DL - LAB 7

Analysis of the obtained results

Task 1 (normalization.py)

> For a fair comparison of the raw and normalised data, we've set an accuracy threshold of 90%. When that accuracy is reached, we stop training and observe the number of epochs passed and the loss convergence.

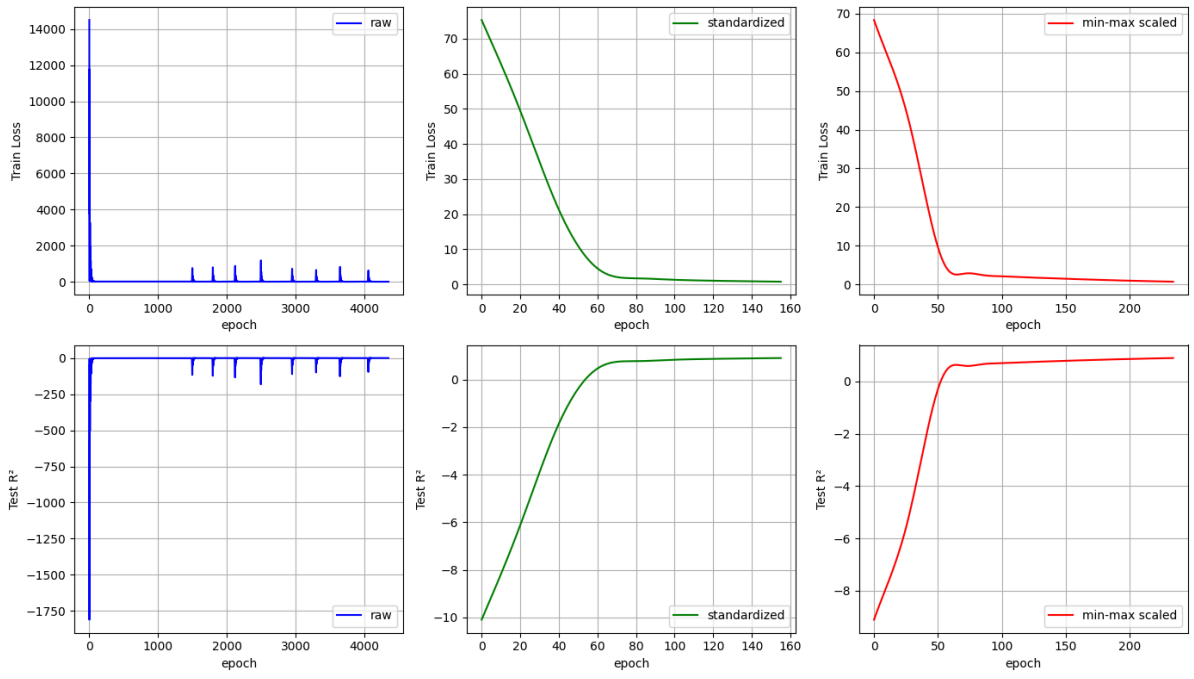
> Console output:

Training model with raw data...
EARLY STOP TRAINING: Accuracy: 0.9000351428985596, Loss: 0.6877 at Epoch: 4349

Training model with standardised data...
EARLY STOP TRAINING: Accuracy: 0.9000875353813171, Loss: 0.7388 at Epoch: 155

Training model with min-max scaled data...
EARLY STOP TRAINING: Accuracy: 0.9000049233436584, Loss: 0.7031 at Epoch: 234

> Progression plot:



> Observations:

1. Raw Data:

   The convergence process is extremely slow. The loss starts at an abominable value (~14.5k in our case) and is still above 3.5 after 1300 epochs. Accuracy (Test R^2) starts at -577, crosses 0 only after epoch 421, and slowly climbs, reaching ~0.59 after 1387 epochs. The training shows significant instability (e.g., loss jumps back up to ~11779 at epoch 2). Convergence is poor even after thousands of epochs.

2. Standardised Data:

   Much faster. Loss starts at ~75 and drops below 2.0 by epoch 70. Accuracy starts at -10, crosses 0 at epoch 54, and reaches ~0.80 by epoch 92. The training appears largely converged by epoch ~100-120, although accuracy continues to slowly improve.

3. Min-Max Scaled Data:

   Also much faster than raw. Loss starts slightly lower at ~68 and drops below 3.0 by epoch 60. Accuracy starts at -9, crosses 0 at epoch 53 (very similar to standardised), but takes longer to reach higher values, hitting ~0.80 around epoch 151. Convergence seems largely achieved around epoch ~150-170, though accuracy continues to improve slowly.

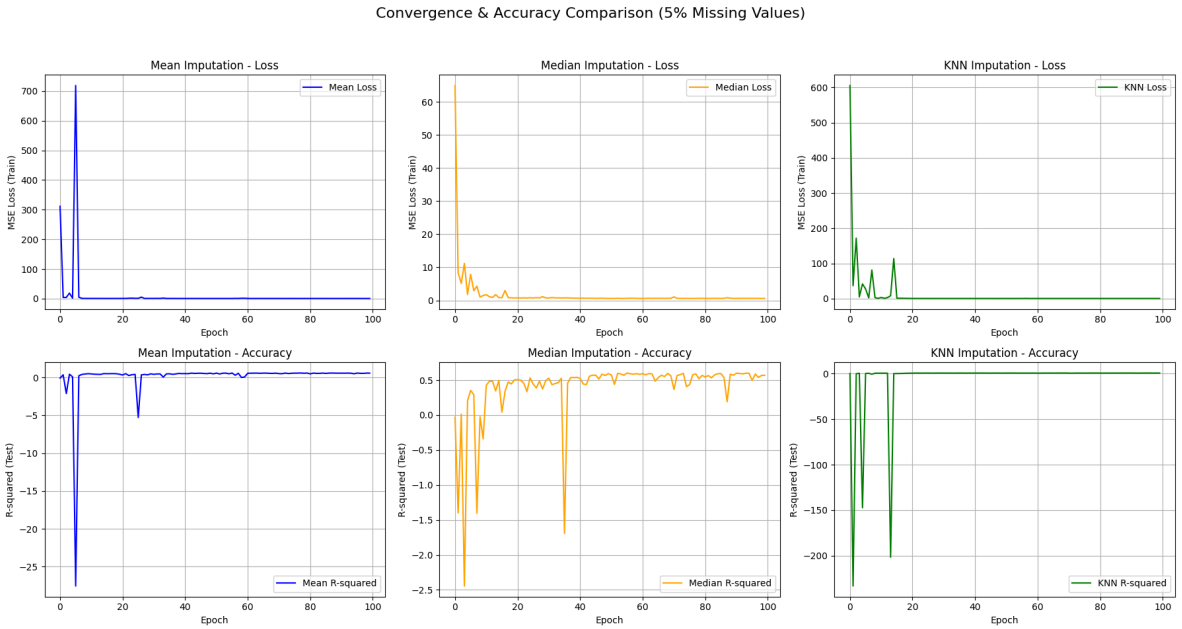
Task 2 (missing_values.py)

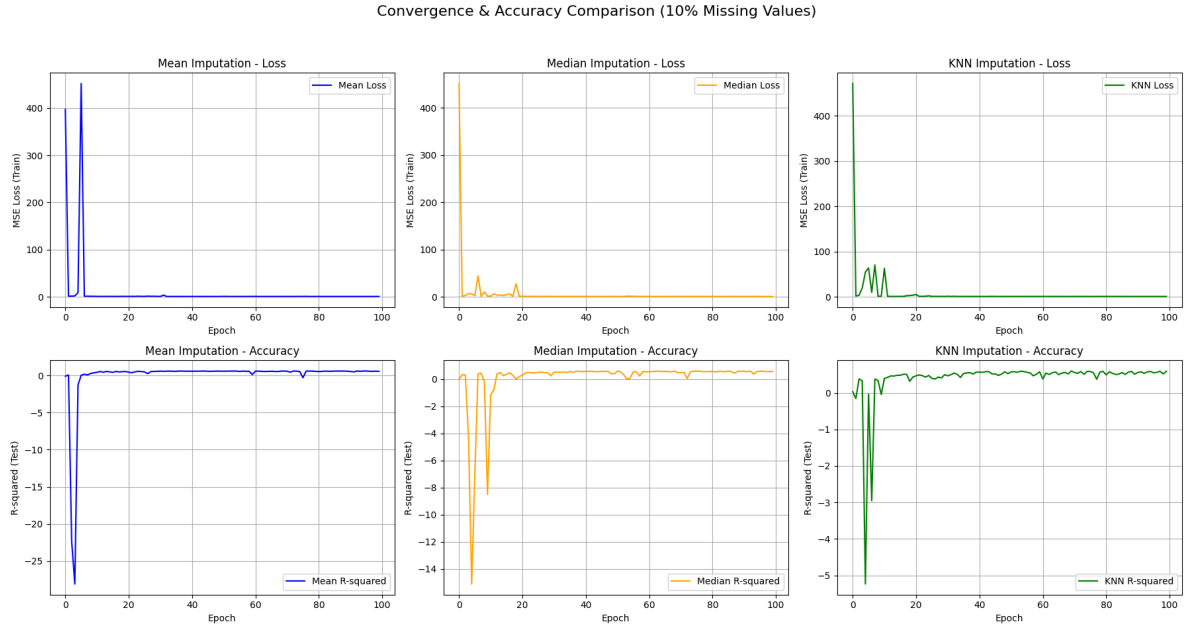
> During our experiment, we've explored 3 different missing quantities in the training dataset (5%,10%, and 15%), epochs number is constant at 100 and accuracy is computed using R_squared, same as the previous task.

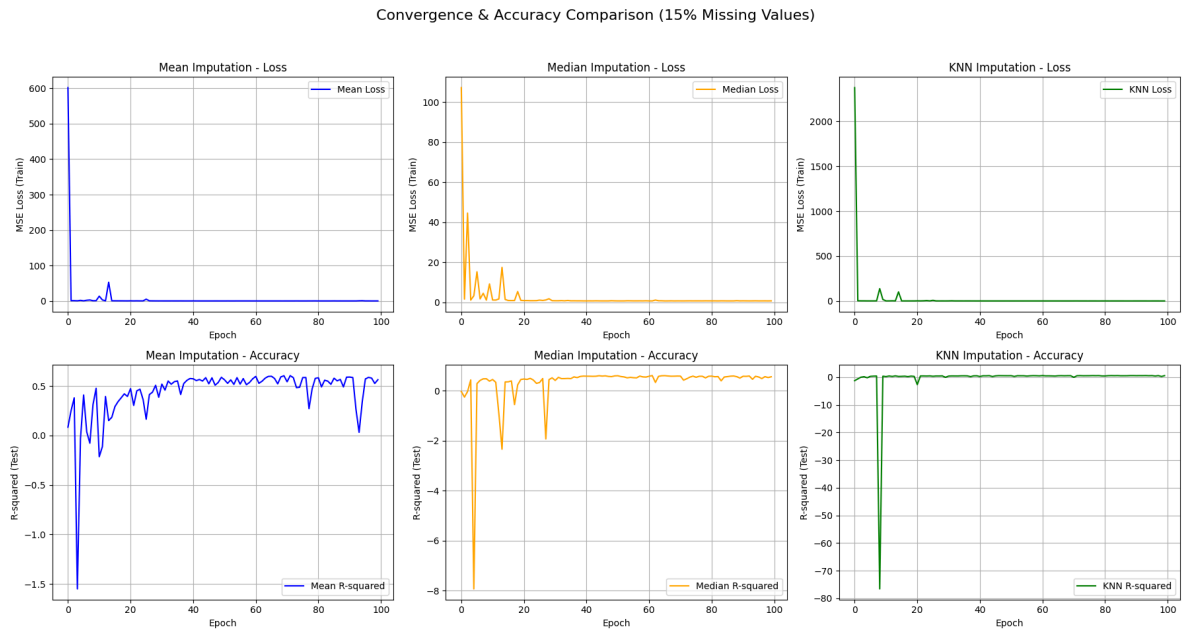
> Best performances obtained:

- 5 % of missing data
  > `Mean_Accuracy,Median_Accuracy,Knn_Accuracy`
  > `0.5667818784713745,0.5857614874839783,0.5872500538825989`
- 10 % of missing data
  > `Mean_Accuracy,Median_Accuracy,Knn_Accuracy`
  > `0.5914099216461182,0.4634761214256286,0.5347585678100586`
- 15 % of missing data
  > `Mean_Accuracy,Median_Accuracy,Knn_Accuracy`
  > `0.564578115940094,0.5716609358787537,0.5623626112937927`

> Progression plot:







> Observations of:

1. Convergence Speed & Stability (Training Loss - MSE)

    a. For mean imputation, we observe that the method is the fastest and smoothest convergence across all missing data percentages (5%, 10%, 15%). The loss drops rapidly in the first few epochs and then stabilises at a low value with minimal subsequent spikes.

    b. Median Imputation: Also converges quickly, but exhibits significant instability, especially as missing data increases. Large loss spikes are visible in the early epochs for 10% and are particularly pronounced (reaching >1500) at 15% missing values before settling down.

    c. KNN Imputation (k=5): Shows rapid initial convergence but is prone to large loss spikes in the early epochs, similar to Median imputation. This instability is noticeable at 5% and becomes more severe at 10% and 15% (spikes >1000 and >800, respectively).

2. Sensitivity to Increased Missing Data

    a. Mean Imputation -> Shows relative robustness up to 10% missing data in terms of R-squared, although stability decreases slightly. Performance collapses in the last experiment with 15% missing data.

    b. Median Imputation -> Is highly sensitive. While adequate at 5%, its performance (both loss stability and R-squared) degrades drastically at 10% and becomes essentially unusable at 15%.

    c. KNN Imputation -> Shows decreasing R-squared and increasing instability (in both loss and R-squared) as missing data increases. While it yields a non-zero average R-squared even at 15%, the extreme volatility makes it a risky choice compared to Mean imputation at lower percentages.

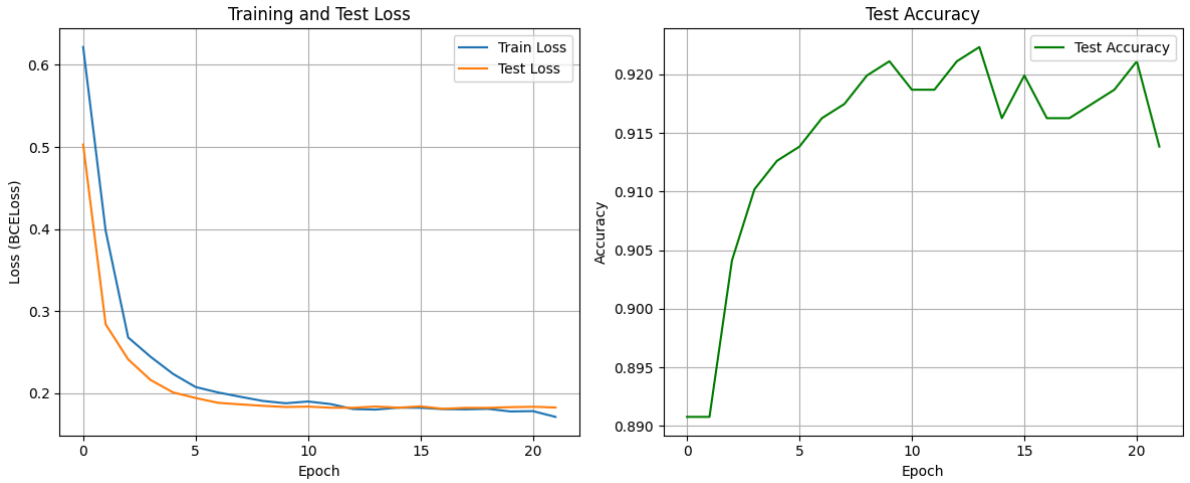
Task 3 (category_encoding.py)

> For this task/experiment, we've used a Bank Marketing Dataset to train our model.

> Dataset structure:

```
"age";"job";"marital";"education";"default";"housing";"loan";"contact";"month";"day_of_week";"duration";"campaign";"pdays";"previous";"poutcome";"emp.var.rate";"cons.price.idx";"cons.conf.idx";"euribor3m";"nr.employed";"y"
```

> Progression plot:



> Observations:

- The model trained well, achieving a high test accuracy of around 91-92%. The learning curves show typical behaviour: decreasing loss and increasing accuracy.
- Early stopping effectively prevented significant overfitting, which appeared as the test loss began to plateau and slightly increase while the training loss continued to decrease.
- The increased dropout rate seems to have kept the model reasonably regularised, but the overall performance characteristics (high accuracy, plateauing, early stopping) are similar to what might be expected if the model is reaching its performance limit for this dataset and architecture.