Utah
Salt Lake City, UT 84112
arefeen.sultan@utah.edu

**Md Rakibul Haque**
u1527423
Kahlert School of Computing
University of Utah
Salt Lake City, UT 84112
rakibul.haque@utah.edu

Github Link: https://github.com/RakibulHaqueSajal/MixtureDensityNetwork

## 1 Dataset

In order to analyze the performance of Mixture Density Networks (MDNs) in modeling multi-modal distributions, we generate two synthetic datasets: a spiral dataset and a sine curve dataset. We also use housing dataset as our high dimensional dataset. These datasets provide varied challenges in capturing complex conditional distributions and evaluating probabilistic predictions.

### 1.1 Spiral Dataset

The spiral dataset is a classic example of a non-linear data distribution. It consists of points sampled from a spiral structure with added Gaussian noise. The x and y coordinates are computed as:

$$x = r\cos(\theta) + \mathcal{N}(0, \text{noise}) \tag{1}$$

$$y = r\sin(\theta) + \mathcal{N}(0, \text{noise}) \tag{2}$$

### 1.2 Sine Curve Dataset

The sine curve dataset provides a simpler yet multi-valued function mapping where the output is conditioned on a single input variable. We generates a 2D dataset based on a sine function.

The y-values are computed as:

$$y = \sin(x) + \mathcal{N}(0, \text{noise}) \tag{3}$$

### 1.3 Housing Dataset

The housing dataset consists of multiple attributes describing residential properties, including price, area, bedrooms, bathrooms, stories, and various amenities. For the regression task, we begin by loading the housing dataset and converting categorical variables into numerical values using label encoding. The dataset is then split into features and target values, where the first column represents the target variable (house price) and the remaining columns serve as predictive features. To enhance model performance, the features are normalized using StandardScaler to maintain consistency and prevent scale disparities. The dataset is subsequently split into training and testing sets using an 80-20 ratio to ensure a fair evaluation of model performance.

## 2 Methodology

### 2.1 Mathematical Formulation of Mixture Density Network (MDN) formulation

We have:

$$p(y \mid x) = \sum_{i=1}^{K} \pi_i(x) \mathcal{N}(y \mid \mu_i(x), \Sigma_i(x))$$

The lower-triangular matrix $L_i(x)$ is obtained from a parameter vector $v_i(x)$ to ensure positive-definiteness of the covariance matrix:

$$\Sigma_i(x) = L_i(x) L_i(x)^T$$

where $L_i(x)$ is computed from $v_i(x)$ using:

$$L_{ij} = \begin{cases} v_{ij}, & j \leq i \\ 0, & j > i \end{cases}$$

and the diagonal elements are constrained to be positive using the softplus function:

$$L_{ii} = \text{softplus}(v_{ii}) + \epsilon$$

where $\epsilon$ is a small positive constant to ensure numerical stability.

Setting K = 1 will give us the desired single gaussian regression model.

#### 2.1.1 Loss Formulation

The loss function for the Mixture Density Network (MDN) is defined as the negative log-likelihood of the mixture of Gaussians:

$$\mathcal{L} = -\frac{1}{N} \sum_{n=1}^{N} \log \sum_{i=1}^{K} \pi_i(x_n) \mathcal{N}(y_n \mid \mu_i(x_n), \Sigma_i(x_n))$$

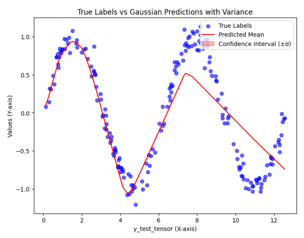
where: - $N$ is the batch size. - $K$ is the number of Gaussian components. - The log-sum-exp trick is used to ensure numerical stability:

$$\mathcal{L} = -\frac{1}{N} \sum_{n=1}^{N} \log \sum_{i=1}^{K} \exp\left(\log \pi_i(x_n) + \log \mathcal{N}(y_n \mid \mu_i(x_n), \Sigma_i(x_n))\right).$$

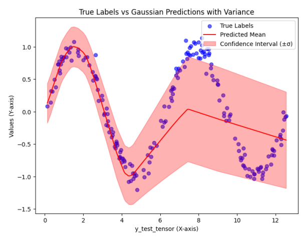
## 3 Result Analysis

### 3.1 Sine Curve

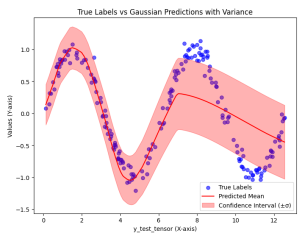
The three figures depict the performance of different models on the sine curve dataset, showcasing their predictions and uncertainty estimations. The Deterministic Neural Network (Figure a) predicts a single mean value without capturing uncertainty, resulting in a rigid fit that struggles with regions exhibiting high variance. The Single Gaussian Regression (Figure b) improves upon this by modeling both the mean and variance, providing a confidence interval ($\pm\sigma$). Finally, the Mixture Density Network (MDN) with 10 Gaussian components, shown in Figure c shows great performance due to it's low variance meaning the model is confident than single regression model. The confidence intervals adaptively vary across regions, highlighting its ability to better represent uncertainty compared to the deterministic and single Gaussian approaches.



(a) Deterministic Neural Network



(b) Single Gaussian Regression

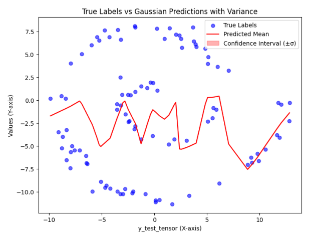


(c) MDN (no of gaussians=10)

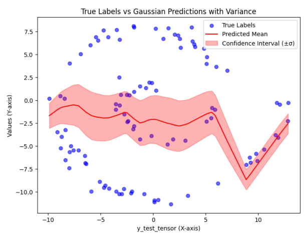
Figure 1: Models trained and tested on Sine Curve.

### 3.2 Spiral Curve

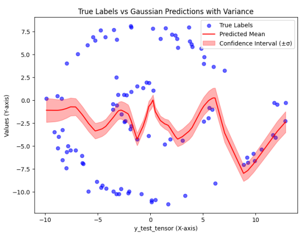
The three figures demonstrate the performance of different models on the spiral dataset, showcasing their ability to predict and capture uncertainty in a highly non-linear task. The Deterministic Neural Network (Figure a) fails to capture the complexity of the spiral, producing a highly erratic and inaccurate fit due to its inability to model uncertainty or adapt to multi-modal distributions. The Single Gaussian Regression (Figure b) provides a smoother fit with confidence intervals ($\pm\sigma$), but it struggles to fully adapt to the spiral's non-linear shape, leading to over-smoothing and suboptimal uncertainty representation in regions of high complexity. The Mixture Density Network (MDN) with 10 Gaussian components (Figure c) demonstrates poor fit with high confidence.



(a) Deterministic Neural Network



(b) Single Gaussian Regression



(c) MDN (no of gaussians=10)

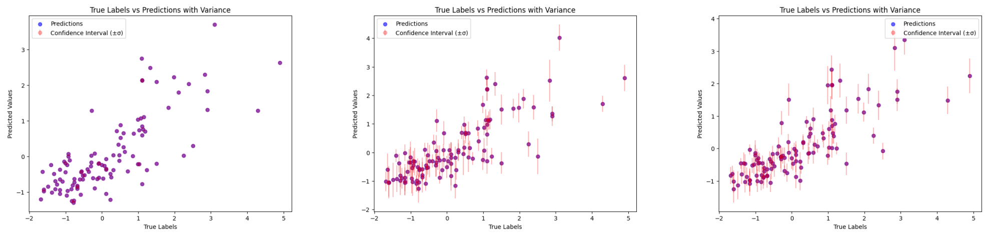
Figure 2: Models trained and tested on Spiral Curve.

### 3.3 Housing Dataset

The three figures compare the performance of different models on the housing dataset by plotting true labels against predicted values along with confidence intervals. The Deterministic Neural Network (Figure a) predicts values directly but lacks any uncertainty quantification, leading to a rigid and potentially misleading representation of predictions, particularly in areas with high variability. The Single Gaussian Regression (Figure b) introduces uncertainty quantification by modeling predictions with variance, providing confidence intervals ($\pm\sigma$). However, its ability to capture uncertainty is limited, and the confidence intervals are often inconsistent, failing to reflect the true variance in complex regions. The Mixture Density Network (MDN) with 10 Gaussian components (Figure c) demonstrates good performance by accurately modeling both the predictions and the uncertainty, with confidence intervals that vary appropriately based on the complexity and variability of the data.

The Table 1 compares the performance of three models—Deterministic Neural Network (NN), Single Gaussian Regression, and Mixture Density Network (MDN)—on three datasets (Sine Curve, Spiral, and Housing) using Root Mean Squared Error (RMSE) as the evaluation metric. Overall, the results emphasize the MDN's utility in datasets requiring uncertainty modeling, while simpler models like Deterministic NN can perform well for less complex tasks.

The Table 2 highlights the effect of hyperparameter tuning on the number of Gaussian components in the Mixture Density Network (MDN) for the housing dataset, evaluated using Train and Test RMSE. As a critical hyperparameter, the number of Gaussians determines the model's flexibility to capture



(a) Deterministic Neural Network (b) Single Gaussian Regression (c) MDN (no of gaussians=10)

Figure 3: Models trained and tested on Housing Dataset.

Table 1: Results of all the models on Test Set

| **Test RMSE** | | | |
|---|---|---|---|
| **Dataset** | **Deterministic NN** | **Single Gaussian Regression** | **MDN** |
| **Sine Curve** | 0.3337 | 0.4799 | 0.3942 |
| **Spiral** | 6.3717 | 6.0347 | 6.1815 |
| **Housing** | 0.8248 | 0.7959 | 0.7894 |

complex data distributions. For 10 Gaussians, the model achieves a low Train RMSE (0.3114) but a relatively high Test RMSE (0.7894), indicating underfitting due to insufficient complexity. Increasing the number of Gaussians improves generalization initially, with the Test RMSE dropping to 0.7522 at 40 Gaussians, the optimal value where the model balances bias and variance effectively. This analysis demonstrates that hyperparameter tuning, particularly selecting the optimal number of Gaussians, is crucial to achieving the best trade-off between model complexity and generalization performance.

Table 2: Results of trying different number of Gaussians for MDN on housing dataset.

| **No of Gaussians** | **Train RMSE** | **Test RMSE** |
|---|---|---|
| 10 | 0.3114 | 0.7894 |
| 20 | .3273 | .8131 |
| 30 | .3222 | .7877 |
| 40 | .3552 | .7522 |
| 50 | .3170 | .7643 |

Figure 4 shows the training loss of MDN on housing dataset. From the loss we can derive that the model is learning.

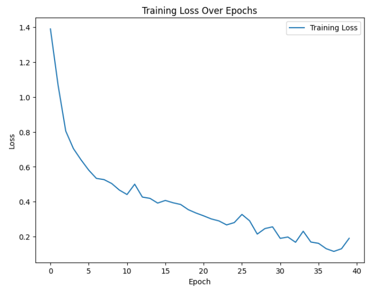


Figure 4: Training Loss of MDN on Housing Dataset