EX-3

_or muller 313599144

1)decision tree building:

please follow the input the way its put in the “main” function in the bottom of the code it, its exactly how we were asked to put the input, only for “is late” function I used an actual line from the Data Frame so please make sure it’s the same as the input (Pattern-wise)

2)Using Neural Networks and Decision Trees in a Boosting Algorithm

Yes, it is possible to use both neural networks and decision trees together in a boosting algorithm. This approach is known as heterogeneous ensemble learning or hybrid boosting. Here's how it could work:

Boosting Algorithm Process:

1. The boosting algorithm would treat each model (neural network or decision tree) as a weak learner (meaning slightly better than random guessing).
2. In each iteration of the boosting process, either a neural network or a decision tree could be trained on the weighted data.
3. The algorithm would adjust the weights of the training samples based on the performance of each model, focusing more on the misclassified samples.
4. The final ensemble would consist of a combination of neural networks and decision trees.

Strengths of Combining Models:

This approach can potentially leverage the strengths of both types of models:
- Decision trees are good at handling categorical data and are highly interpretable.
- Neural networks excel at capturing complex, non-linear relationships in the data.

Using Boosting Weights in Training a Deep Neural Net

a) Sample weighting:

Assign each training sample a weight based on the boosting algorithm.
During the training of the neural network, use these weights to calculate a weighted loss function.

This ensures that the network pays more attention to the samples that were misclassified by previous models in the ensemble.

b) Loss function modification:
Modify the loss function to incorporate the sample weights.
For example, if using cross-entropy loss, you could multiply each sample's loss by its corresponding weight before averaging.

c) Gradient calculation:
When calculating gradients during backpropagation, scale them by the sample weights.
This will cause larger updates for misclassified samples with higher weights.

d) Batch selection:
Use weighted random sampling to create mini-batches for training.
Samples with higher weights would have a higher probability of being selected in each batch.

e) Learning rate adjustment:
Potentially adjust the learning rate based on the average weight of the current batch.
This could help prevent overfitting to a few high-weight samples.

f) Early stopping:
Modify early stopping criteria to consider the weighted validation error rather than the standard error.

Summary
By incorporating these modifications, the neural network can effectively learn from the boosting weights, focusing more on the difficult examples and potentially improving the overall performance of the ensemble.
This approach allows for a synergy between the boosting algorithm and the neural network, potentially leading to a more robust and accurate model than either method alone.