SMAI Assignment 2 Report

Tags SMAI

Name : Meet Gera

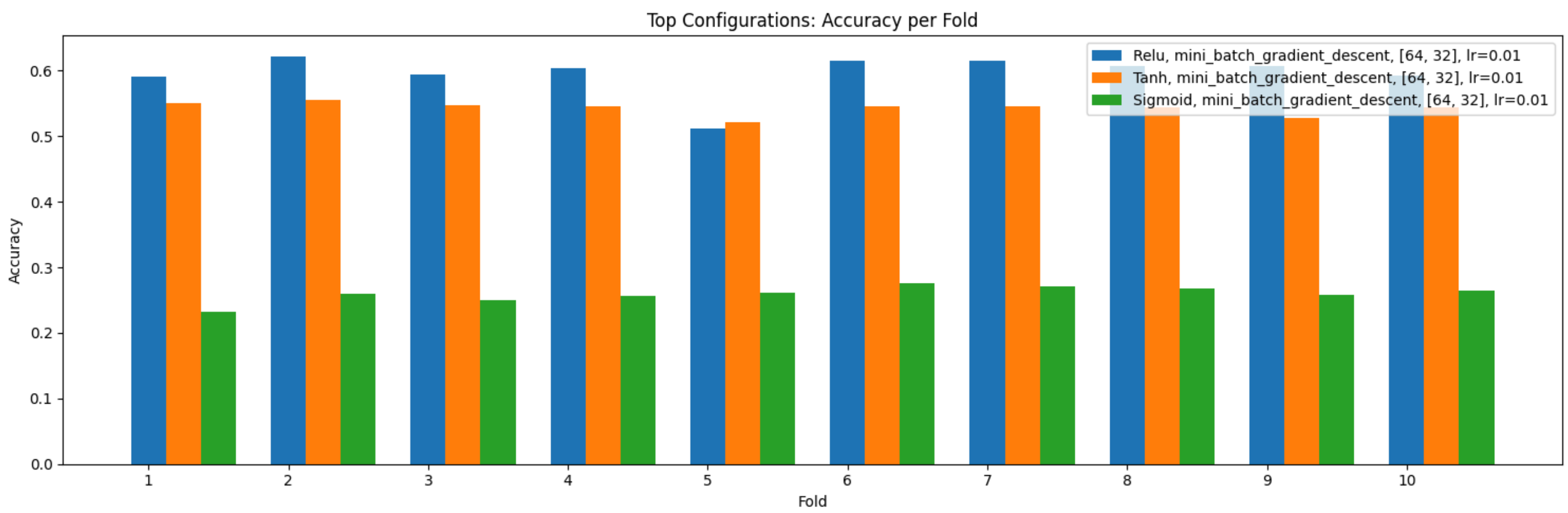
Roll Number : 2022102039

Question 2.1 : MLP Multi Class Classifier

2. Mean and Std of Evaluation Metrics across all folds to find the best set of hyperparams

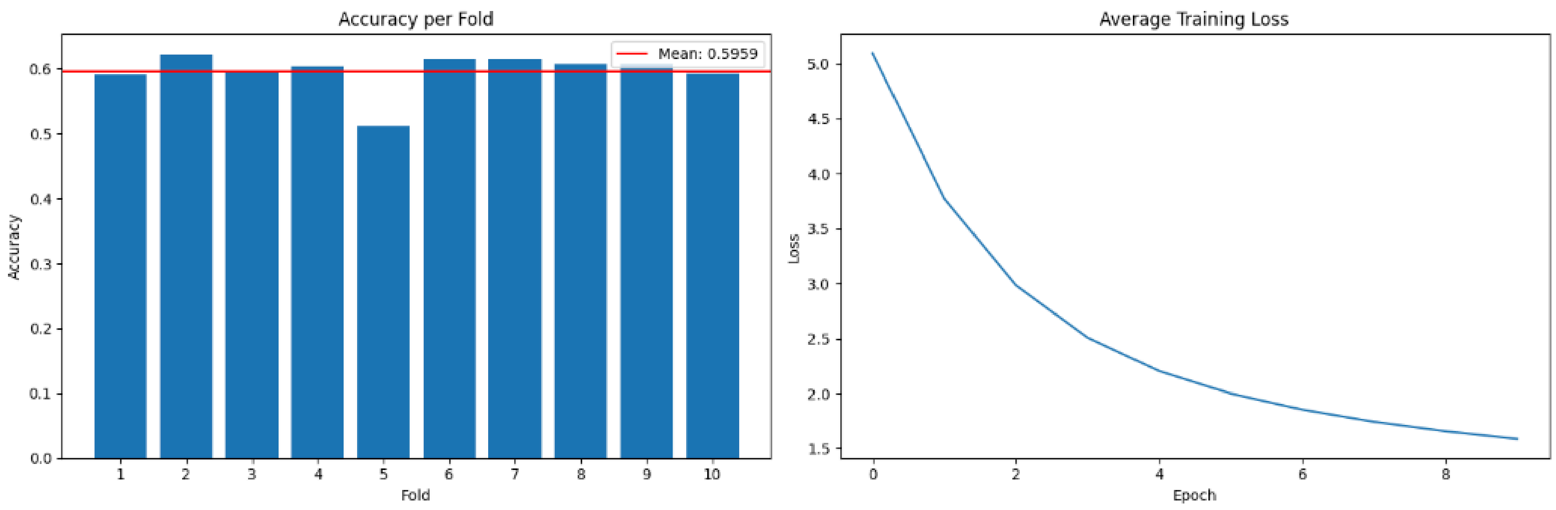
📊 RESULTS SUMMARY (Ordered by Mean Accuracy) [TOP 4 TAKEN ONLY]

Rank	Hidden Layers	Activation	Optimizer	Learning Rate	Batch Size	Mean Accuracy ± Std Dev
1	[64, 32]	relu	Mini-batch Gradient Descent	0.01	32	**0.5959 ± 0.0299**
2	[64, 32]	tanh	Mini-batch Gradient Descent	0.01	32	0.5425 ± 0.0096
3	[64, 32]	sigmoid	Mini-batch Gradient Descent	0.01	32	0.2597 ± 0.0117
4	[64, 32]	relu	Batch Gradient Descent	0.01	Full batch	0.0054 ± 0.0036

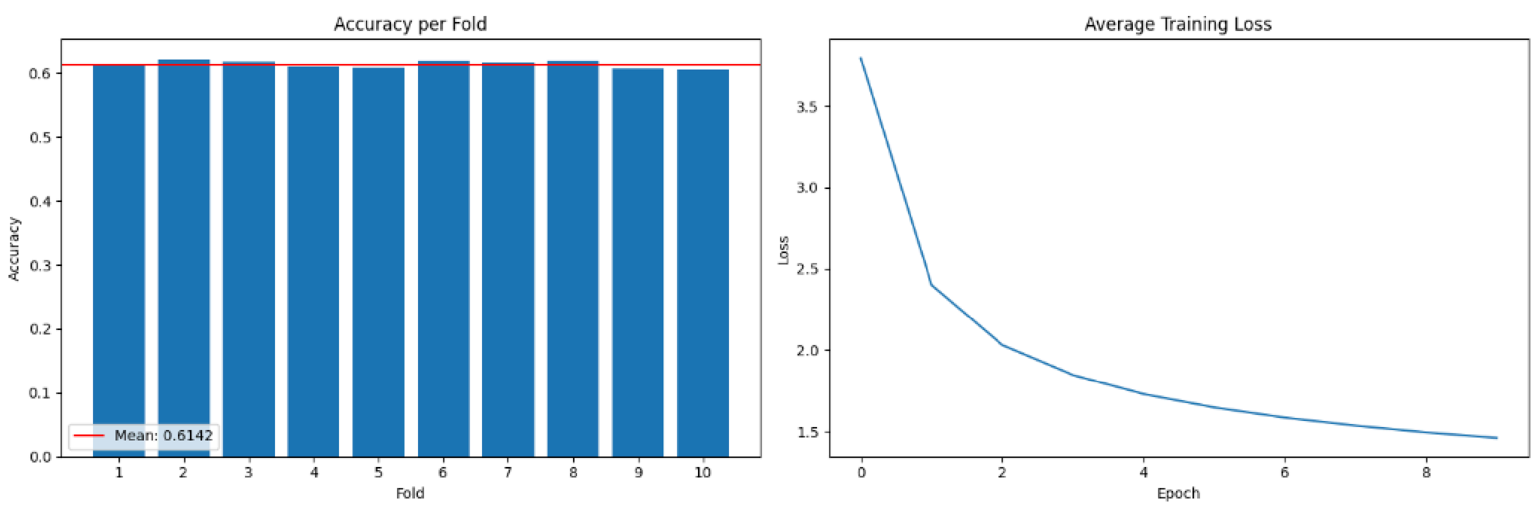


Best Configuration : Relu, mini_batch_gradient_descent, layers=[64, 32], lr=0.01
Mean accuracy: 0.5959±0.0299

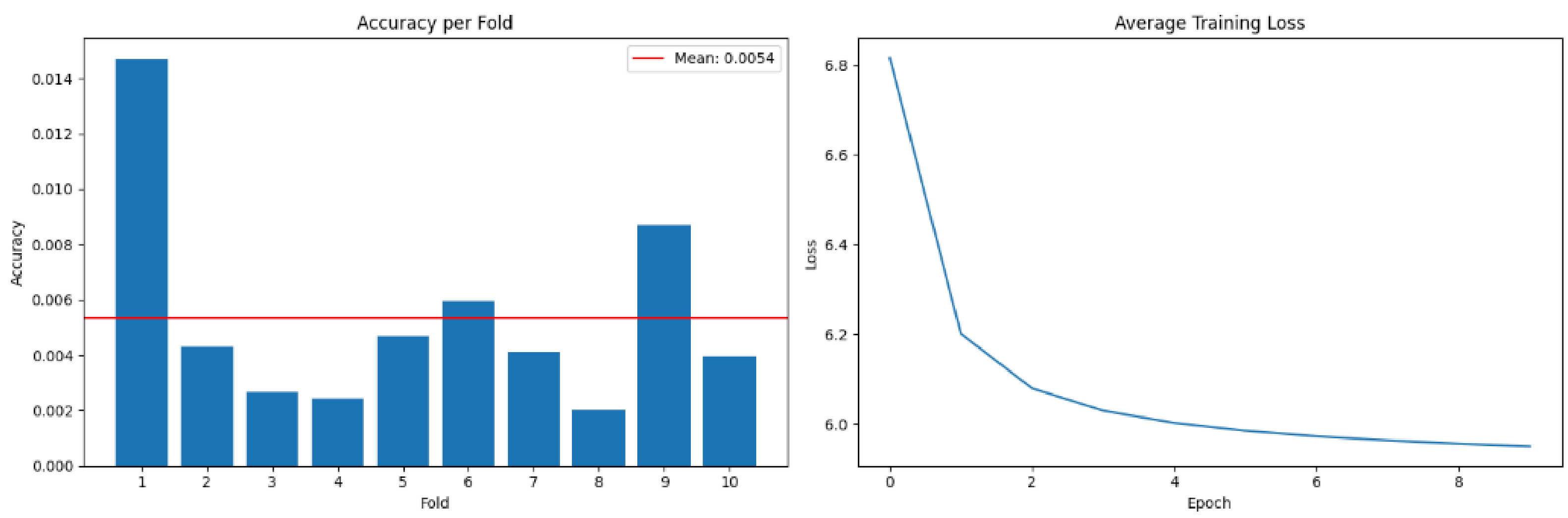
```
================================================================================
Running experiment with:
Hidden Layers: [64, 32]
Activation: relu
Optimizer: mini_batch_gradient_descent
Learning Rate: 0.01
Batch Size: 32
================================================================================

Fold 1: Accuracy = 59.14%
Fold 2: Accuracy = 62.21%
Fold 3: Accuracy = 59.40%
Fold 4: Accuracy = 60.39%
Fold 5: Accuracy = 51.10%
Fold 6: Accuracy = 61.45%
Fold 7: Accuracy = 61.49%
Fold 8: Accuracy = 60.76%
Fold 9: Accuracy = 60.69%
Fold 10: Accuracy = 59.26%

<Figure size 1500x500 with 0 Axes>
```




```
================================================================================
Running experiment with:
Hidden Layers: [64, 32]
Activation: sigmoid
Optimizer: sgd
Learning Rate: 0.01
Batch Size: 1
================================================================================

Fold 1: Accuracy = 61.38%
Fold 2: Accuracy = 62.22%
Fold 3: Accuracy = 61.91%
Fold 4: Accuracy = 60.96%
Fold 5: Accuracy = 60.87%
Fold 6: Accuracy = 62.07%
Fold 7: Accuracy = 61.61%
Fold 8: Accuracy = 61.98%
Fold 9: Accuracy = 60.64%
Fold 10: Accuracy = 60.58%

<Figure size 1500x500 with 0 Axes>
```



```
================================================================================
Running experiment with:
Hidden Layers: [64, 32]
Activation: relu
Optimizer: batch_gradient_descent
Learning Rate: 0.01
Batch Size: Full batch
================================================================================

Fold 1: Accuracy = 1.47%
Fold 2: Accuracy = 0.43%
Fold 3: Accuracy = 0.27%
Fold 4: Accuracy = 0.24%
Fold 5: Accuracy = 0.47%
Fold 6: Accuracy = 0.59%
Fold 7: Accuracy = 0.41%
Fold 8: Accuracy = 0.20%
Fold 9: Accuracy = 0.87%
Fold 10: Accuracy = 0.40%

<Figure size 1500x500 with 0 Axes>
```



OTHERS IN IPYNB FILE

1. What do the mean and standard deviation tell you about model performance and consistency?

- The **mean** indicates the model's **average performance** across runs.
- The **standard deviation** reflects **consistency**—lower values mean more stable and predictable results.

2. How does a high vs. low standard deviation impact confidence in the model's generalization?

- A **low standard deviation** increases confidence that the model will perform similarly on unseen data.
- A **high standard deviation** suggests variability and less reliable generalization.

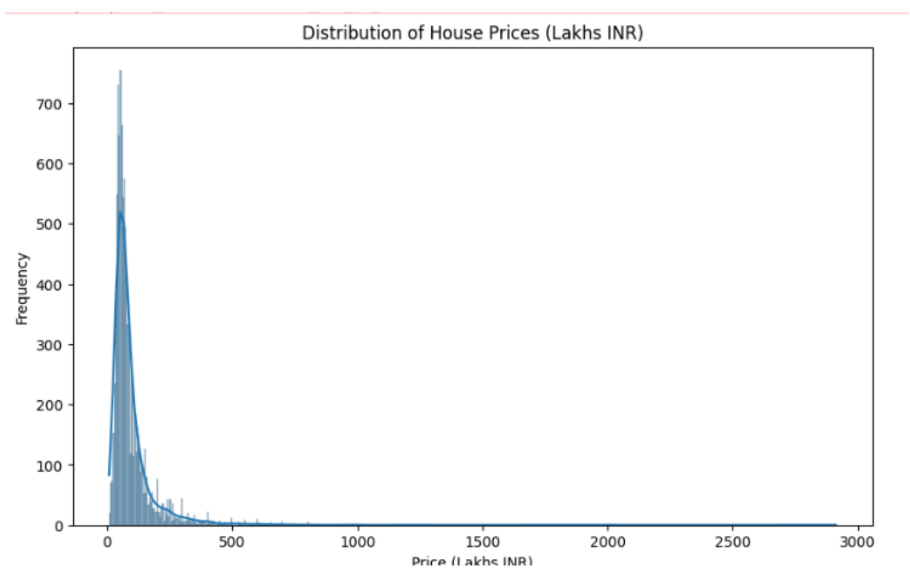
3. If one configuration has a slightly higher mean accuracy but a significantly higher standard deviation compared to another with marginally lower mean accuracy, which would you choose and why?

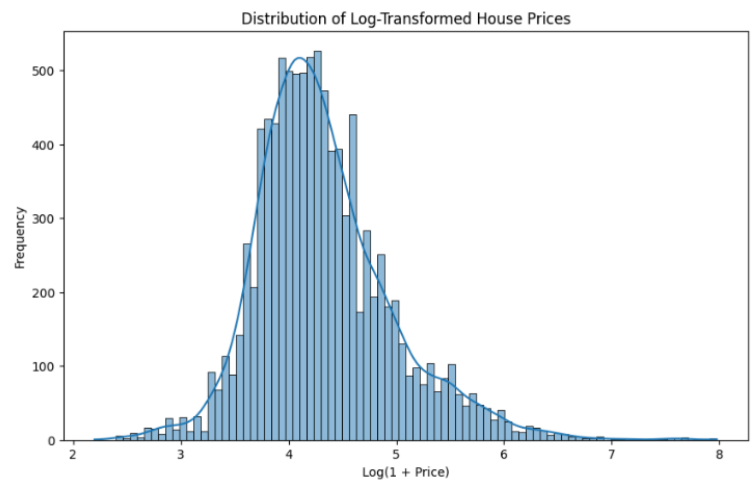
- I would choose the configuration with **lower standard deviation** and **slightly lower mean accuracy**, as it's **more consistent and reliable**, which is crucial for generalization.

Question 2.2 : Banglore House Price Prediction

1. Dataset

```
--- Descriptive Statistics (Cleaned Data) ---
              mean         std   min      max
total_sqft  1492.700714  892.158845  11.0  30400.0
bath           2.489306    0.974454   1.0     10.0
balcony        1.604237    0.785983   0.0      3.0
price         96.130970  112.523074   8.0   2912.0
bhk            2.622327    0.967114   1.0     12.0
--------------------------------------------
```





2. Different Activation Functions and Optimisers

📊 Full Results Table

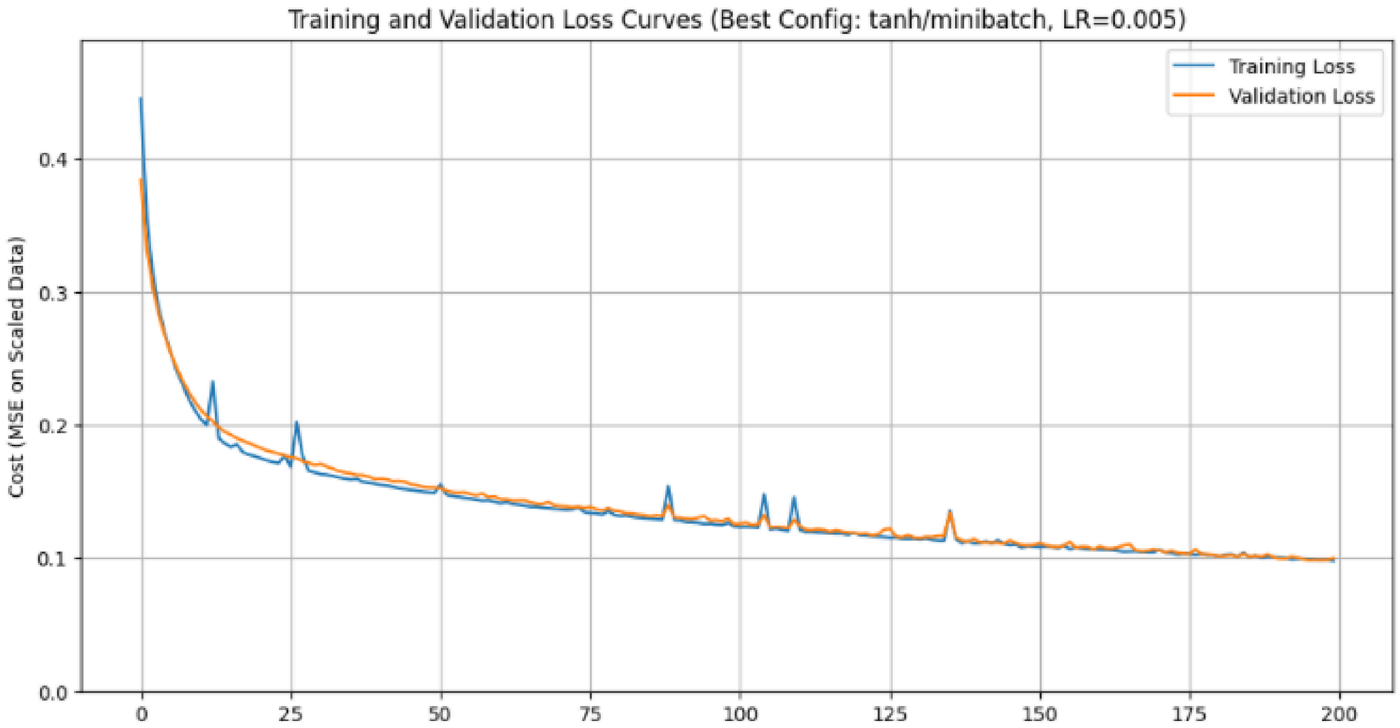
#	LR	Epochs	Activation	Optimizer	Architecture	R^2	RMSE	MSE	Training Time
0	0.005	200	tanh	minibatch	[193, 64, 32, 1]	0.771575	58.2585	3393.16	22.17s
1	0.005	200	relu	minibatch	[193, 64, 32, 1]	0.768904	58.6170	3435.96	16.94s
2	0.005	200	tanh	minibatch	[193, 64, 32, 1]	0.766594	59.0268	3484.38	39.19s
3	0.005	200	relu	minibatch	[193, 64, 32, 1]	0.764232	59.1807	3502.39	38.79s
4	0.005	200	relu	minibatch	[193, 64, 32, 1]	0.759372	59.7860	3574.44	14.46s
5	0.005	200	relu	minibatch	[193, 64, 32, 1]	0.758583	59.8509	3582.14	17.88s
6	0.010	200	relu	minibatch	[193, 64, 32, 1]	0.756653	60.1229	3614.36	28.20s
7	0.010	200	tanh	minibatch	[193, 64, 32, 1]	0.755328	60.1521	3621.10	23.31s
8	0.010	300	relu	minibatch	[193, 64, 32, 1]	0.754223	60.5291	3663.77	41.85s
9	0.010	300	relu	minibatch	[193, 64, 32, 1]	0.750207	60.6781	3680.48	53.91s
10	0.010	300	relu	minibatch	[193, 64, 32, 1]	0.745902	61.3334	3758.90	59.29s
11	0.010	300	tanh	minibatch	[193, 128, 64, 1]	0.738769	62.7476	3934.73	40.24s
12	0.010	300	tanh	minibatch	[193, 128, 64, 1]	0.728449	62.2944	3880.47	34.47s
13	0.010	300	tanh	minibatch	[193, 128, 64, 1]	0.721157	64.0245	4096.16	48.27s
14	0.010	300	tanh	minibatch	[193, 128, 64, 1]	0.711162	64.3153	4136.46	43.39s
15	0.010	300	relu	batch	[193, 128, 64, 1]	0.714652	64.3153	4136.46	43.39s
16	0.010	200	relu	batch	[193, 64, 32, 1]	0.692937	64.3596	4124.22	59.53s
17	0.010	200	relu	batch	[193, 64, 32, 1]	0.679338	65.1987	4252.15	22.19s
18	0.010	200	relu	batch	[193, 64, 32, 1]	0.631190	67.5813	4567.26	11.45s
19	0.010	200	relu	batch	[193, 64, 32, 1]	0.625802	69.1184	4777.36	14.65s
20	0.010	200	relu	batch	[193, 64, 32, 1]	0.565382	74.0189	5478.23	7.39s
21	0.010	200	tanh	batch	[193, 128, 64, 1]	0.561921	74.5679	5566.18	22.29s
22	0.010	200	tanh	batch	[193, 128, 64, 1]	0.558944	80.5772	6486.38	12.23s
23	0.010	200	tanh	batch	[193, 64, 32, 1]	0.507896	82.7979	6856.99	10.96s
24	0.010	200	tanh	batch	[193, 64, 32, 1]	0.490844	85.3283	7280.26	14.93s
25	0.010	200	tanh	batch	[193, 64, 32, 1]	0.480865	85.4886	7318.06	15.43s
26	0.010	200	tanh	batch	[193, 64, 32, 1]	0.447978	86.6523	7500.80	20.53s
27	0.010	200	tanh	batch	[193, 64, 32, 1]	0.439632	90.4426	8180.83	15.30s
28	0.010	200	tanh	batch	[193, 64, 32, 1]	0.436216	90.5244	8195.07	8.91s
29	0.005	200	tanh	batch	[193, 64, 32, 1]	0.406120	98.9371	9791.45	8.04s
30	0.005	200	tanh	batch	[193, 128, 64, 1]	0.382116	91.9325	8441.92	15.41s
31	0.005	200	tanh	batch	[193, 64, 32, 1]	0.344532	93.6784	8763.72	10.87s

🥇 Best Configuration Summary:

- **Learning Rate**: 0.005
- **Epochs**: 200
- **Activation**: tanh
- **Optimizer**: minibatch
- **Architecture**: [193, 64, 32, 1]
- **Test R^2**: 0.771575
- **Test RMSE**: 58.2585
- **Test MSE**: 3393.16

3. Training Curves

```
--- Plotting Training Curves for Best Configuration ---
```



Question 2.3 : Multi label News Article Classification

Metrics :

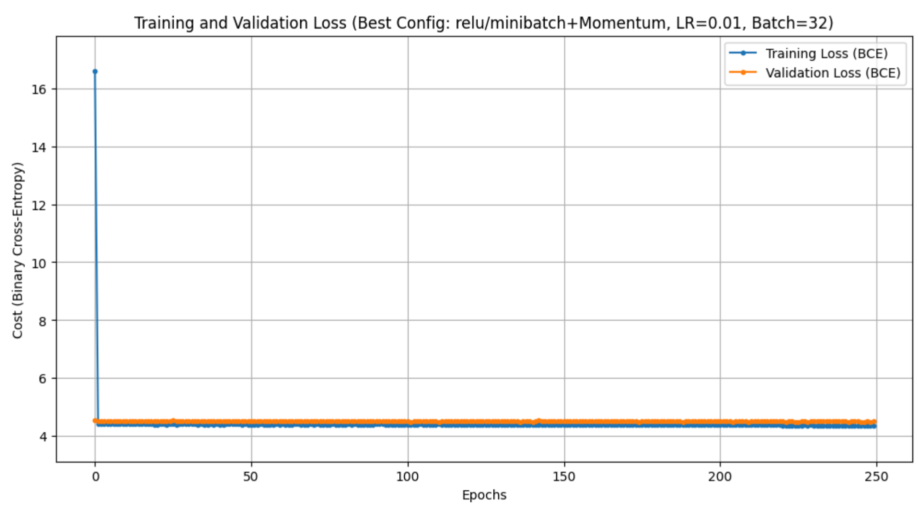
Experiment Results

🏆 Top 5 Configurations (Sorted by F1 Micro Score): Total 48 Combinations tried with 100 to 150 epochs

Rank	Learning Rate	Epochs	Activation	Optimizer	Batch Size	Architecture	F1 Micro	Accuracy	Hamming Loss	F1 Samples
1	0.010	250	relu	minibatch+Momentum	32	[5000, 256, 128, 90]	**0.0160**	0.001	0.0138	0.001
2	0.005	250	relu	minibatch+Momentum	64	[5000, 128, 90]	**0.0009**	0.0006	0.0138	0.0006
3	0.005	250	tanh	minibatch+Momentum	64	[5000, 128, 90]	**0.0008**	0.0005	0.0138	0.0005
4	0.005	250	relu	minibatch+Momentum	32	[5000, 256, 128, 90]	**0.0007**	0.0004	0.0138	0.0004
5	0.005	250	relu	minibatch+Momentum	64	[5000, 256, 128, 90]	**0.0006**	0.0003	0.0138	0.000

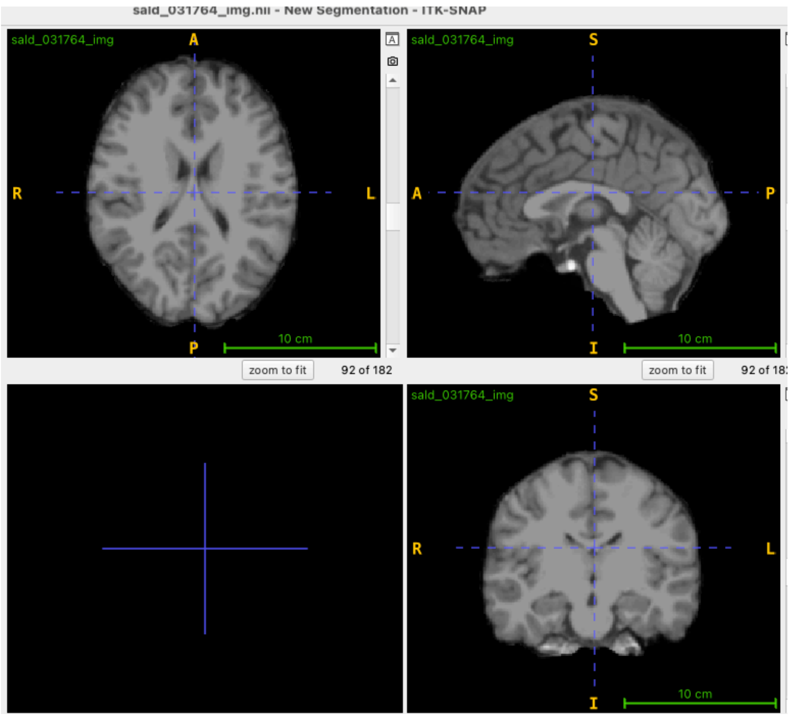
Full Results Overview

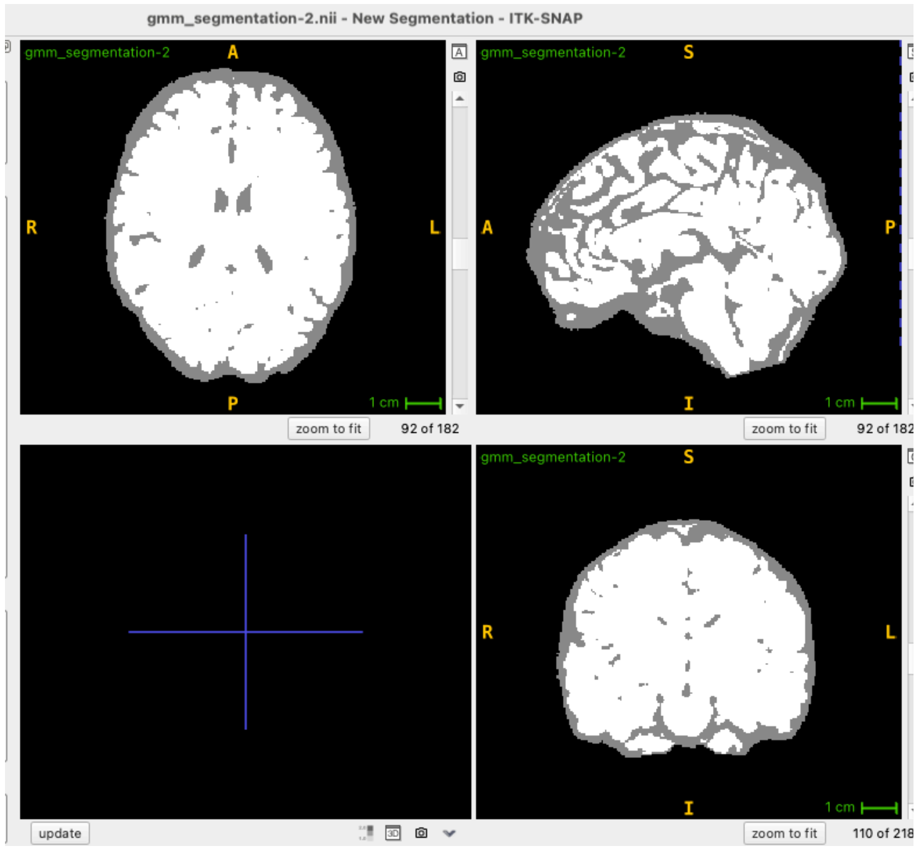
- **Total Configurations Tested**: 48
- **Most Frequent Architectures**: [5000, 128, 90] and [5000, 256, 128, 90]
- **Common Epochs**: 150 or 250
- **Dominant Optimizer**: Mini-batch + Momentum
- **Activations Used**: ReLU, Tanh
- **Best Score Observed (F1 Micro)**: **0.0160**



Question 3 : GMM

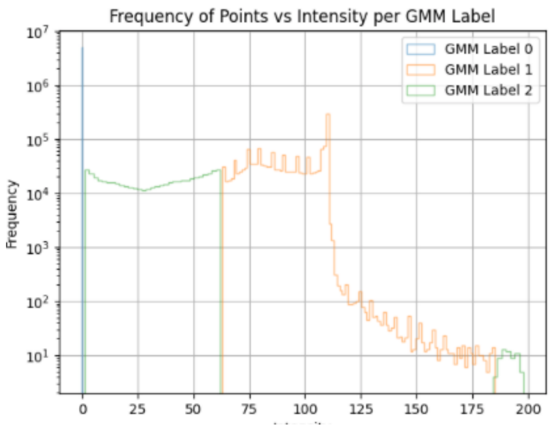
1. Visualise Original and segmented fMRI scan.

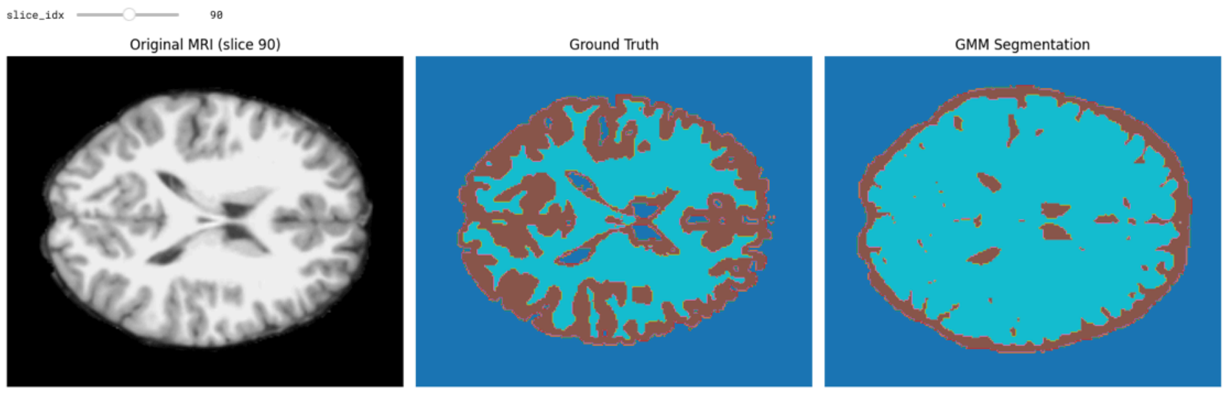




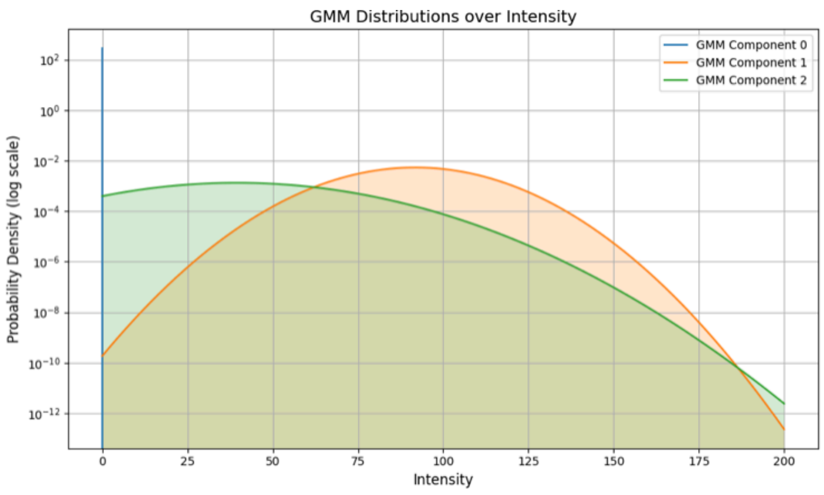
2. Pointwise Accuracy : 81.74%

3. Visualise Frequency of Points vs intensity graph for three labels.





4. Visualise GMMs distribution Generated



5. Where is Highest Misclassification and why?

Where and Why?

- Misclassification is often **highest at boundaries** between tissue types (e.g., gray matter ↔ white matter).
- GMM assumes Gaussianity and no spatial context → smooth transitions confuse the model.
- Low-intensity contrast regions and partial volume effects lead to overlap in feature distributions.

> Key Point : GMM does not model spatial continuity or complex textures—resulting in ambiguity around boundaries.

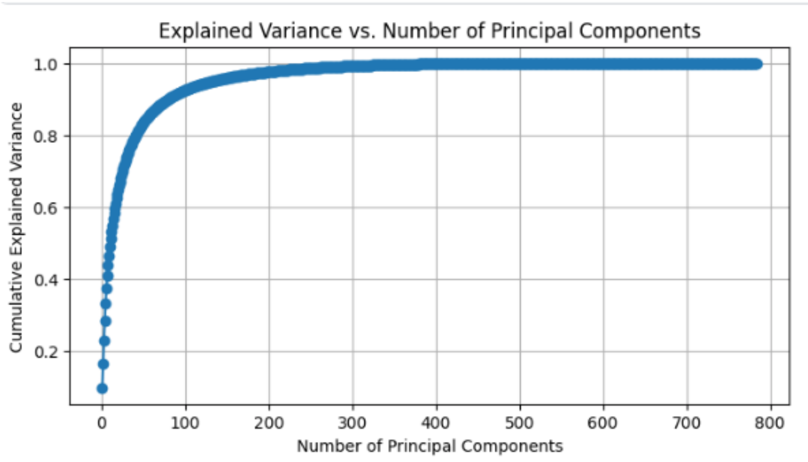
Question 4 : PCA

1. Variance and Number of Principal Components

- **Explained Variance vs. Number of Principal Components**:

  The curve shows a sharp increase initially and saturates beyond ~150 components.

  > Observation : The first few components capture most of the variance; diminishing returns after 150.

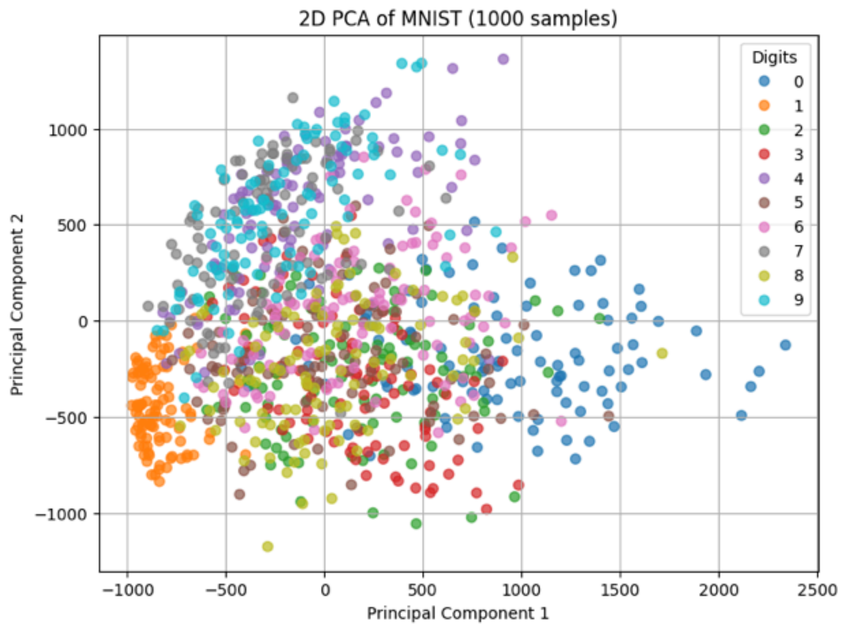


2. Visualisation using first 2 PCs, Observations

- **2D PCA Scatter Plot Visualization**:

  When projected onto the first two principal components, digits form distinguishable clusters with some overlap.

> Observation： PCA captures broad structure but loses fine-grained class separation in low dimensions



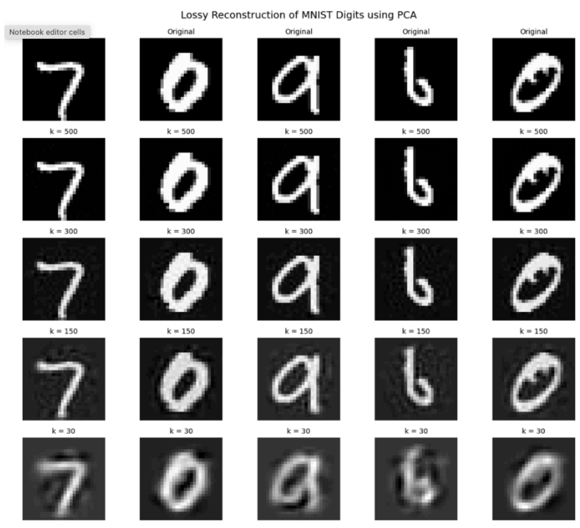
3. Reconstruction of 5 images

- **Image Reconstruction from PCA (Dimensions = 500, 300, 150, 30)**:

  Visual comparison of 5 original vs. reconstructed images:

  - **500 PCs**: Near-lossless reconstruction, visually similar to original.
  - **300 PCs**: Slight loss of sharpness; still very close to original.
  - **150 PCs**: Noticeable blur, but digit shape preserved.
  - **30 PCs**: Significant detail loss; digits still recognizable but less distinct.

> Observation： Lower dimensions lead to lossy compression. 150 PCs preserve class-level detail, 30 leads to poor visual quality.



4. Recall , Accuracy and Precision for MLP on different dimensions(PCA), Observations

PCA with 500 Dimensions

- Accuracy: 97.17%

> Observation ：Needs More computation to get higher

PCA with 300 Dimensions

- Accuracy： 97.20%

> Observation：Needs more computation/ epochs to get higher

PCA with 150 Dimensions

- Accuracy: 97.45%

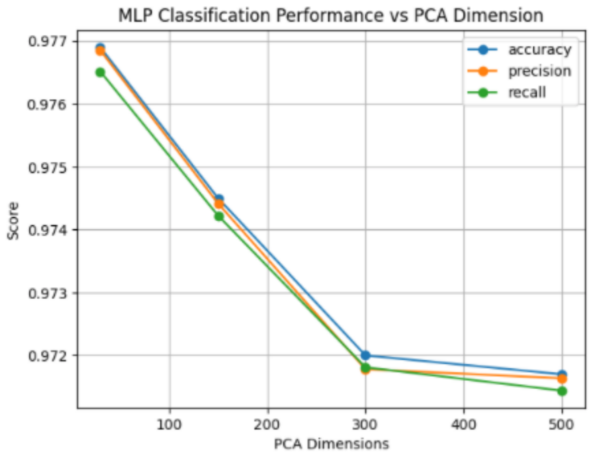
> Observation : Lower Dimension keeps most information and needs lesser compute for getting good accuracy

PCA with 30 Dimensions

- Accuracy: 97.69%

📊 PCA Classification Results:

	accuracy	precision	recall
500	0.9717	0.971636	0.971440
300	0.9720	0.971781	0.971811
150	0.9745	0.974416	0.974223
30	0.9769	0.976853	0.976516



Theory Questions :

How does PCA help mitigate the curse of dimensionality?

PCA reduces the number of features by projecting data onto a lower-dimensional space that retains the most variance. This simplifies models, reduces overfitting, and improves computational efficiency, which helps mitigate the curse of dimensionality.

When might PCA be ineffective in high-dimensional spaces?

- When the **signal-to-noise ratio is low**, PCA might preserve noise.
- If **important information is non-linear**, PCA (a linear method) fails to capture it.
- In **sparse high-dimensional data** (e.g., text or genetics), PCA may not find meaningful projections.

Is maximum variance always the most informative?

No, high variance does not always imply high information for the task.

Example:

In **classification problems**, class-separating features may have low variance.

> E.g., a small but consistent difference in pixel intensity across two digits in MNIST might be crucial for classification but may be ignored by PCA.

Question 5 : Auto Encoder

1. Histograms for Reconstruction Error for Normal digit and Anomaly with Different Bottleneck Dimensions and their Scores.

Reconstruction errors were computed as the Mean Squared Error (MSE) between the input and output images.

Observation:

- Normal digits showed low reconstruction error.
- Anomalous digits had significantly higher reconstruction error.

  Histograms clearly separated the two distributions, indicating effectiveness of the autoencoder in detecting anomalies.

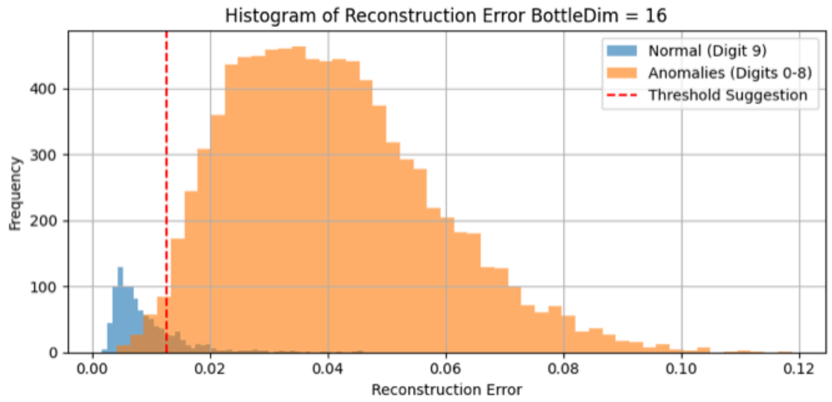
A threshold was selected based on the point that best separates the two histogram peaks (using visual inspection or methods like Youden's J statistic).
Images with reconstruction error above this threshold were classified as anomalies.

The model was evaluated on the test set using:

- **Precision**: Proportion of correctly detected anomalies among all predicted anomalies.
- **Recall**: Proportion of actual anomalies that were correctly detected.
- **F1-Score**: Harmonic mean of precision and recall.

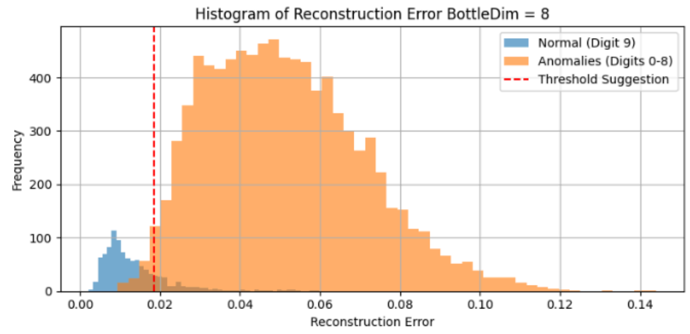
These metrics demonstrated the effectiveness of using reconstruction error for anomaly detection.

```
Results for Bottleneck dimension = 16
```



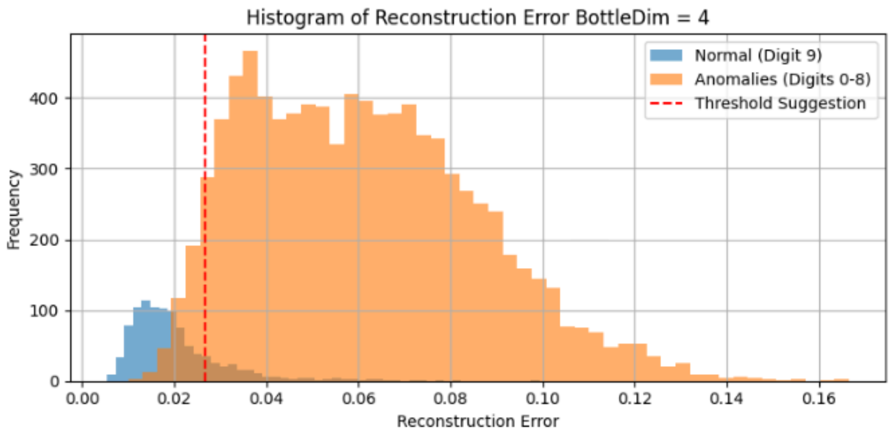

```
Threshold : 0.0127
Precision: 0.8494
Recall:    0.8107
F1-score:  0.8296
```

```
Results for Bottleneck dimension = 8
```



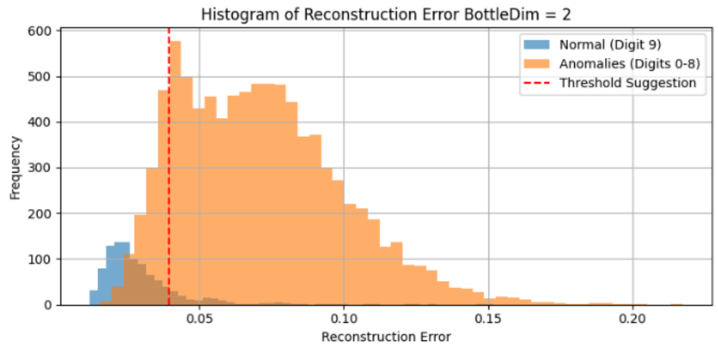

```
Threshold : 0.0185
Precision: 0.8562
Recall:    0.8087
F1-score:  0.8318
```

```
Results for Bottleneck dimension = 4
```

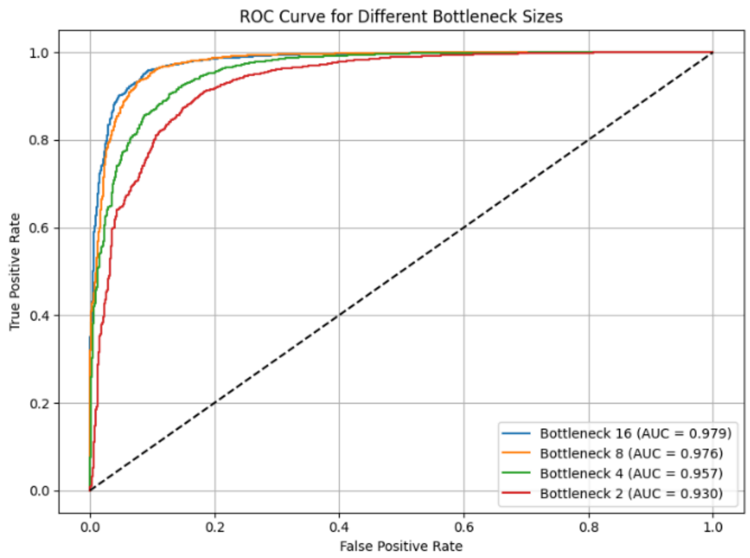



```
Threshold : 0.0268
Precision: 0.6407
Recall:    0.8147
F1-score:  0.7173
```

```
Results for Bottleneck dimension = 2
```



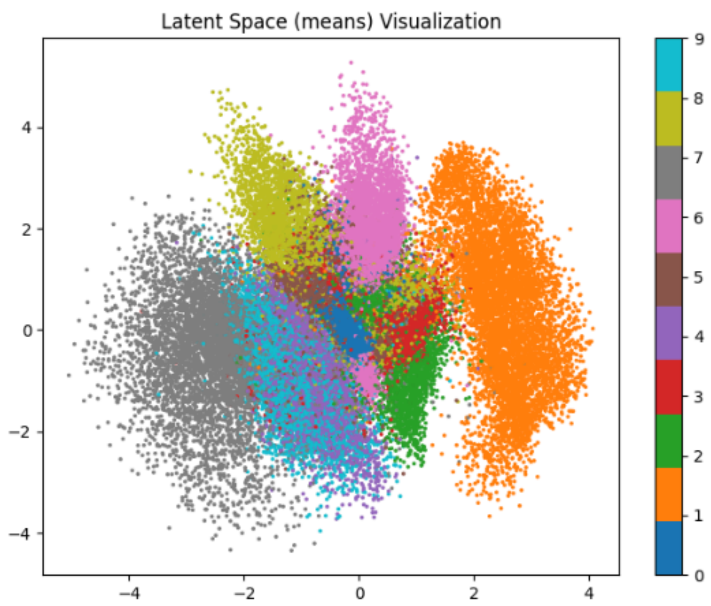

```
Threshold : 0.0394
Precision: 0.4391
Recall:    0.8434
F1-score:  0.5775
```



Bottleneck = 16 had the highest AUC Score, Bottleneck 8 was very close. Bottleneck 2 had was comparatively poor.

Question 6 : Variational Auto Encoder

1. **Latent Space when both KL and Reconstruction Loss are present**



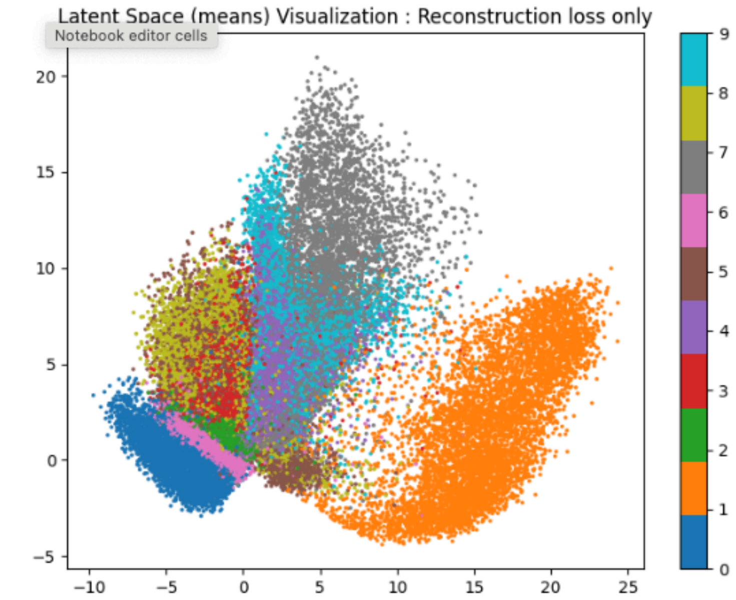
Sampling points from a 2D Gaussian grid over the latent space and passed them through the decoder to generate images.

Observation: The reconstructions were smooth across the grid. Each region of the latent space corresponded to a specific digit, and transitions between digits were gradual.

Conclusion: The model successfully learned a meaningful 2D latent representation, allowing structured sampling and smooth interpolation.

Epoch 20 Loss : 151.463

2. **Latent Space means when only Reconstruction Loss is Present.**



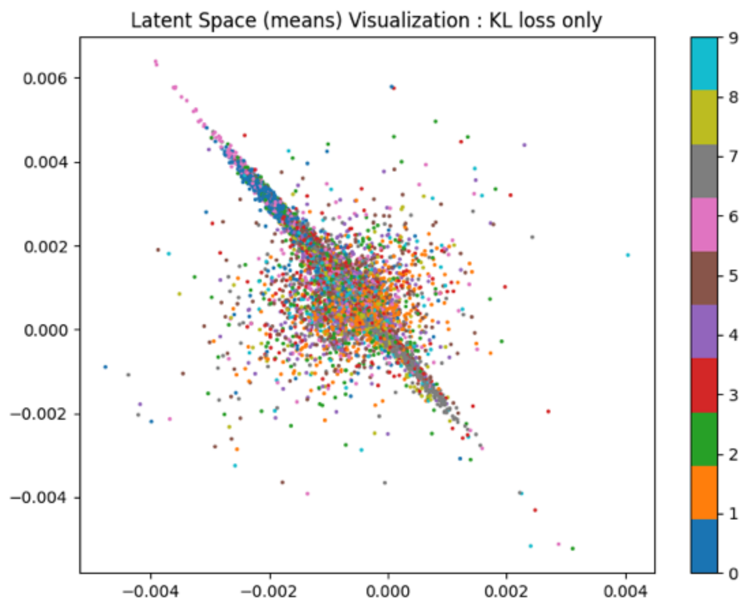
Removing the KL Divergence loss and trained the VAE using only the BCE reconstruction loss.

Observation: The latent space showed irregular, discontinuous clusters. Some digits were overrepresented while others were poorly formed.

Conclusion: KL Divergence loss is necessary for enforcing a smooth and continuous latent space. It ensures that the latent vectors follow a known distribution, enabling meaningful interpolation and sampling.

Epoch 10 Loss : 147.952

3. **Latent Space Means when only KL Loss is present**



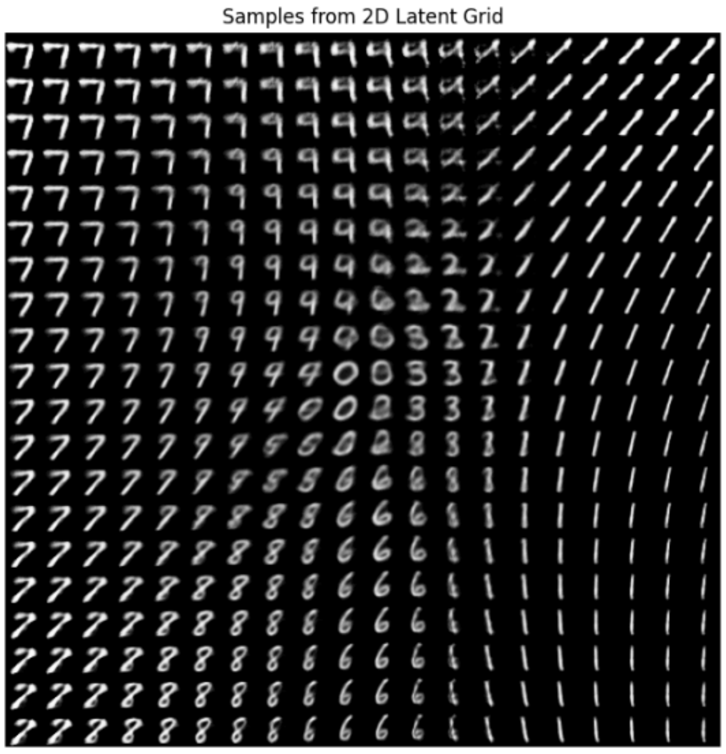
Removing the reconstruction loss and trained the model using only the KL Divergence loss.

Observation: The latent space collapsed to a tight Gaussian distribution with no clear class separation. Generated reconstructions were nonsensical and resembled noise.

Conclusion: The reconstruction loss is essential for ensuring that the decoder learns to map latent variables to valid data samples. Without it, the model fails to learn the data structure.

Epoch 10 Loss : 2.14e-7

4. **Sampling from Gaussian and reconstruction with latent space dim = 2**

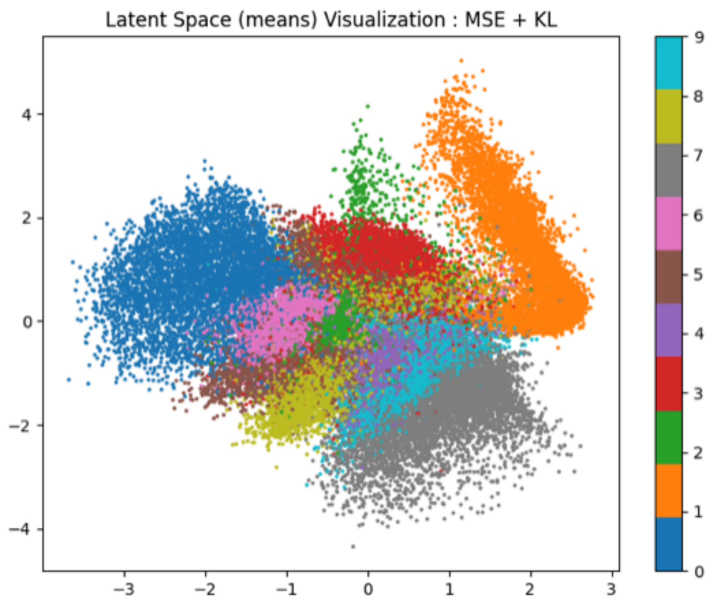


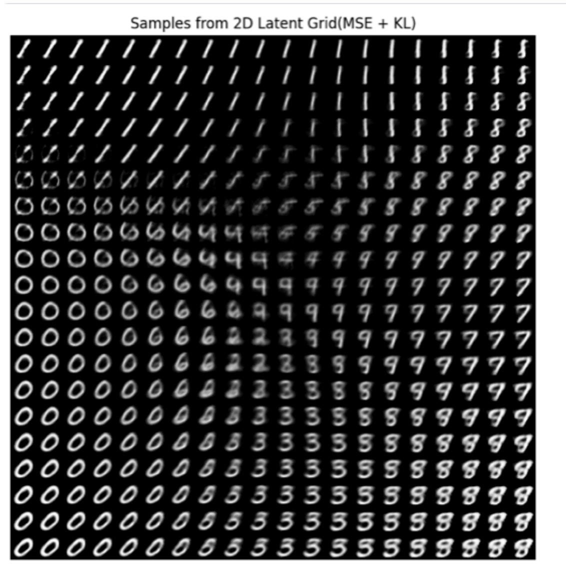
Sampling points from a 2D Gaussian grid over the latent space and passed them through the decoder to generate images.

Observation: The reconstructions were smooth across the grid. Each region of the latent space corresponded to a specific digit, and transitions between digits were gradual.

Conclusion: The model successfully learned a meaningful 2D latent representation, allowing structured sampling and smooth interpolation.

5. **Changing Binary Cross entropy loss to MSE.**





Replacing the BCE loss with Mean Squared Error (MSE) and repeated the sampling experiment.

Observation: The reconstructed images were slightly blurrier than those from BCE, and class boundaries in the latent space were less distinct.

Conclusion: While MSE can be used for reconstruction, it may result in less sharp outputs and slightly less well-separated latent representations compared to BCE in binary image datasets like MNIST.

Epoch 10 Loss : 39.2819