Group 3

Karan Punjabi: N01514624

Amrita Sodhi: N01513240

Richu Thankachan: N01516068

Remya Raj: N015113680

1) Introduction:

In this project, we aim to develop a recurrent neural network (RNN) model to predict the 'pkh' column based on the 'rate' and 'ems' features from a tabular time-series dataset. The dataset contains 290 monthly records spanning from 1997 to 2021, and the goal is to build an accurate predictive model for the 'pkh' values.

2) Description of the ML model and its architecture:

The initial model was implemented using an RNN architecture in PyTorch. It took the 'rate' and 'ems' columns as input features and predicted the 'pkh' column as the output. The model was composed of a single RNN layer followed by a linear (fully connected) layer. It was trained using the Mean Squared Error (MSE) loss function and the Adam optimizer. The hidden size of the RNN was set to 64 units.

3) Report results on both training and test (losses and accuracies), and training/test graphs:

The initial model was trained for 1000 epochs using the training dataset. The training loss decreased over epochs, indicating the model's learning. The test loss, calculated on the separate test dataset, was recorded as 0.0502. The accuracy metric may not be directly applicable in this regression task, so we primarily focused on loss evaluation.

4) Implement an improvement of the prediction outcomes based on modifications to the model’s architecture. Describe the modifications and report the new outcomes as stated in sections (2), (3):

In the second notebook, we introduced modifications to the RNN architecture to potentially improve predictive performance. We increased the hidden size of the RNN to 128 units, aiming to capture more complex patterns in the data. This modification was intended to enhance the model's representational capacity and potentially lead to better predictions.

After training the modified model, we observed that the training loss continued to decrease, indicating that the model was able to learn even with the increased complexity. However, it's crucial to note that model performance should not be solely judged based on training loss. The test loss was recorded as 0.0502, where isn't much difference but it still allows us to compare the performance of the modified model to the initial model.

We made the following changes to the initial RNN model:

Batch Normalization (BN): We added batch normalization layers after the RNN layer. This helps stabilize and accelerate training by addressing the vanishing/exploding gradient problem and adds a regularization effect to prevent overfitting.

Weight Initialization: We used weight initialization for both the RNN layer and the fully connected output layer. This improved weight initialization helps the model converge more effectively, reducing the risk of getting stuck in poor local minima.

Increased Model Complexity (More RNN Layers): We increased the number of RNN layers from 1 to 2. This enhances the model's ability to capture complex temporal patterns in the time-series data, allowing it to learn more meaningful features.

5) Interpretation and discussion of all results:

The results from both the initial and modified models suggest that the modified model, with the increased hidden size, might have the potential to improve prediction accuracy. However, it's essential to perform more rigorous evaluation, potentially using cross-validation, and exploring other model enhancements to draw more definitive conclusions.

6) Conclusion:

In conclusion, this project aimed to predict the 'pkh' column from a tabular time-series dataset using recurrent neural networks. We implemented an initial RNN model and a modified version with increased complexity. The modified model showed promise, but further analysis and enhancements are needed to confirm its superiority.

7) The tasks that each member of the group performed during the project activities:

Amrita Sodhi: Loading Dataset and preprocessing.

Remya Raj: Created Initial RNN Model with Test Loss as 0.0502

Richu Thankachan = Created Modified RNN Model with same Test Loss as 0.0502

Karan Punjabi: Created Imporved RNN Model with Test Loss as 0.0473