2- Clearly describe each of these characteristics of a neural network:

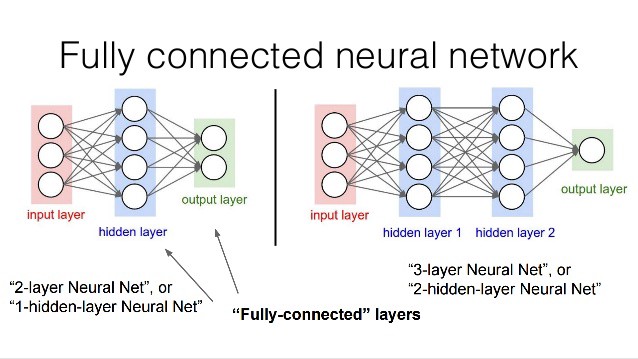
a. Layered:

b. Feedforward:

A neural network consists of a layered, feedforward, completely connected network of artificial neurons, or nodes. The **feedforward nature** of the network restricts the network to a single direction of flow and does not allow looping or cycling. It means the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. **The neural network is composed of two or more layers**, although most networks consist of three layers: an input layer, a hidden layer, and an output layer. There may be more than one hidden layer, although most networks contain only one, which is sufficient for most purposes.   
In the other word, Perceptron are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and hence are called hidden layers.

C. Fully Connected Neural Network/Completely Connected Neural Network

**Fully connected** layers connect every neuron in one layer to every neuron in another layer. We can see a schematic of a Fully Connected Neural Network below:



As we can see, each of the nodes in Neural Network, either in input, hidden or output layer is connected with a path. Because this image shows the Feedforward Fully Connected Neural Network, all the arrows are toward right. In other words, in a fully connected layer each neuron is connected to every neuron in the previous layer, and each connection has its own weight. This is a totally general-purpose connection pattern and makes no assumptions about the features in the data. It's also very expensive in terms of memory (weights) and computation (connections).

4- Should we prefer a large hidden layer or a small one? Describe the benefits and drawbacks of each.

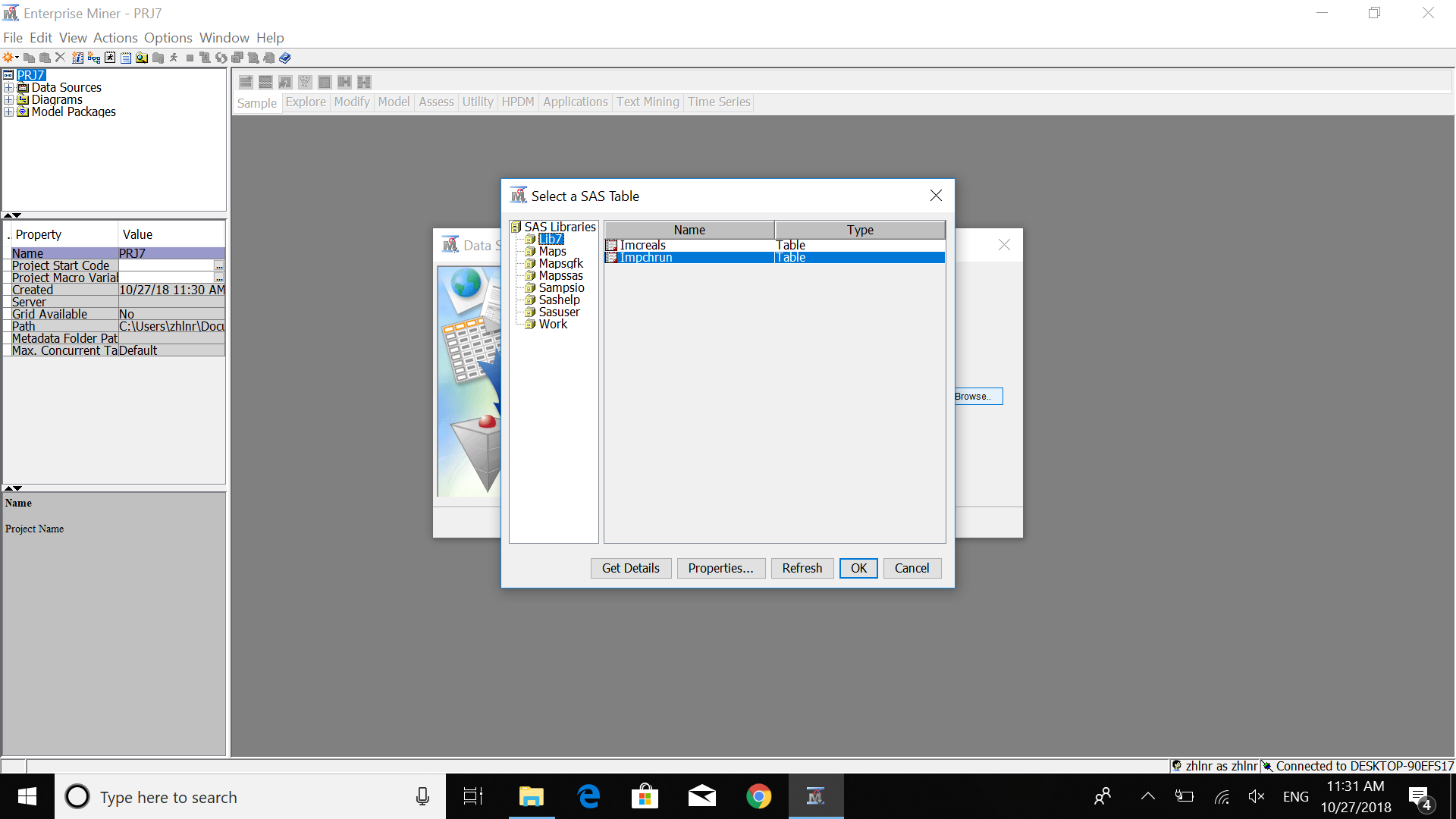
The number of hidden layers and the number of nodes in each hidden layer are both configurable by the user, depending on the particular classification task at hand.

Since more nodes in the hidden layer **increases** the power and flexibility of the network for identifying complex patterns, one might be tempted to have a large number of nodes in the hidden layer. On the other hand, an overly large hidden layer leads to overfitting, memorizing the training set at the expense of generalizability to the validation set. If overfitting is occurring, one may consider **reducing** the number of nodes in the hidden layer; conversely, if the training accuracy is unacceptably low, one may consider increasing the number of nodes in the hidden layer.

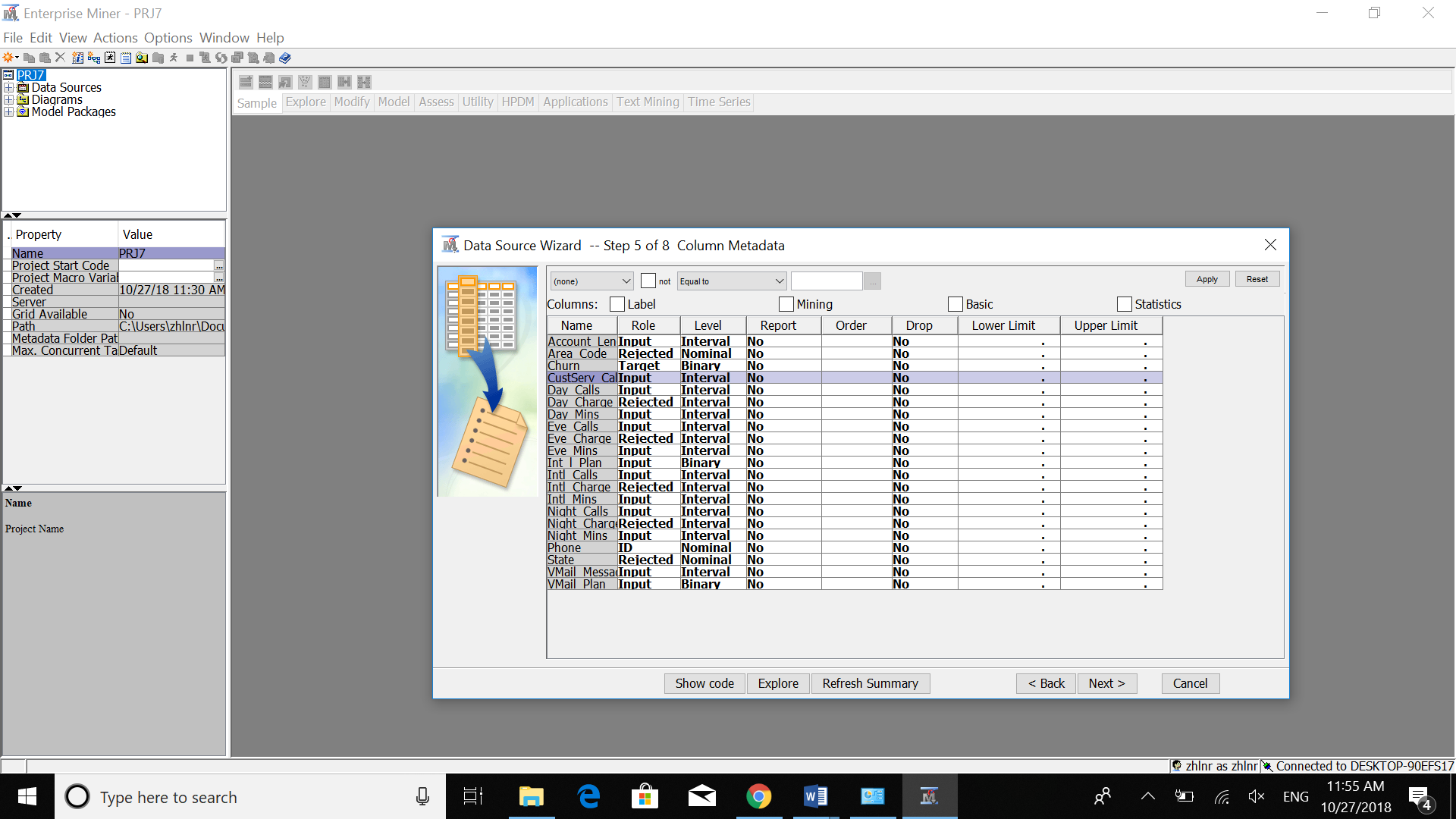
12-Generate a neural network model for classifying *churn* based on the other variables. Describe the topology of the model.

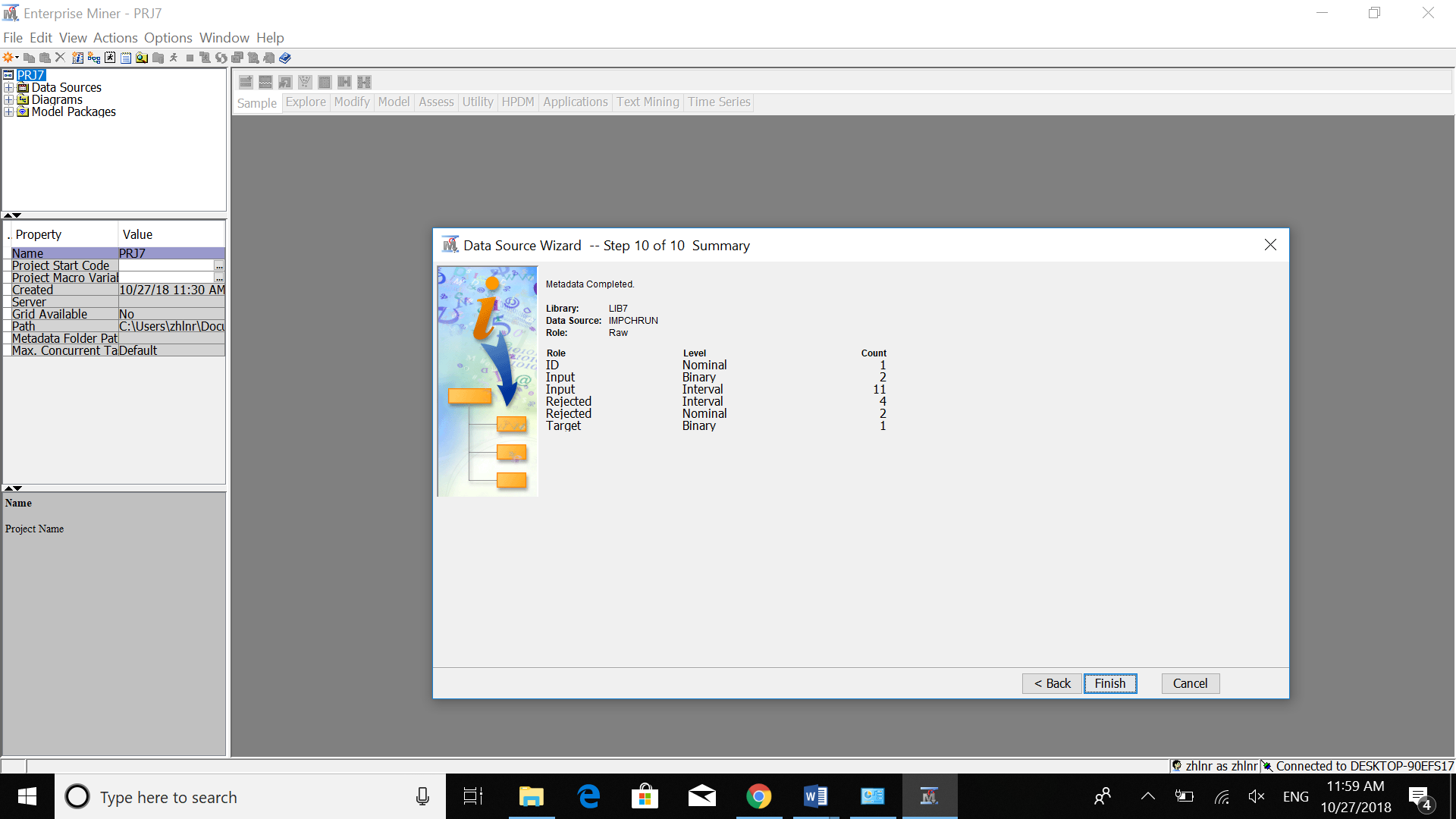
In order to do this question, we should follow these steps:

First of all, as we know, the neural network works with numerical variables and not categorical. Thus, we have to change the type of these kind of variables in our data set. In order to do so, I created another data set and change the level of variables as explained.

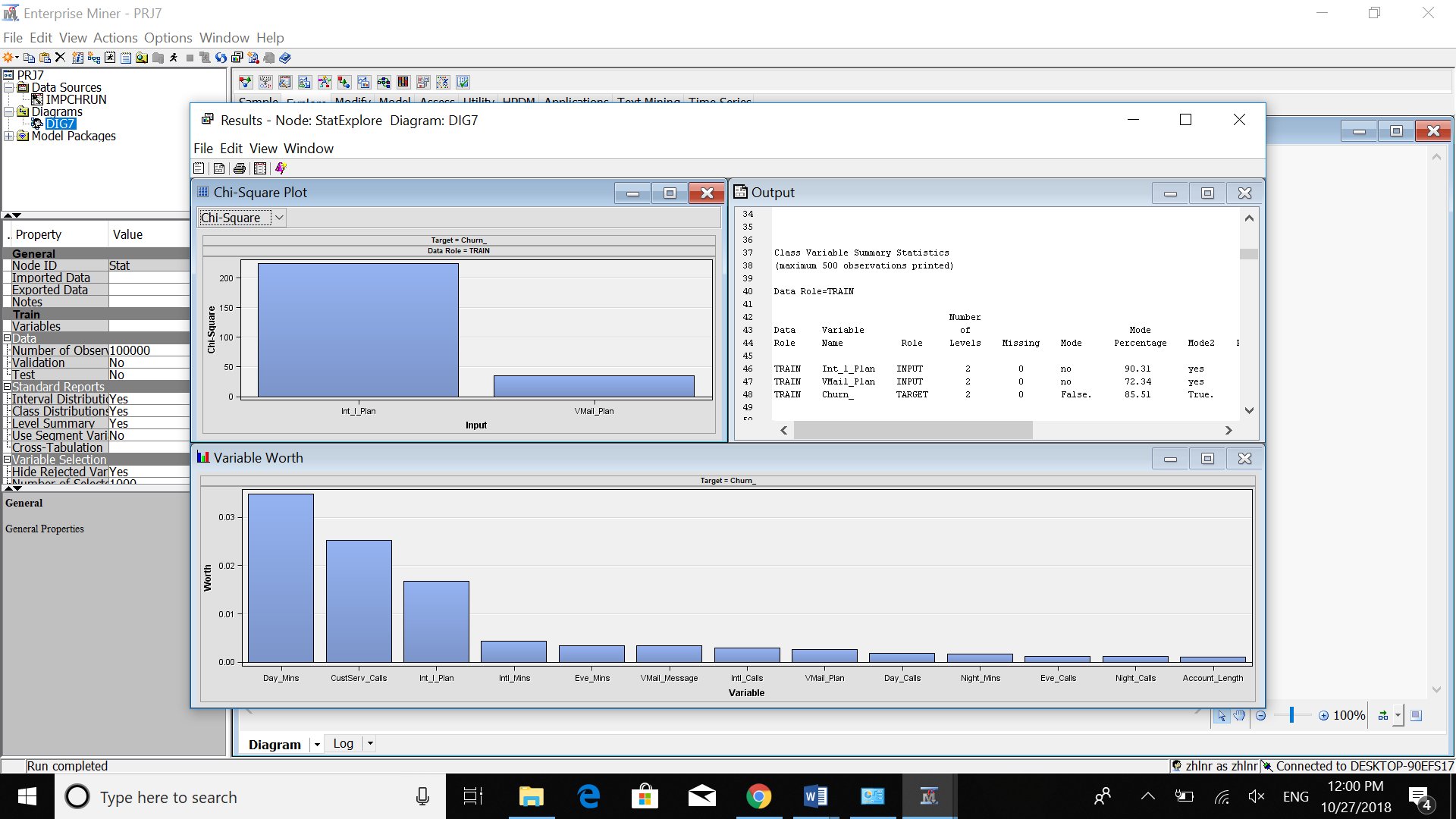


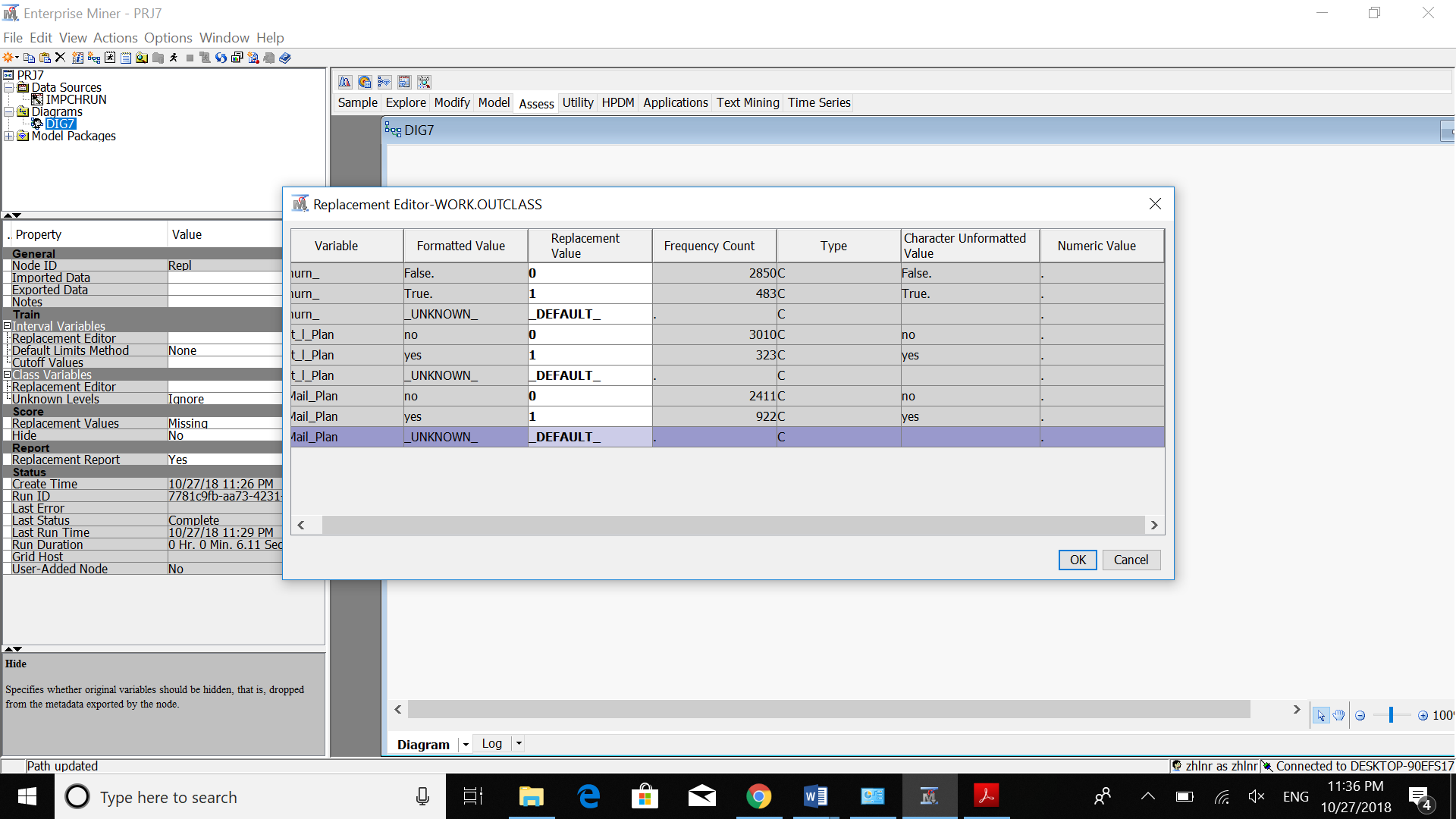
Furthermore, as we showed in previous homework, we have some variables which are perfectly correlated to each other. Like, day charge and day mins, night charge and night min and etc. So I rejected the “charge” variables as the step of preparation of data. Here, we can see the list of variables with their changes role and level:

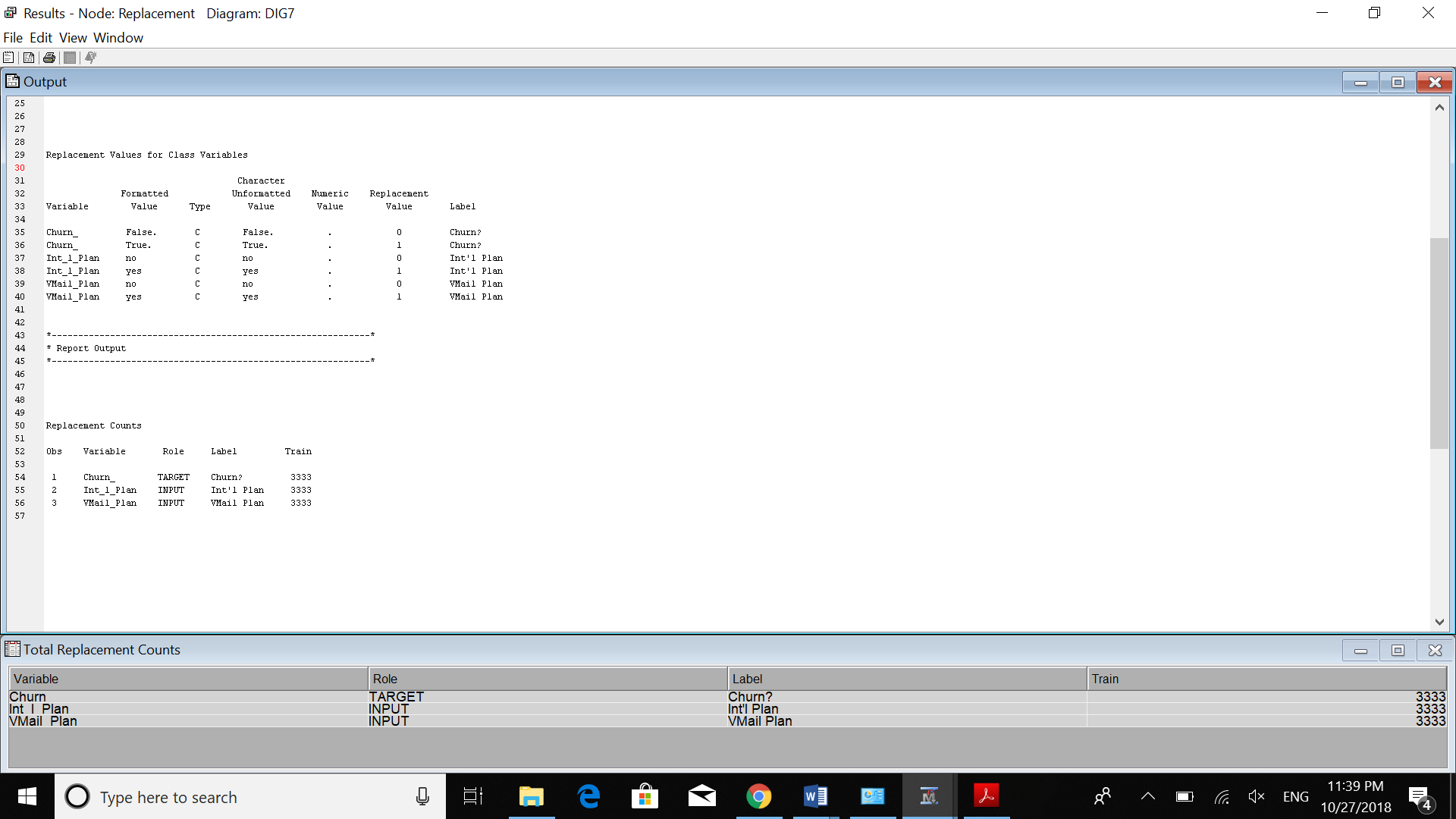




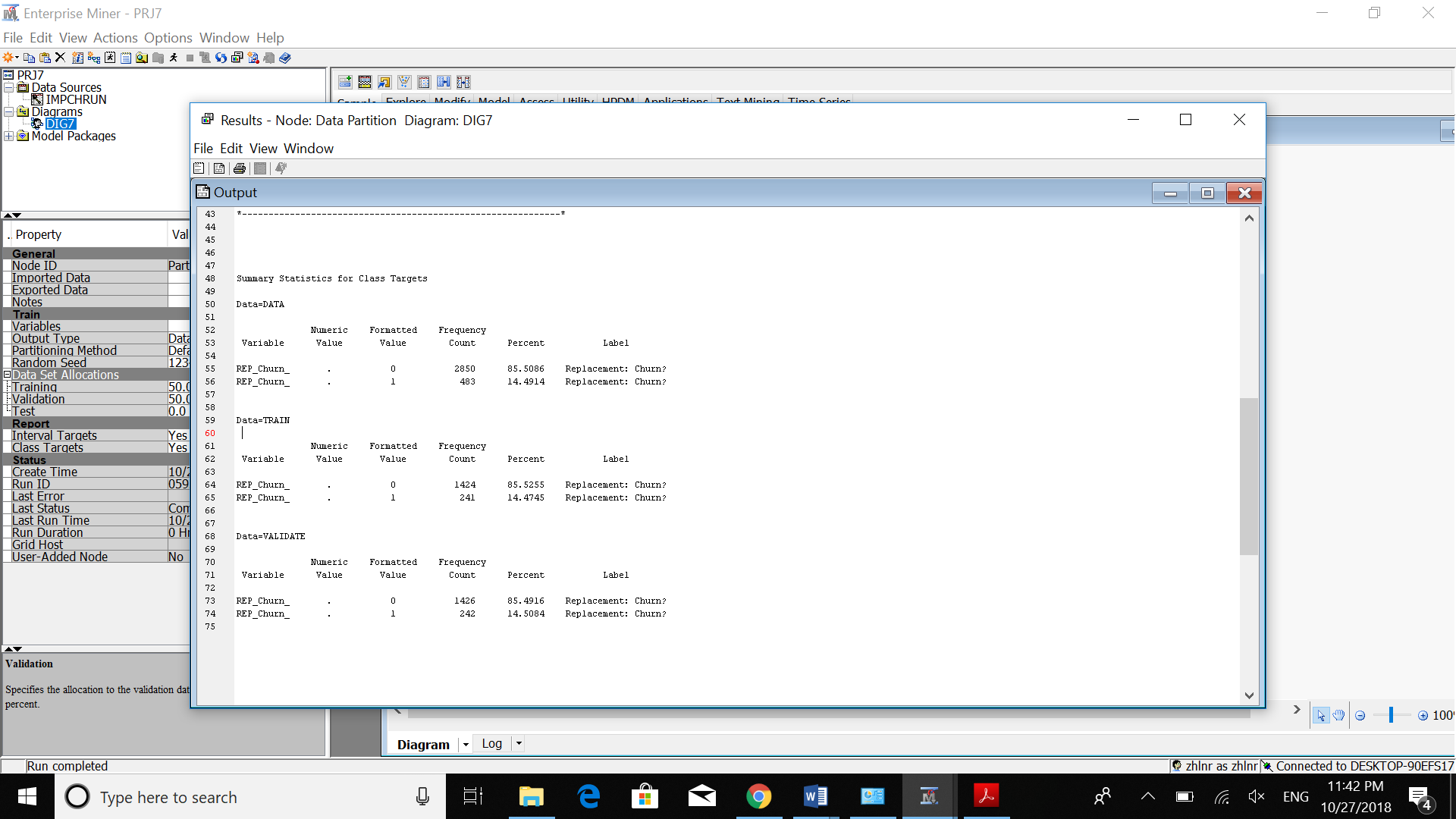
Furthermore, I defined the type of binary variables as flag variable. For example, if a customer has international plan which equals to “True” we can consider it as 1 and if a customer does not have international plan which equals to “False”, we can consider it as 0. We can see this step in following images:



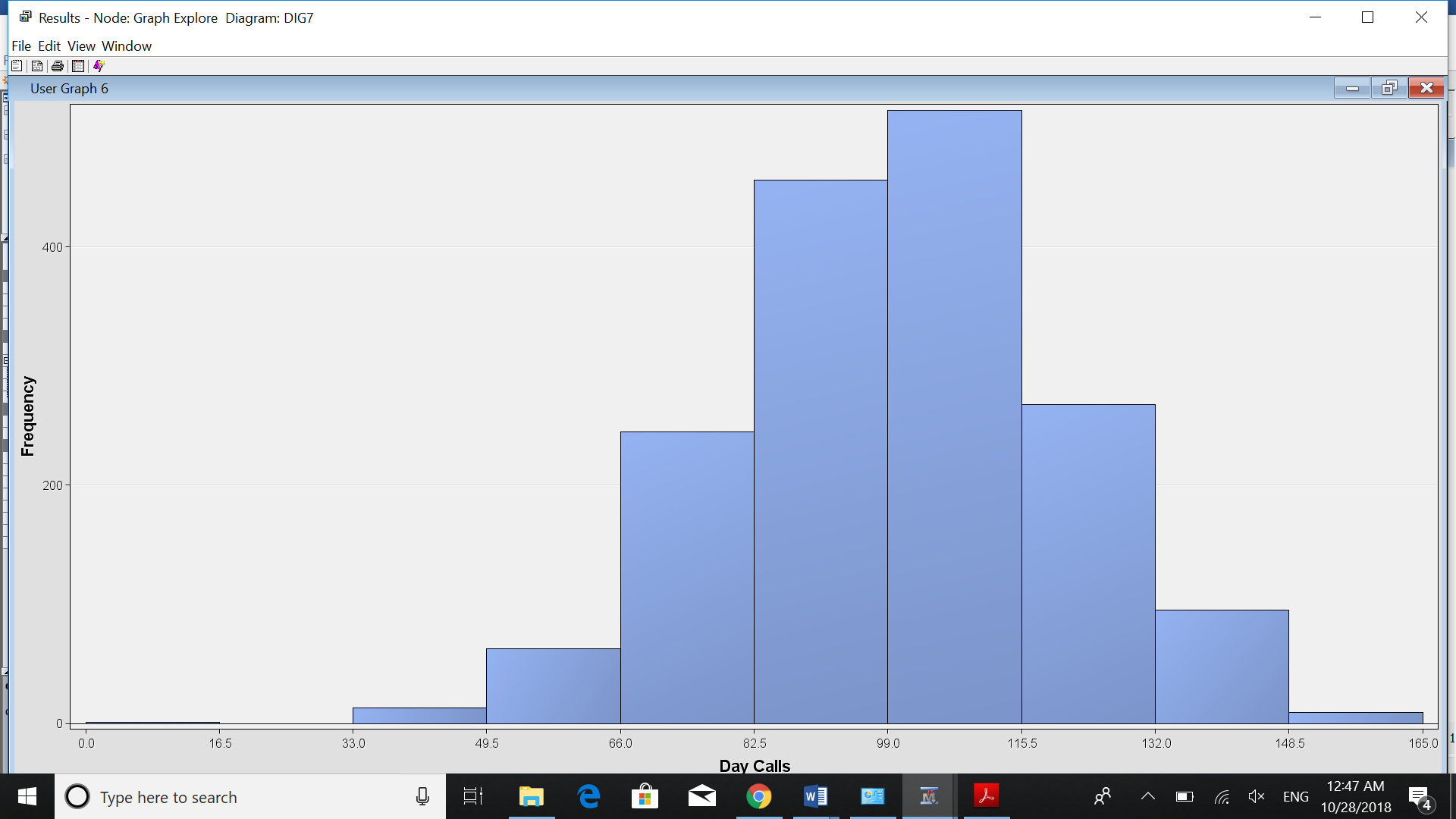


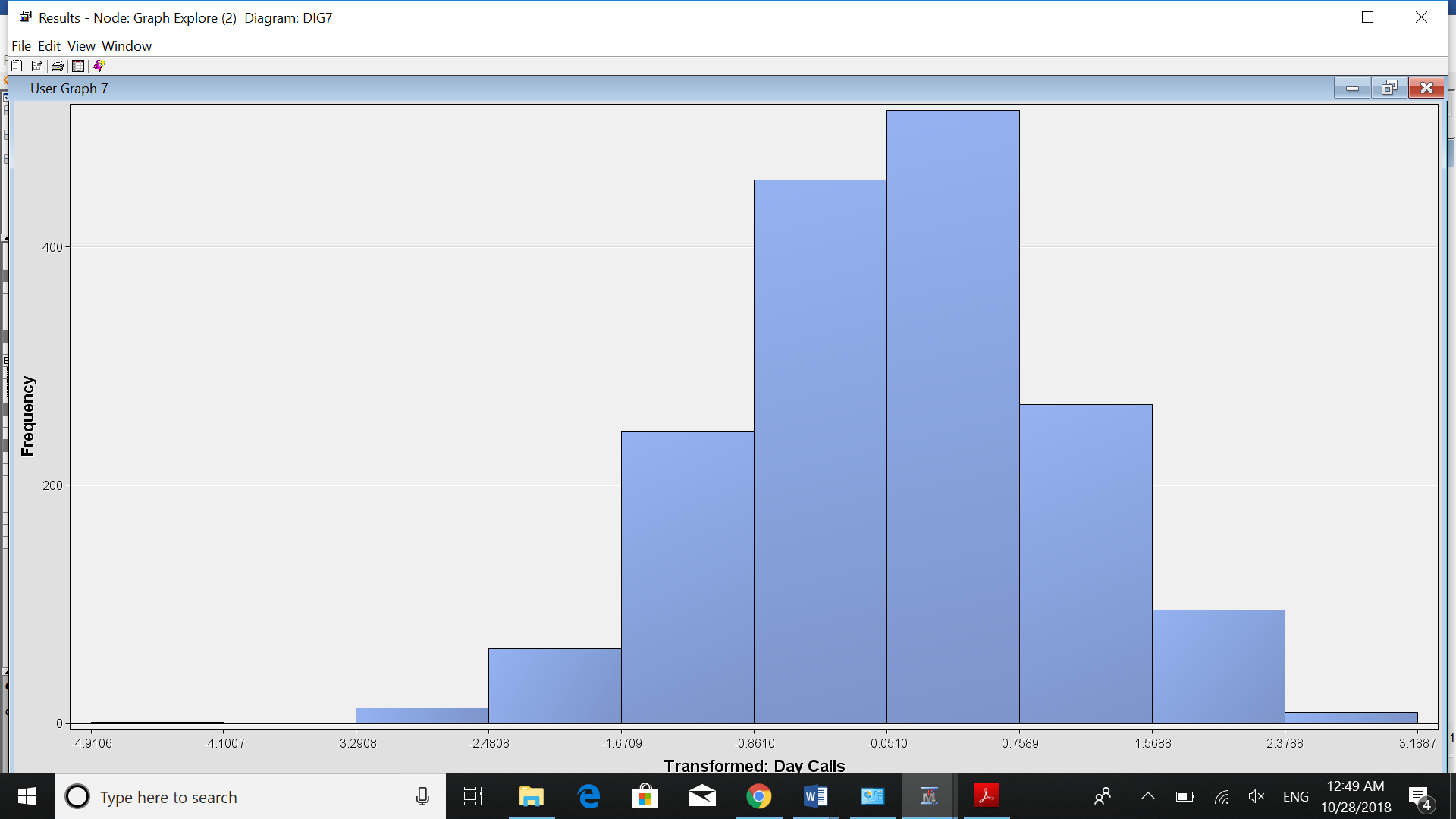


Then, we have to partition our data set as Training and Validation parts:

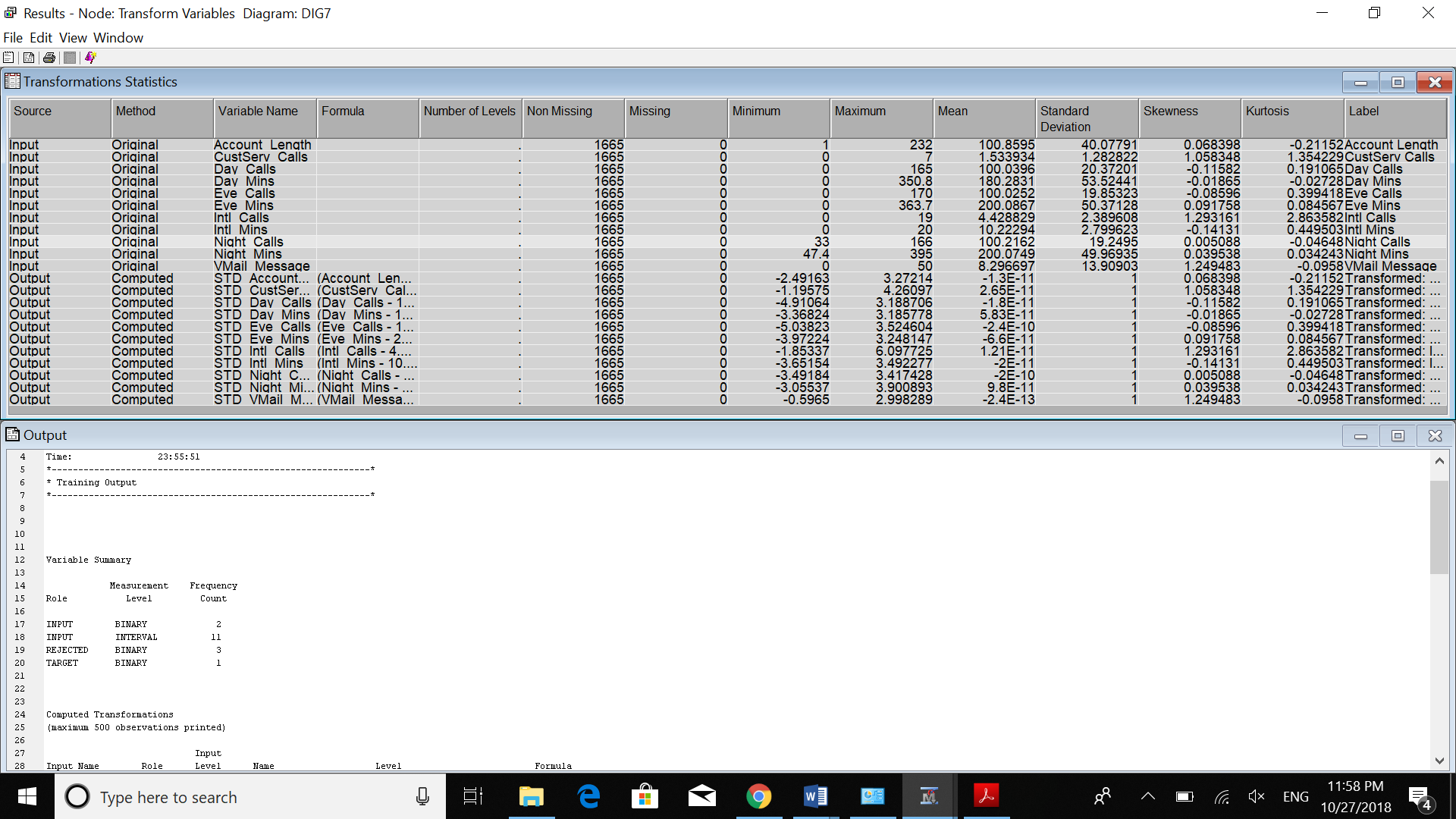


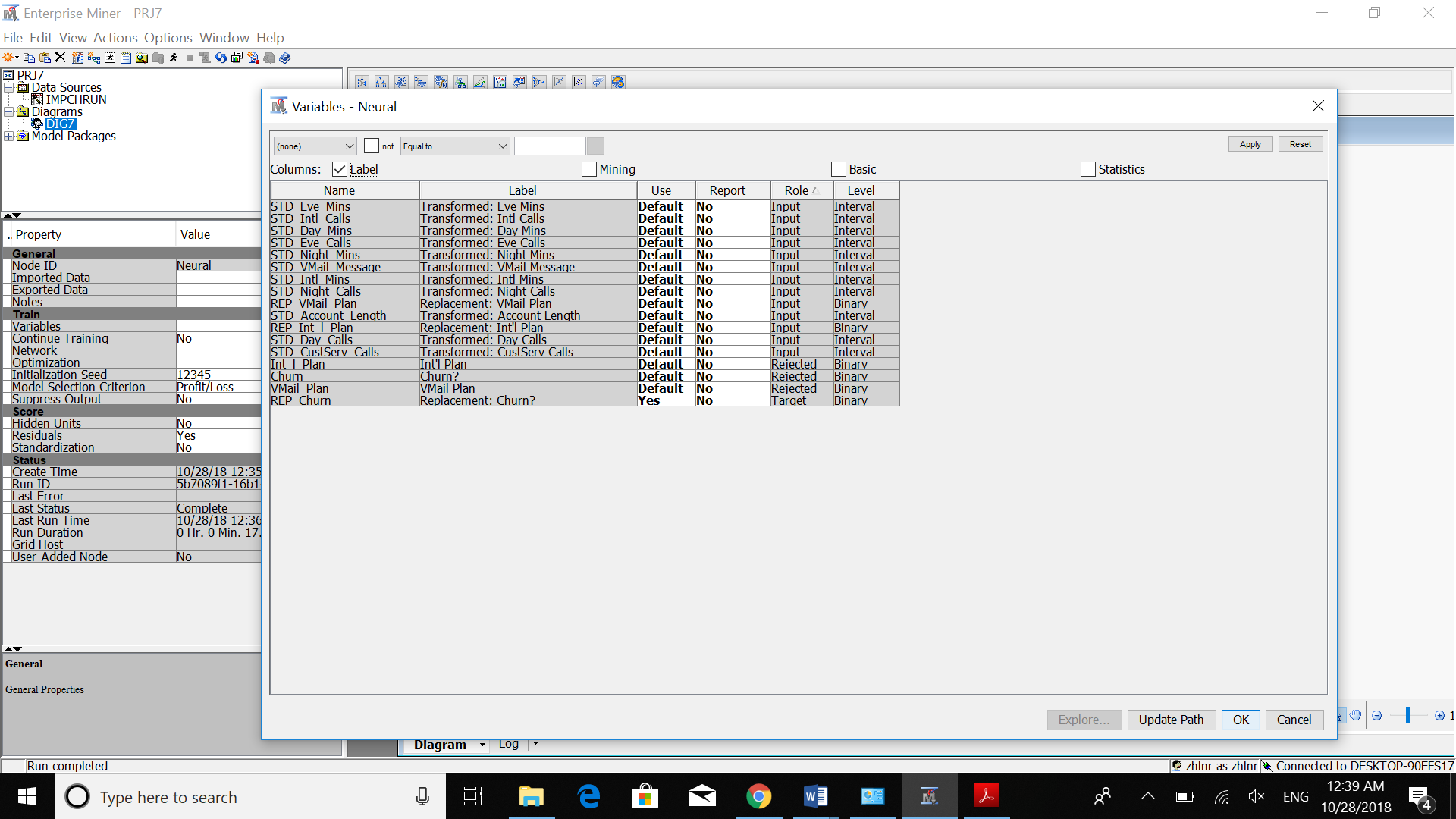
For the next step, I tried to standardized the variables in the data set. For example, we can see the day calls in below image. In what follows, we can see the difference in values of day charge in shown histogram before and after standardization. We did that because we want the variables be in the range of 0 and 1 by the Standard Derivation and mean value. So that the mean value is 0 and each of those bars is far by std=1.

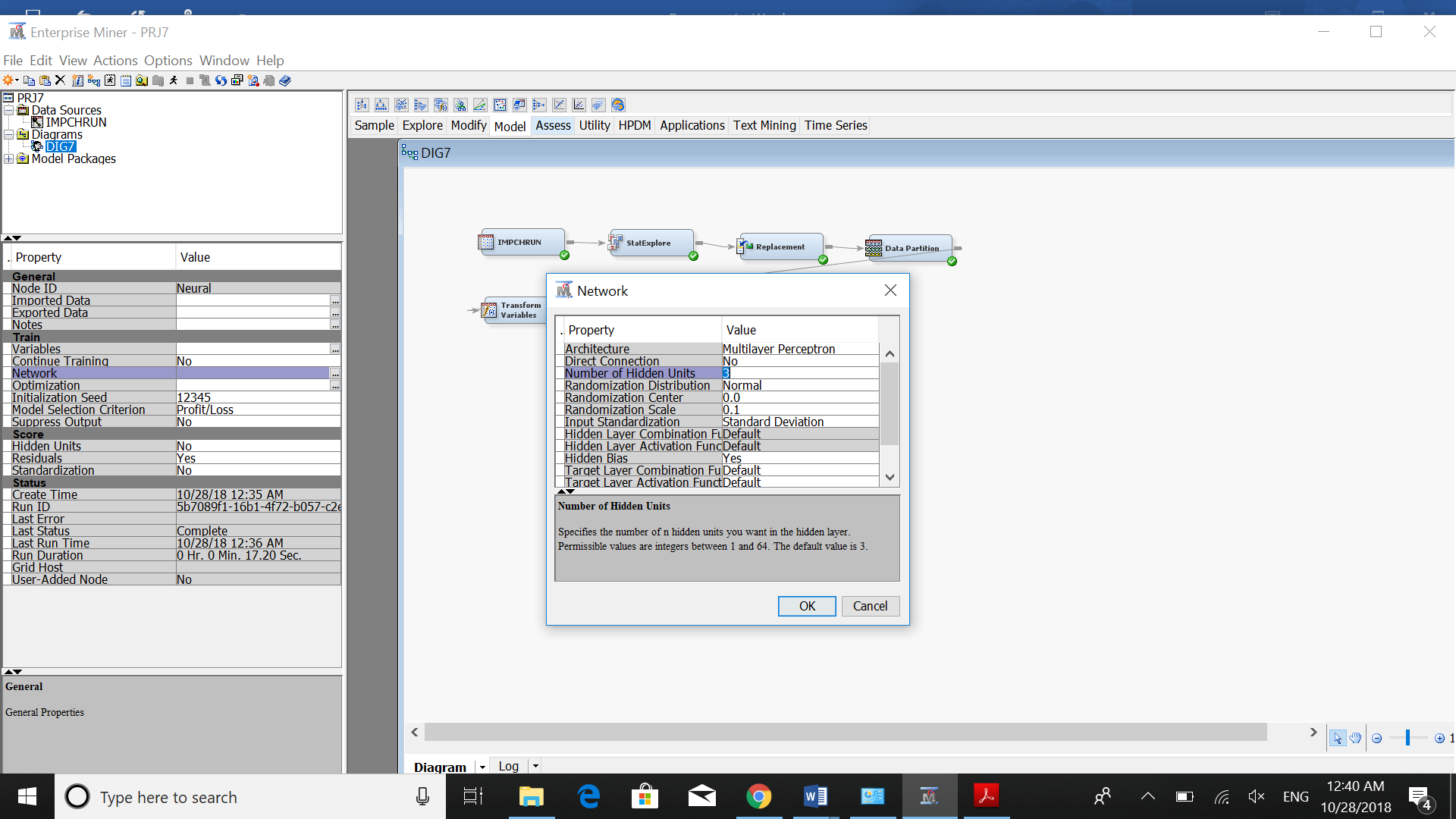


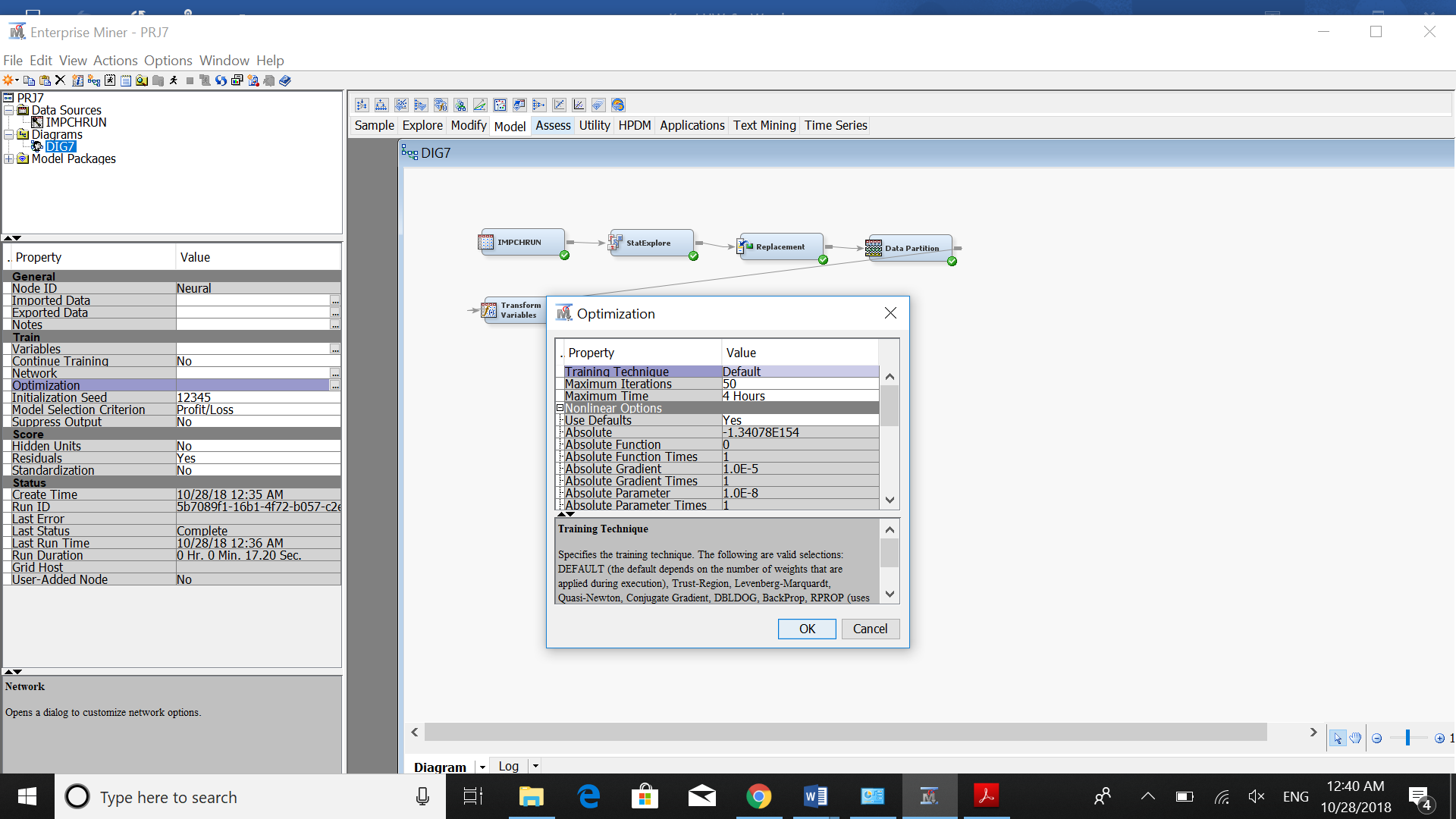


For the next step, I used the neural network node for creating the neural network.

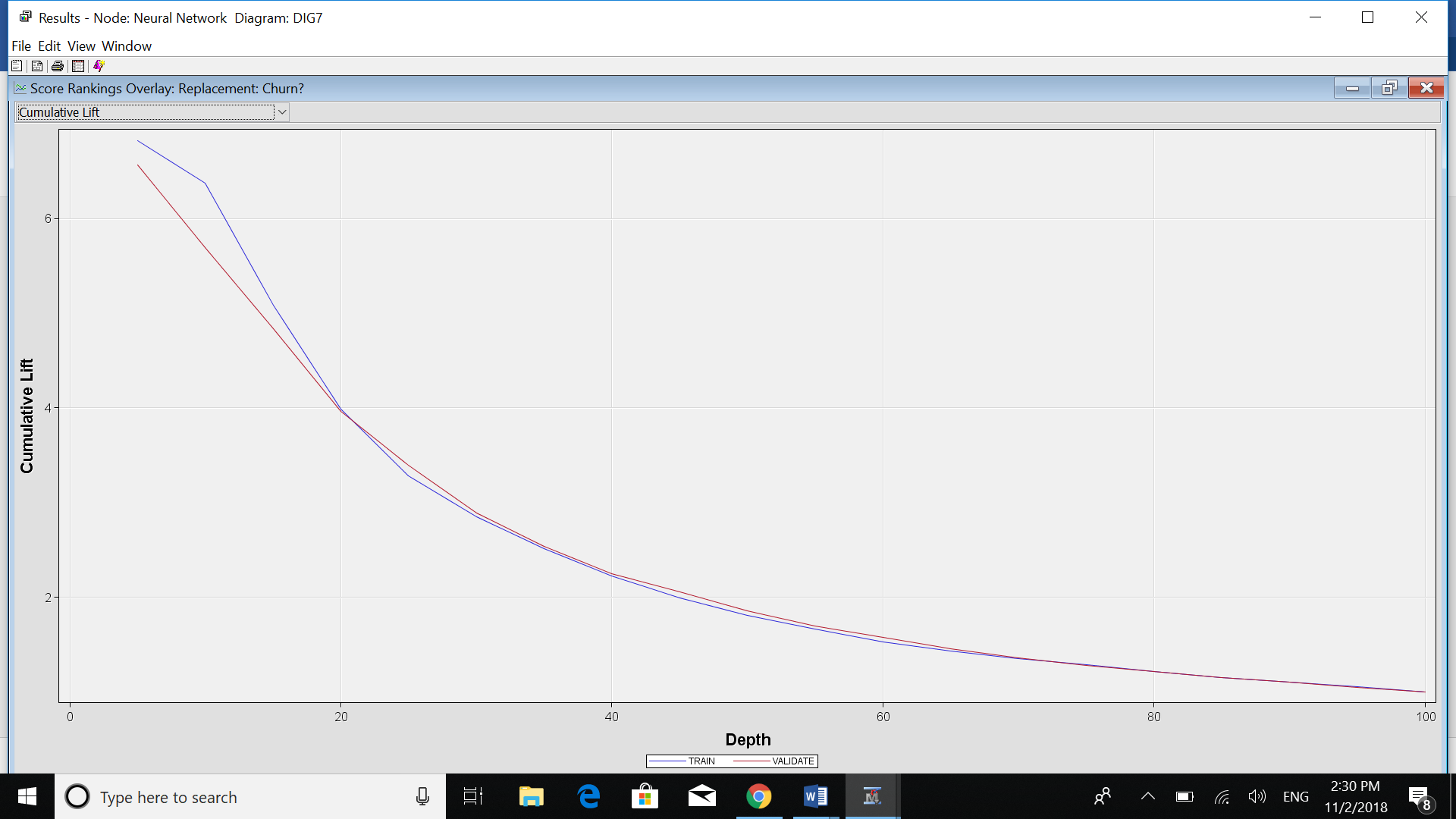




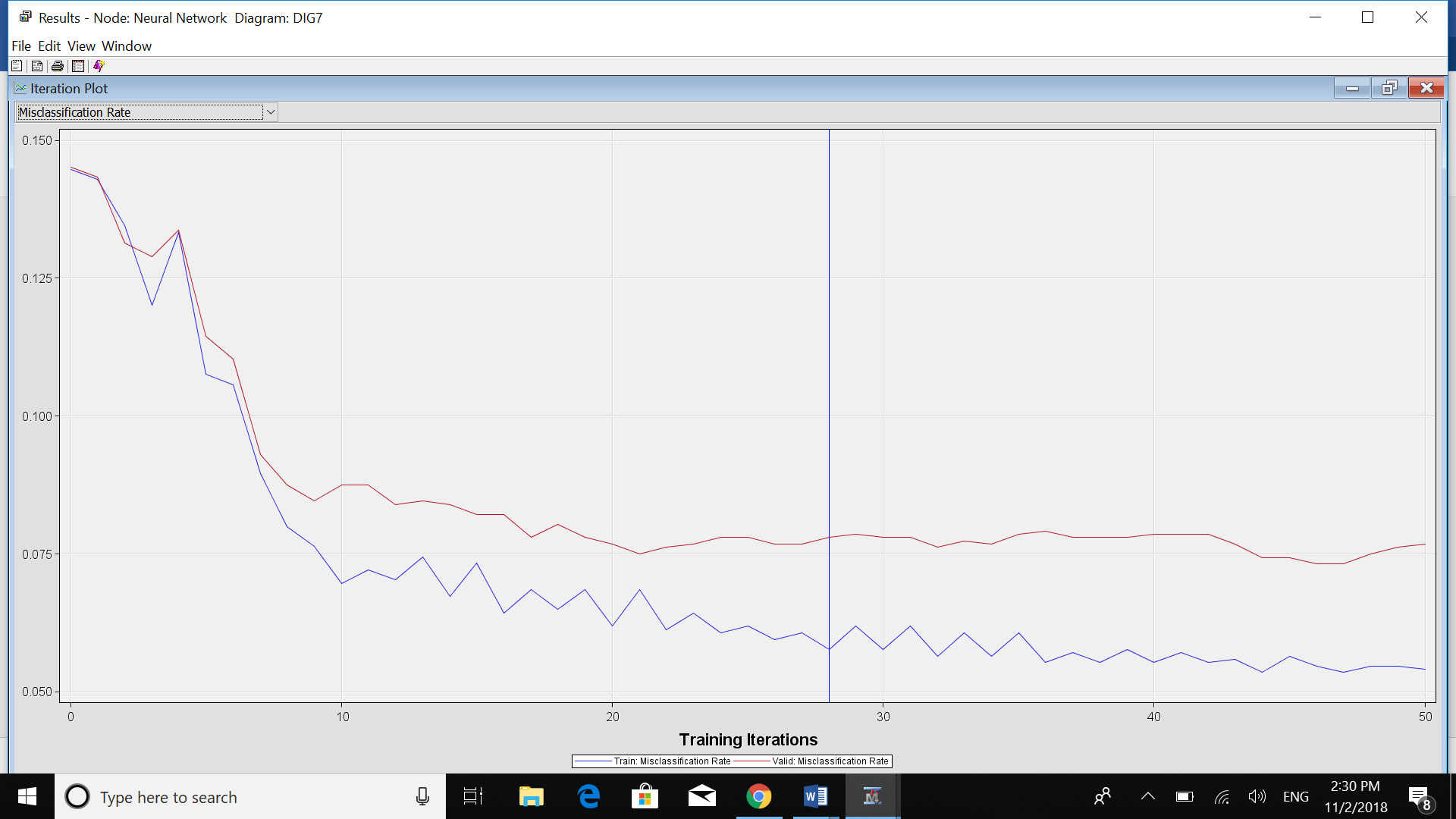




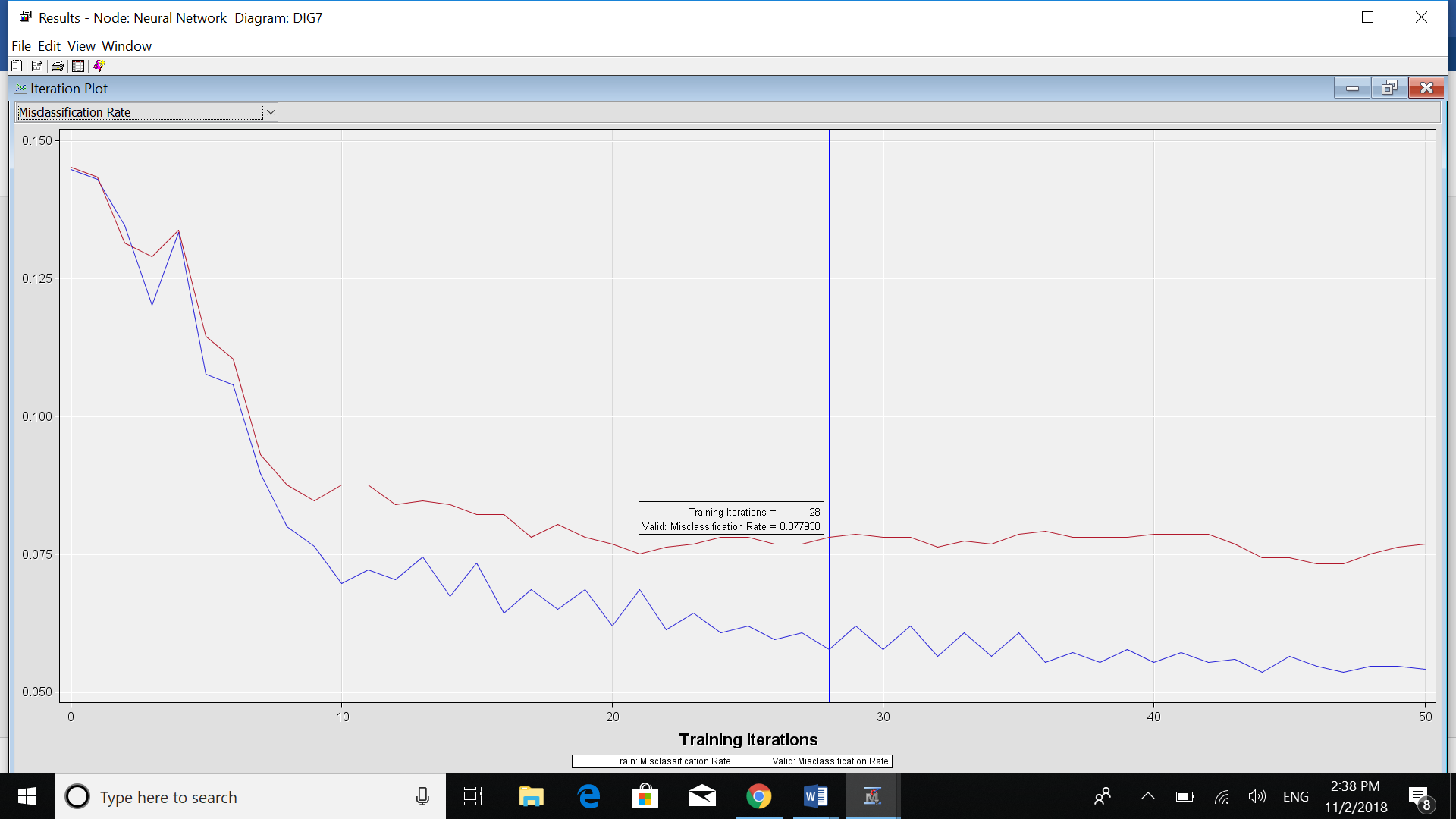
Here we can see the result:



For misclassification rate we have:



At iteration 28 we have:





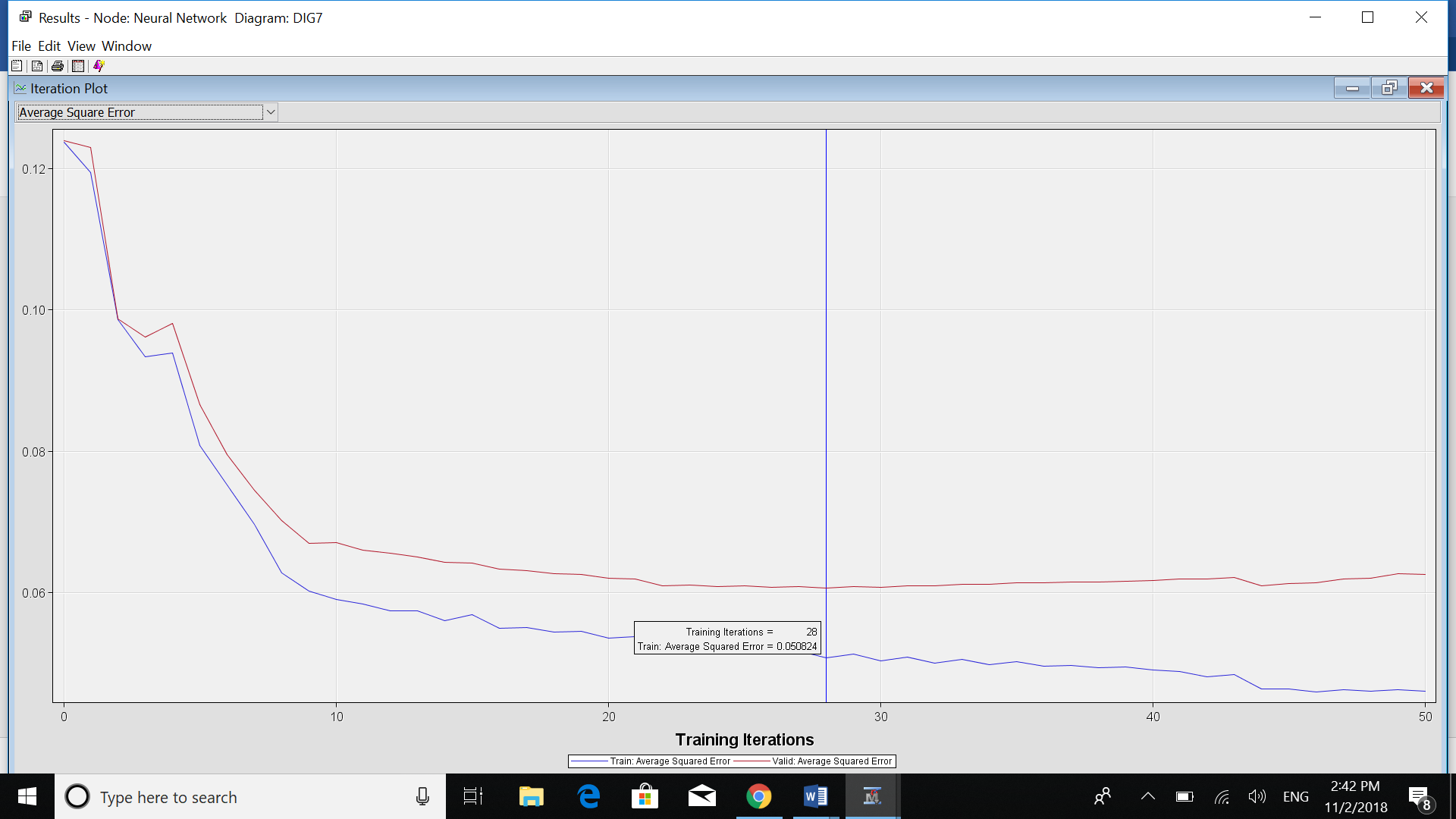
So the MR is 0.077 vs 0.057. So the difference between training and validation is about 0.02 which is reasonably small and acceptable for our model.

Also, for Average Square Error:



At iteration 28 we have:

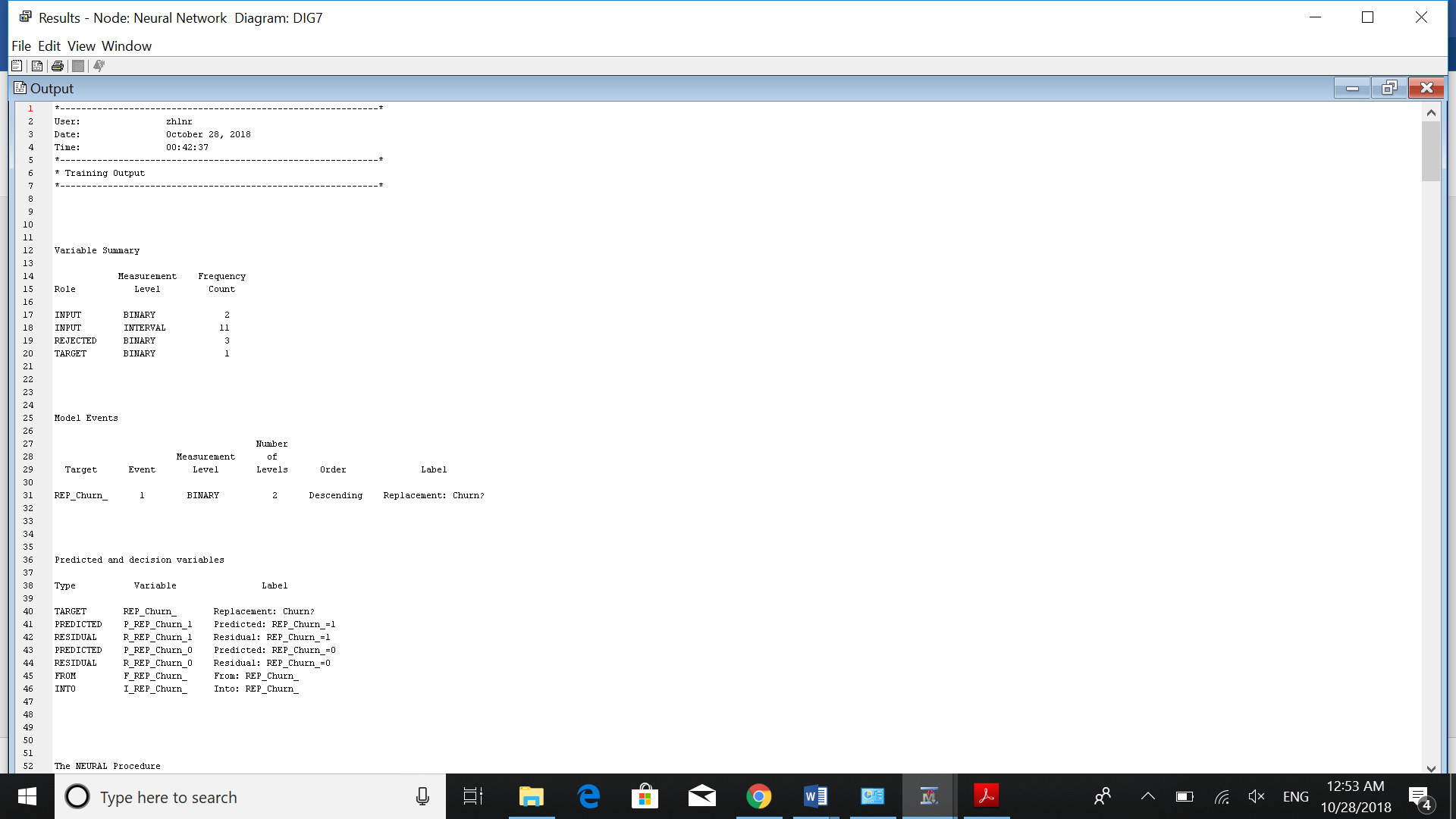




So the ASR is 0.060 vs 0.050. So the difference between training and validation is about 0.01 which is reasonably small and acceptable for our model.

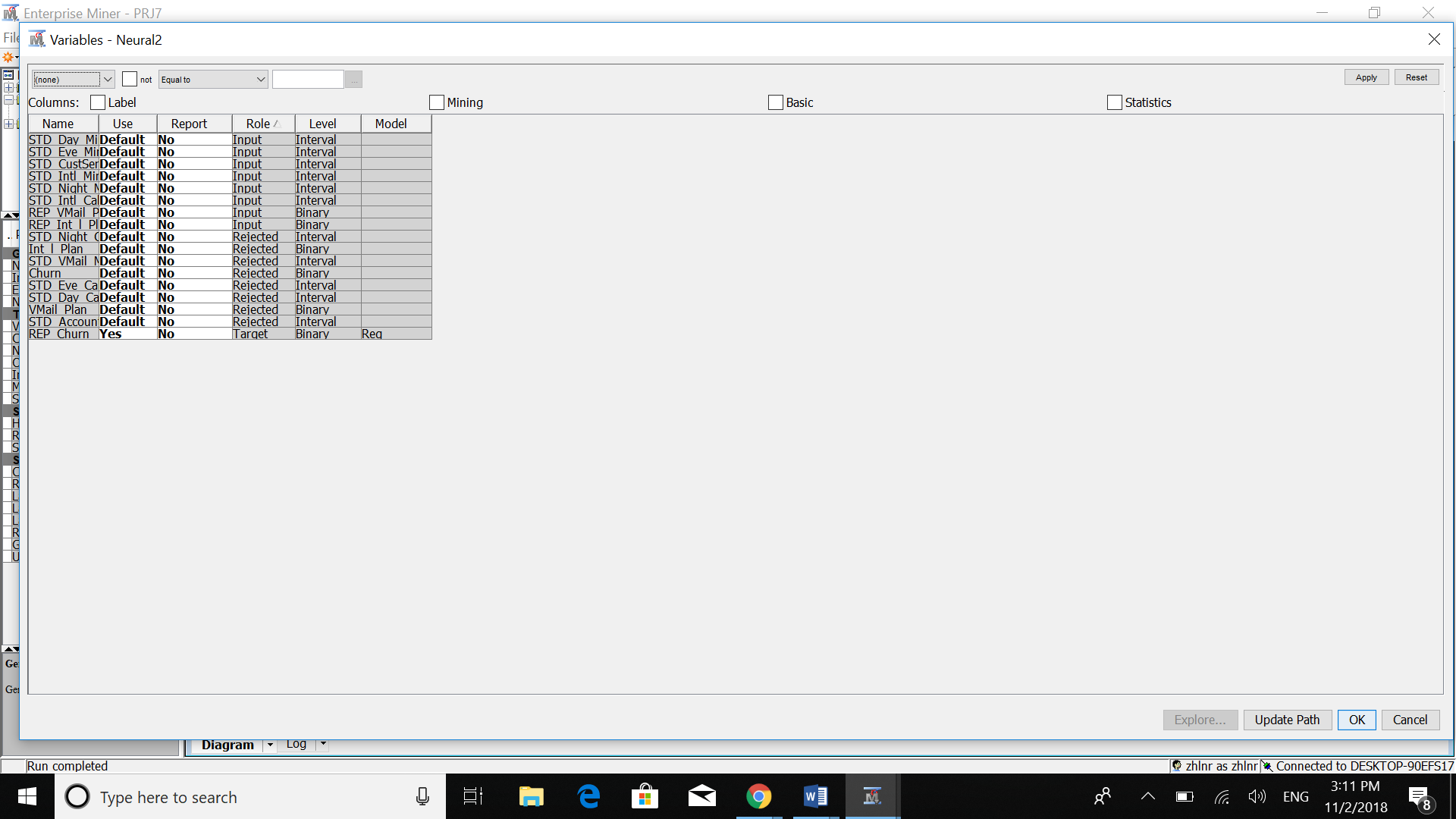
We can see the difference between training data vs validation data in terms of Misclassification Rate and Average Square Error. At first, they are both starting at the same exact amount as we haven’t done any iterations yet. We can see the optimal point here is the iteration 28. It means that before 28 iterations we may have underfitted model and after 28 the model is too complicated that we may encounter an overfitted model.

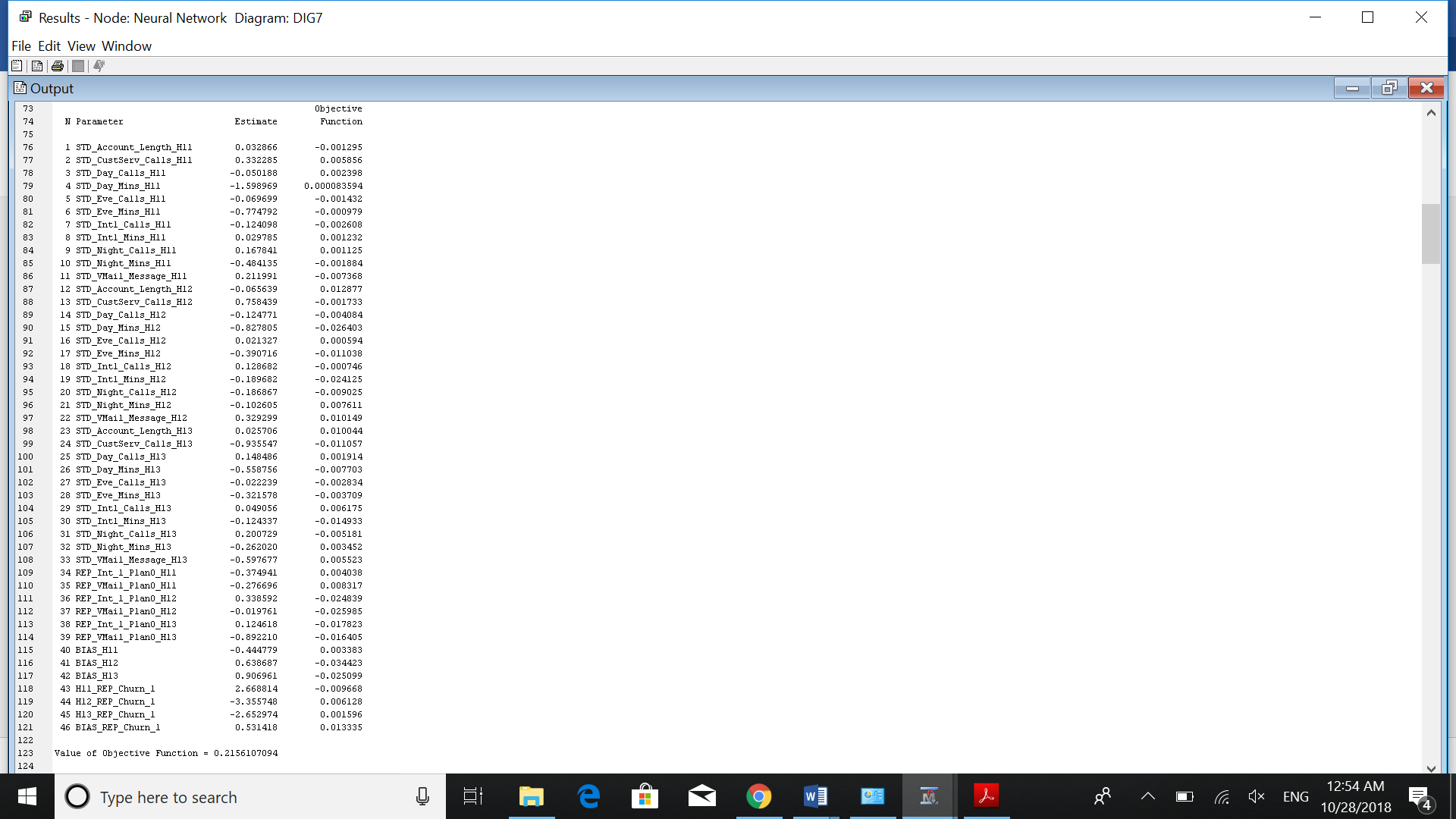
Here we can see that 13 input variables were used for this model:



In this image, we can see that we have 3 neurons in hidden layer as H11, H12 and H13. Also, we have 3 layers: input, hidden, output layers.

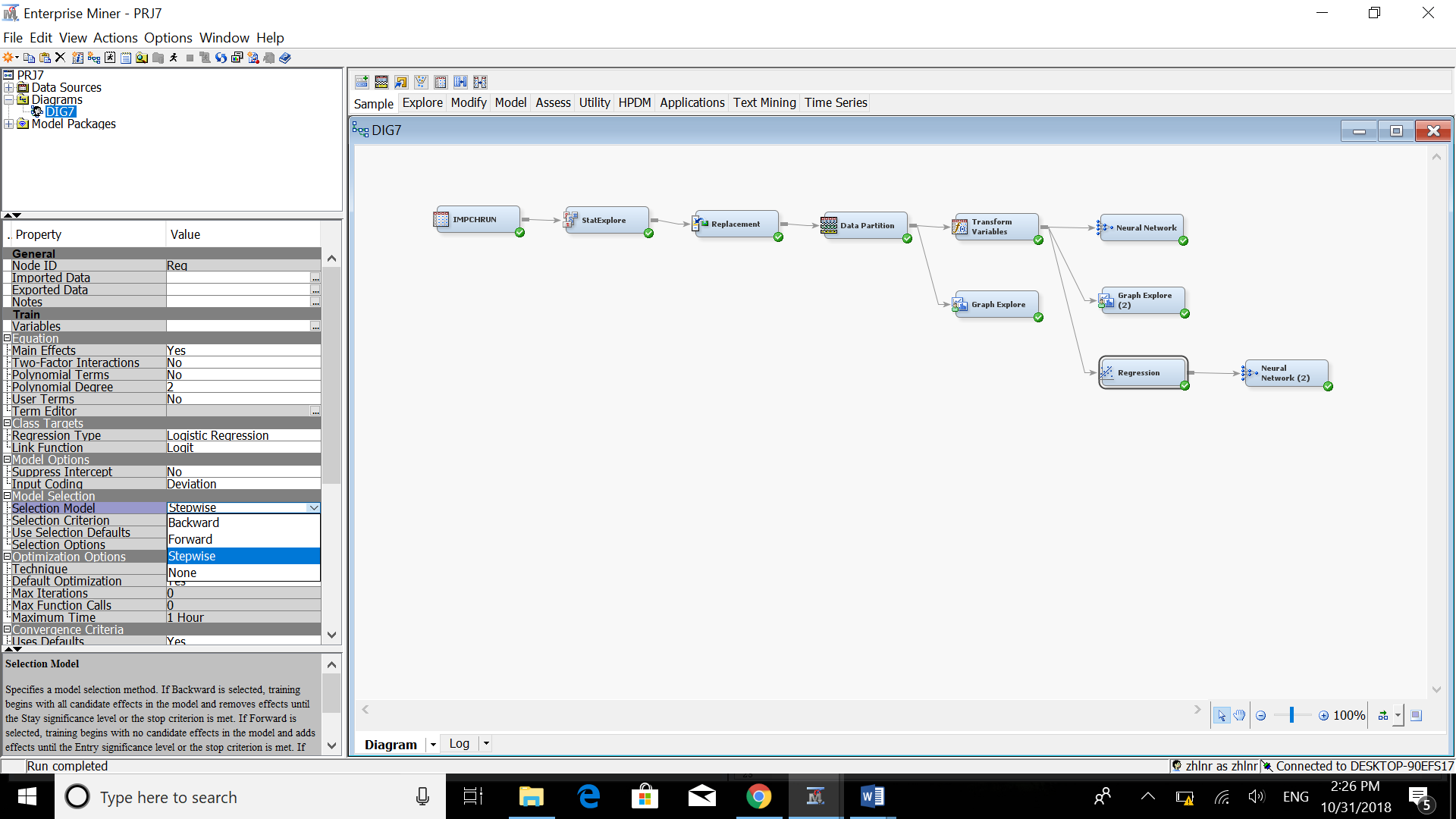
Here we can see all of the variables sorted by their roles:





In this step, I add one regression node and tried to create the neural network node after regression node and compare the result to neural network without regression.

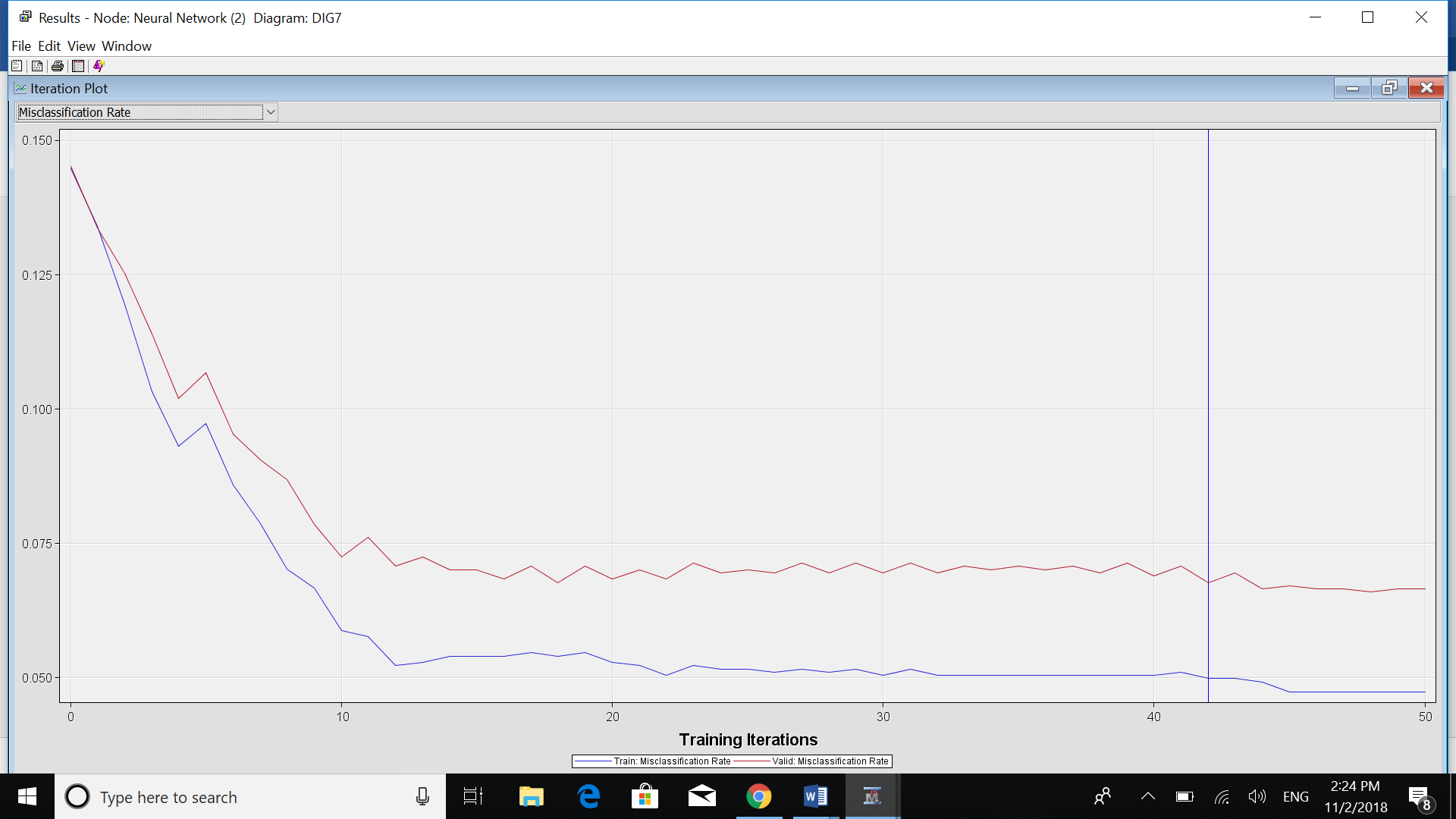
First, we should change the selection model of Regression node to stepwise.

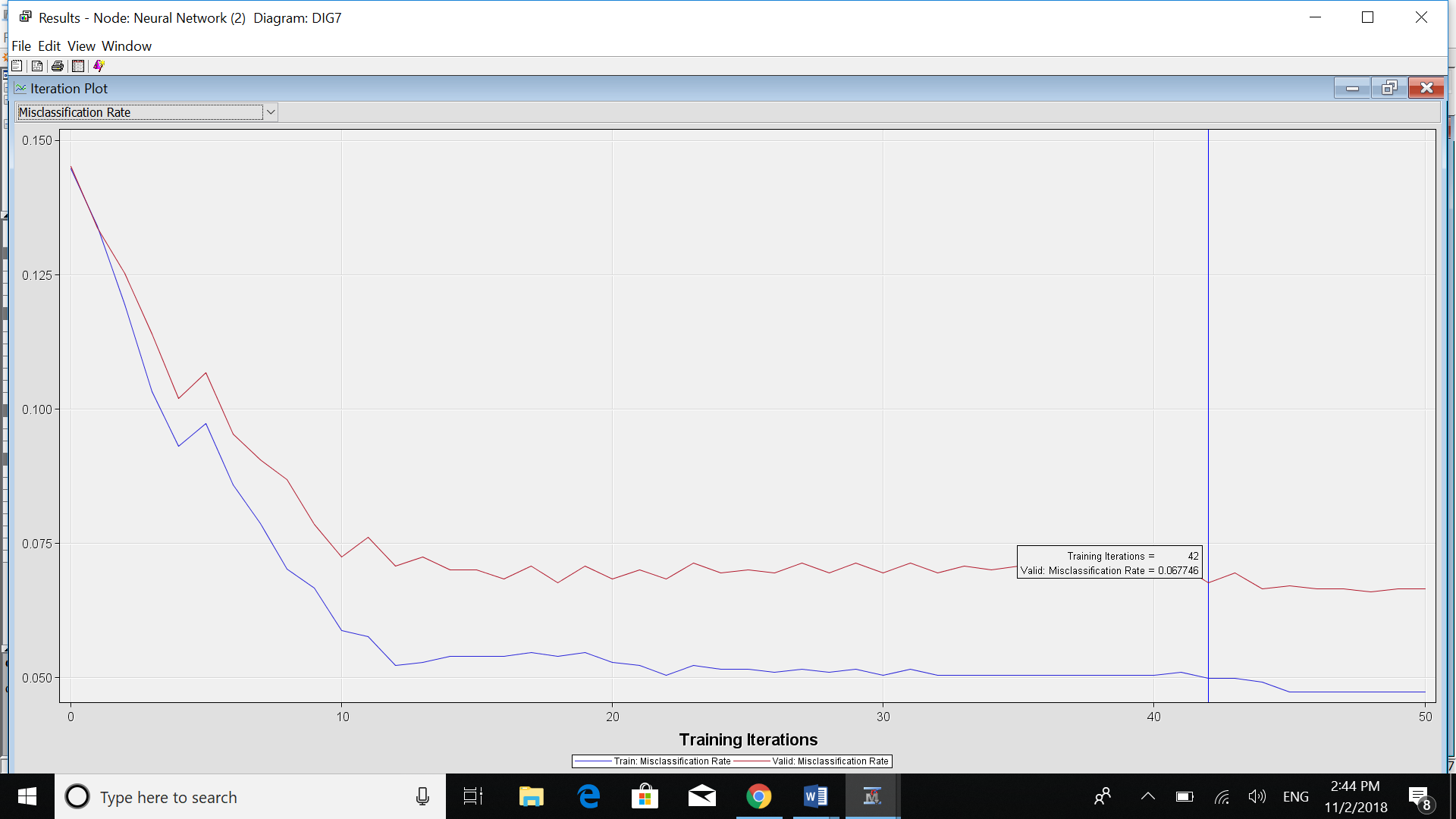


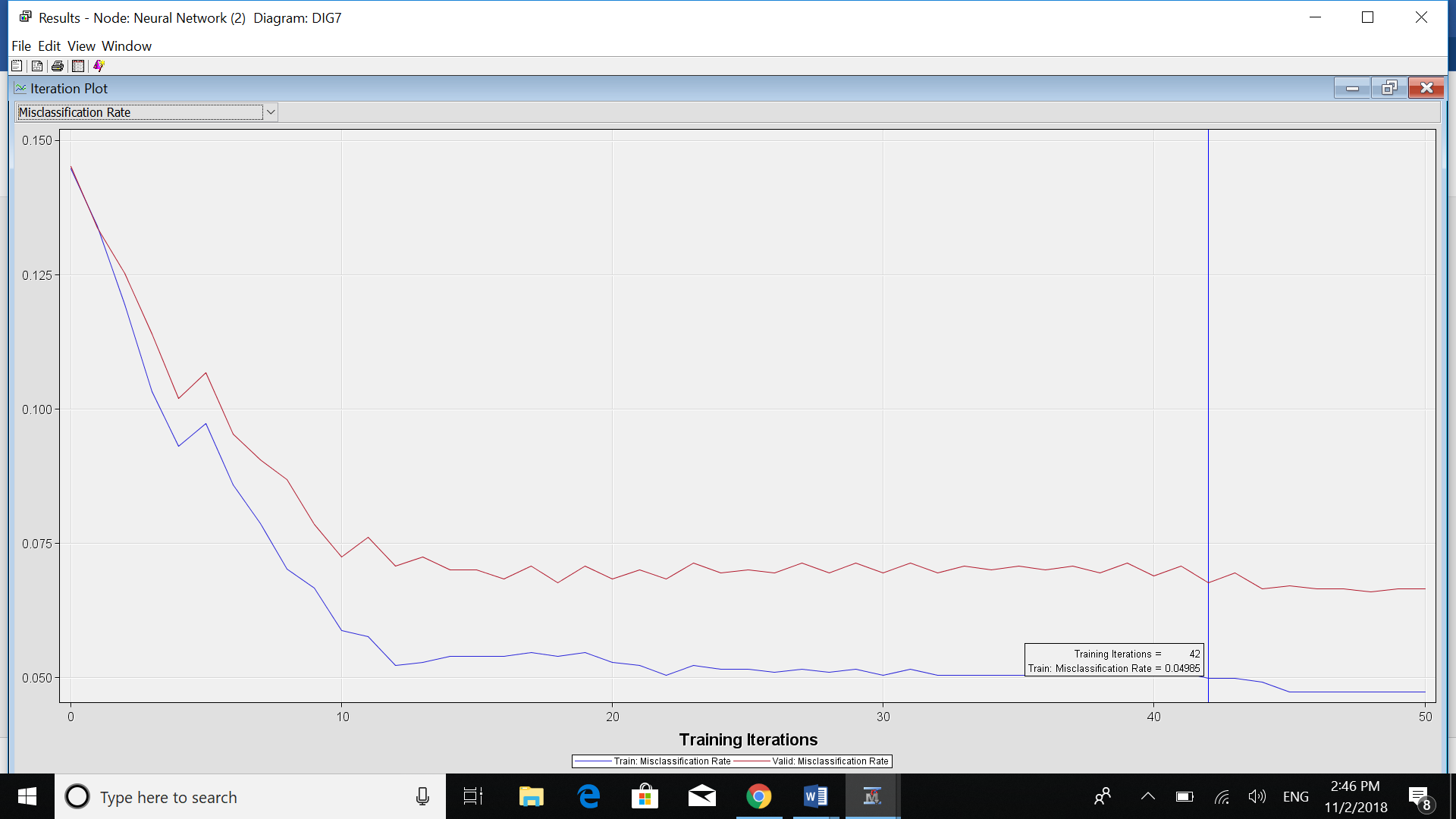
After running the neural network node after regression we have:



For the misclassification rate we have:

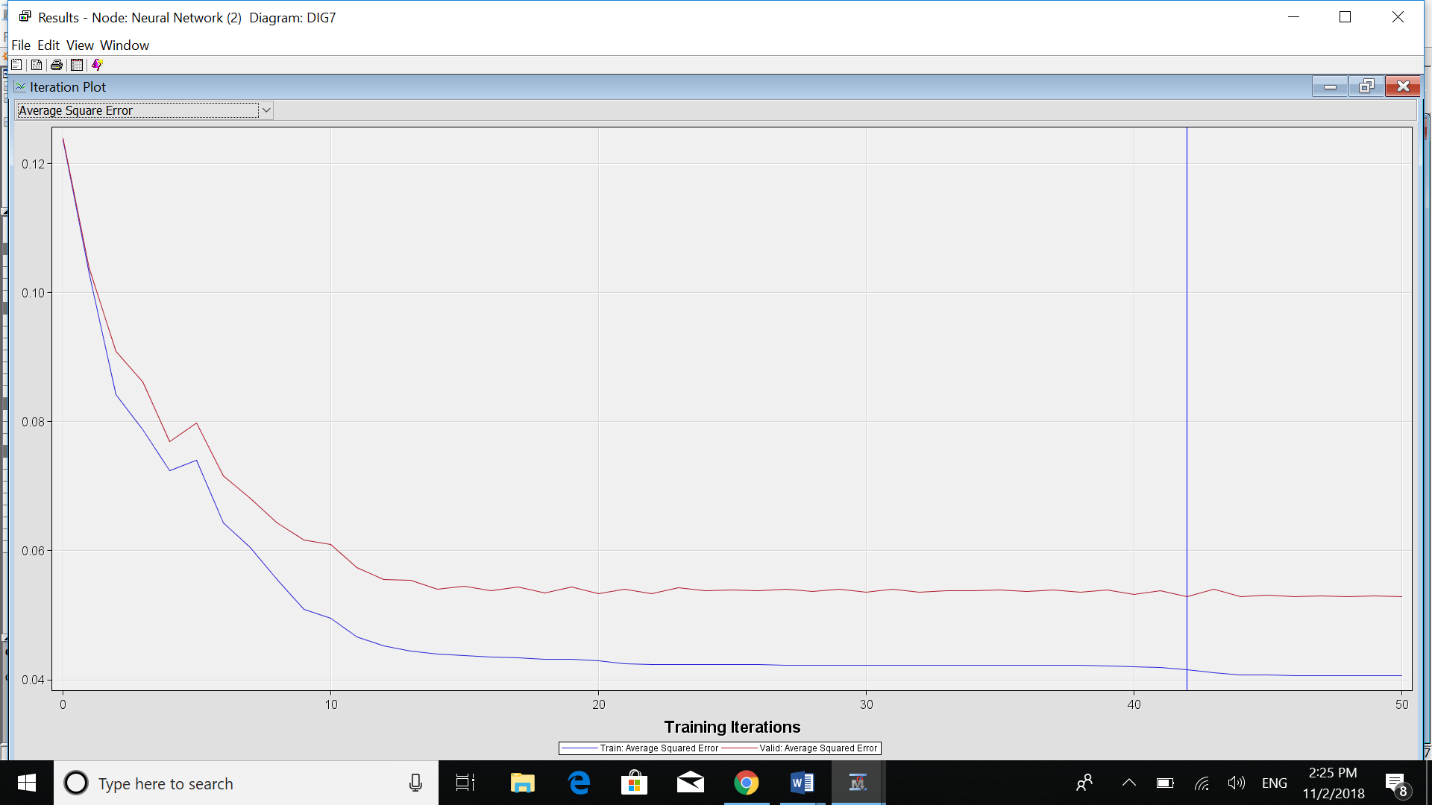


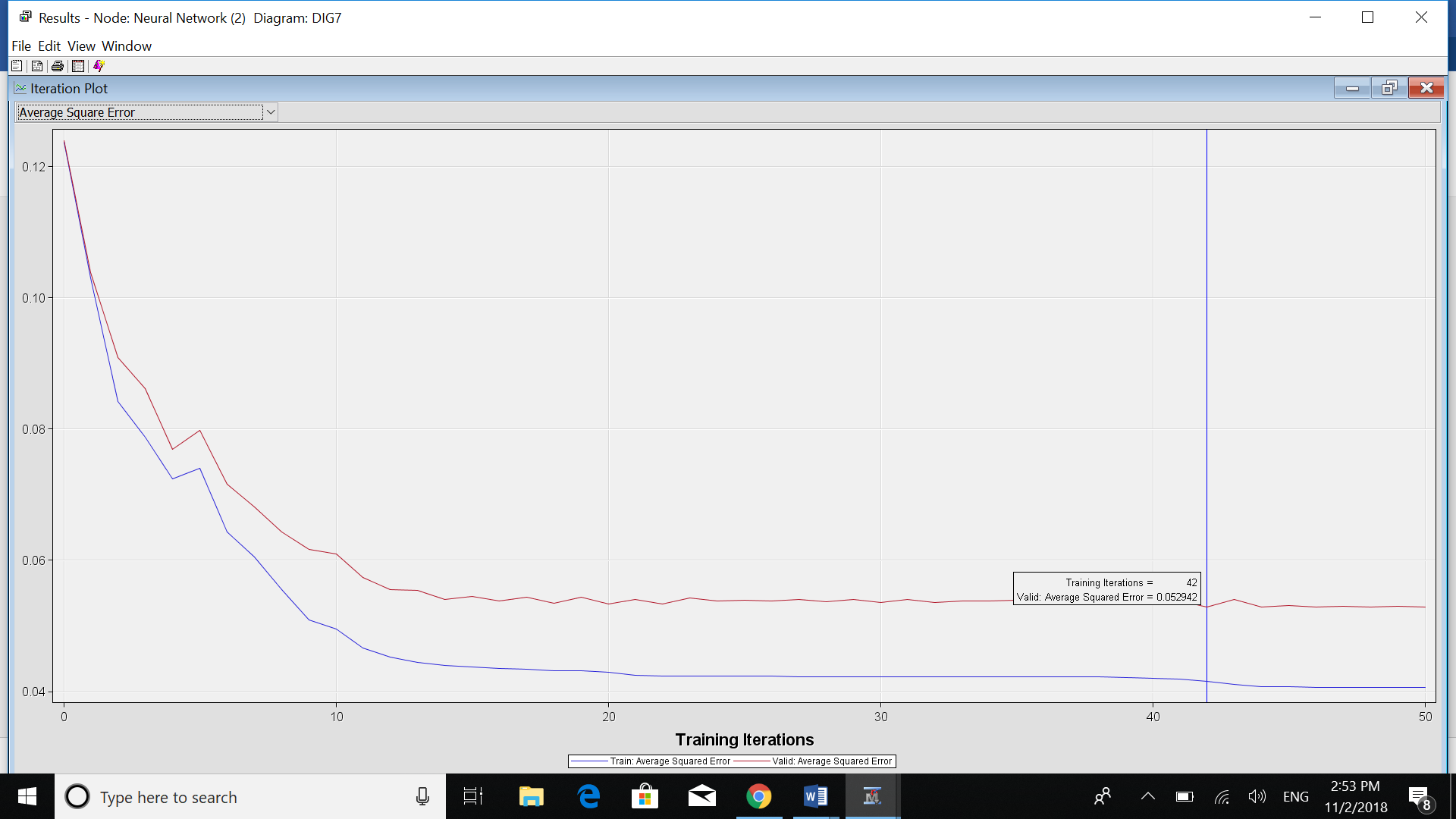


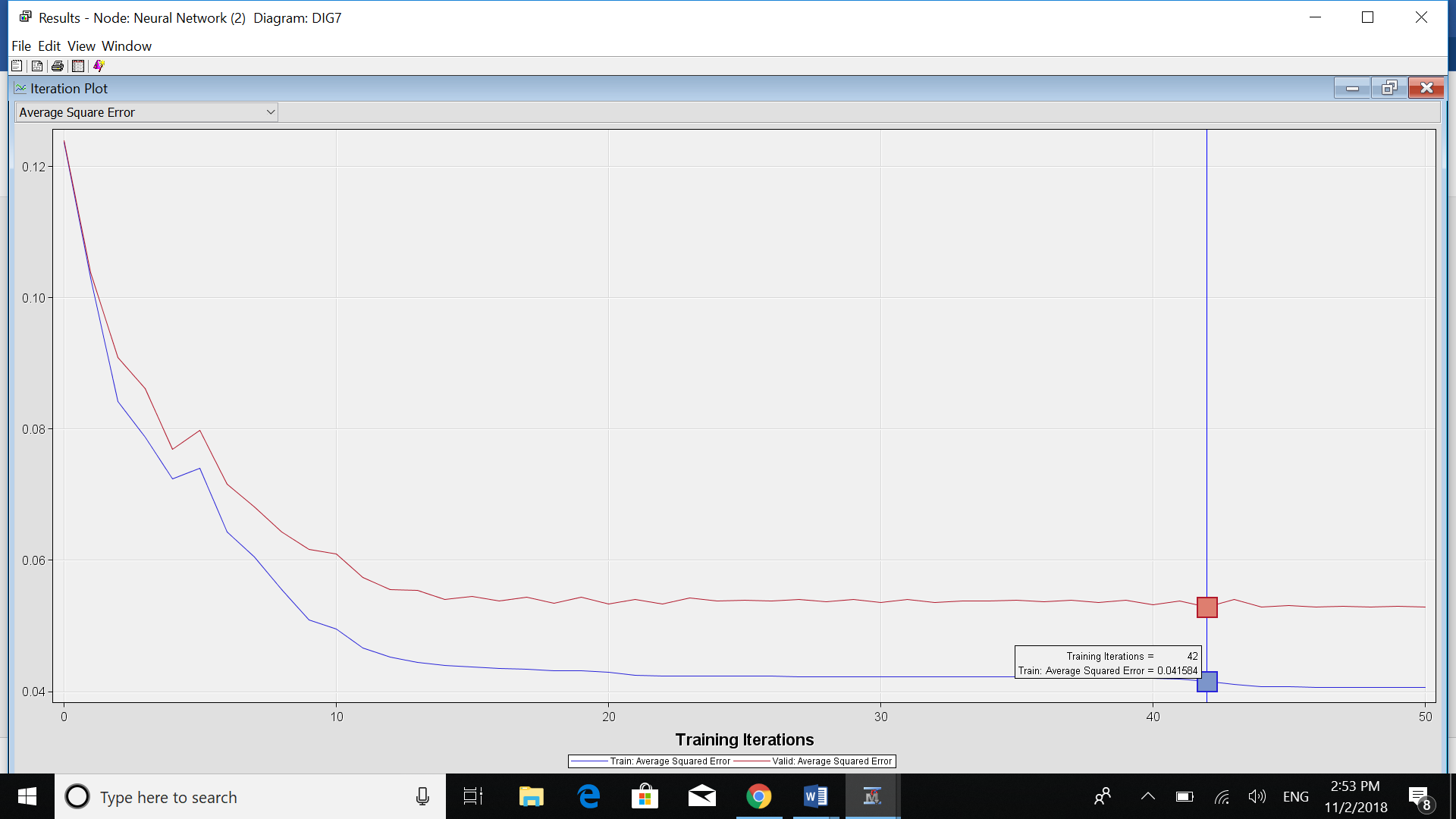


So the MR is 0.067 vs 0.049. So the difference between training and validation is about 0.018 which is reasonably small and acceptable for our model.

Also, for the Average Square Error:



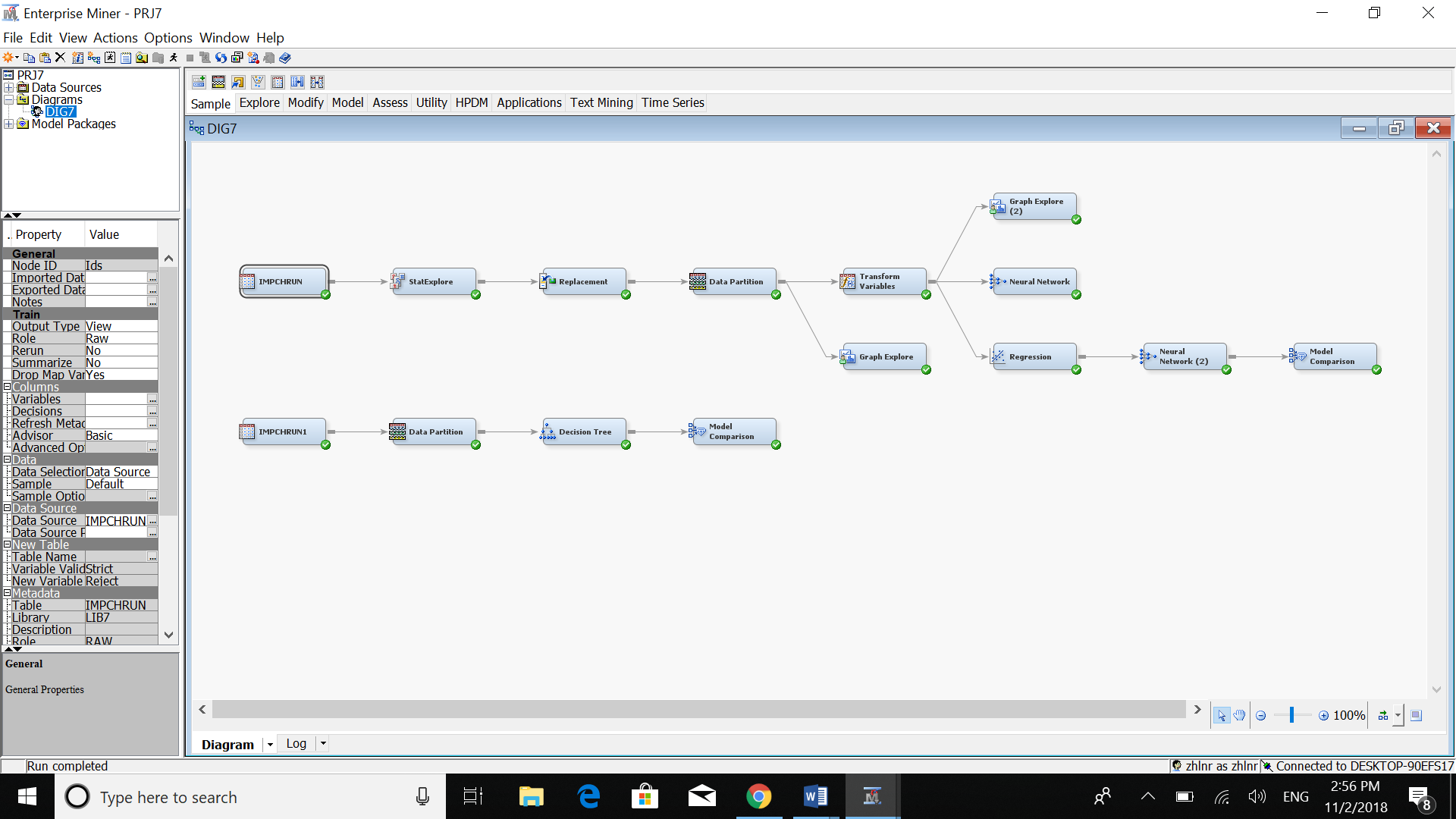




So the ASR is 0.052 vs 0.041. So the difference between training and validation is about 0.011 which is reasonably small and acceptable for our model.

We can see the difference between training data vs validation data in terms of Misclassification Rate and Average Square Error. At first, they are both starting at the same exact amount as we haven’t done any iterations yet. The optimal point for NN after Regression is 42. So after that number of iterations we may have an overfitted model. Similarly, before the 42 iterations we may encounter to an underfitted model.

Here is my work flow at the end:

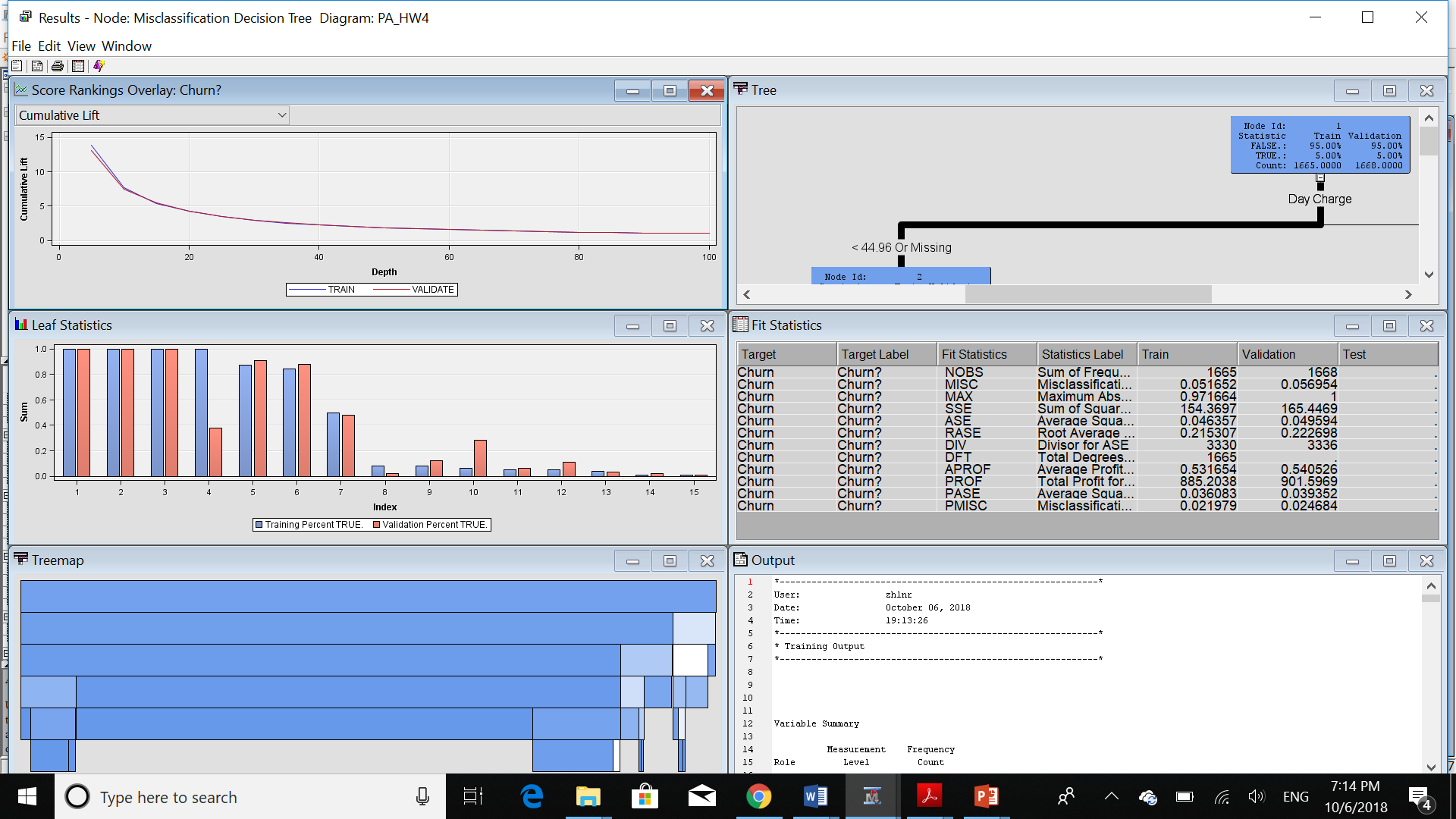


So, we can see that Neural Network model after regression is a better model in terms of Misclassification and Average Square error rates.

14-Compare the neural network with best decision tree.

First, let’s run the Best Decision Tree on our data set again.

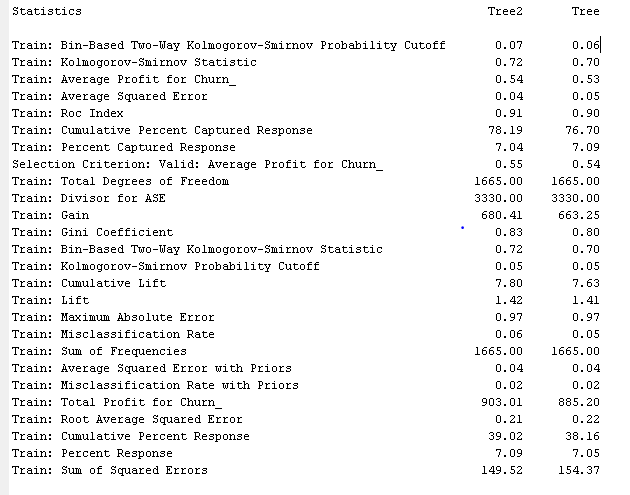
Here is the result of running the best decision tree on the data set:



We can see the number of leaves vs misclassification rate:

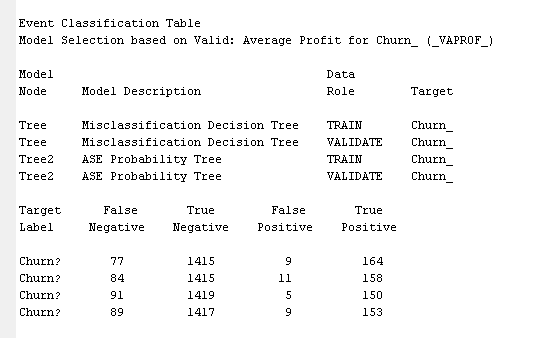


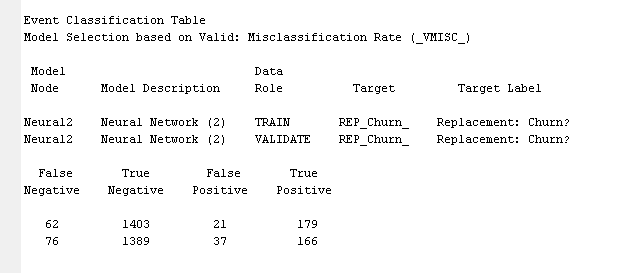
In terms of comparing NN and DT based on their Average Square Error and Misclassification Rate we have:



So, the Misclassification Rate is 0.06 for DT while it was 0.049 for NN. Also, the Average Square Error is 0.04 for DT while it was 0.041. So we can conclude that, in terms of Average Square Error they are doing almost the same. But for Misclassification Rate, NN did a bit better. But both numbers are reasonably acceptable.

In terms of comparing True Positive, True Negative, False Positive and False Negative we have these two charts for DT and NN:





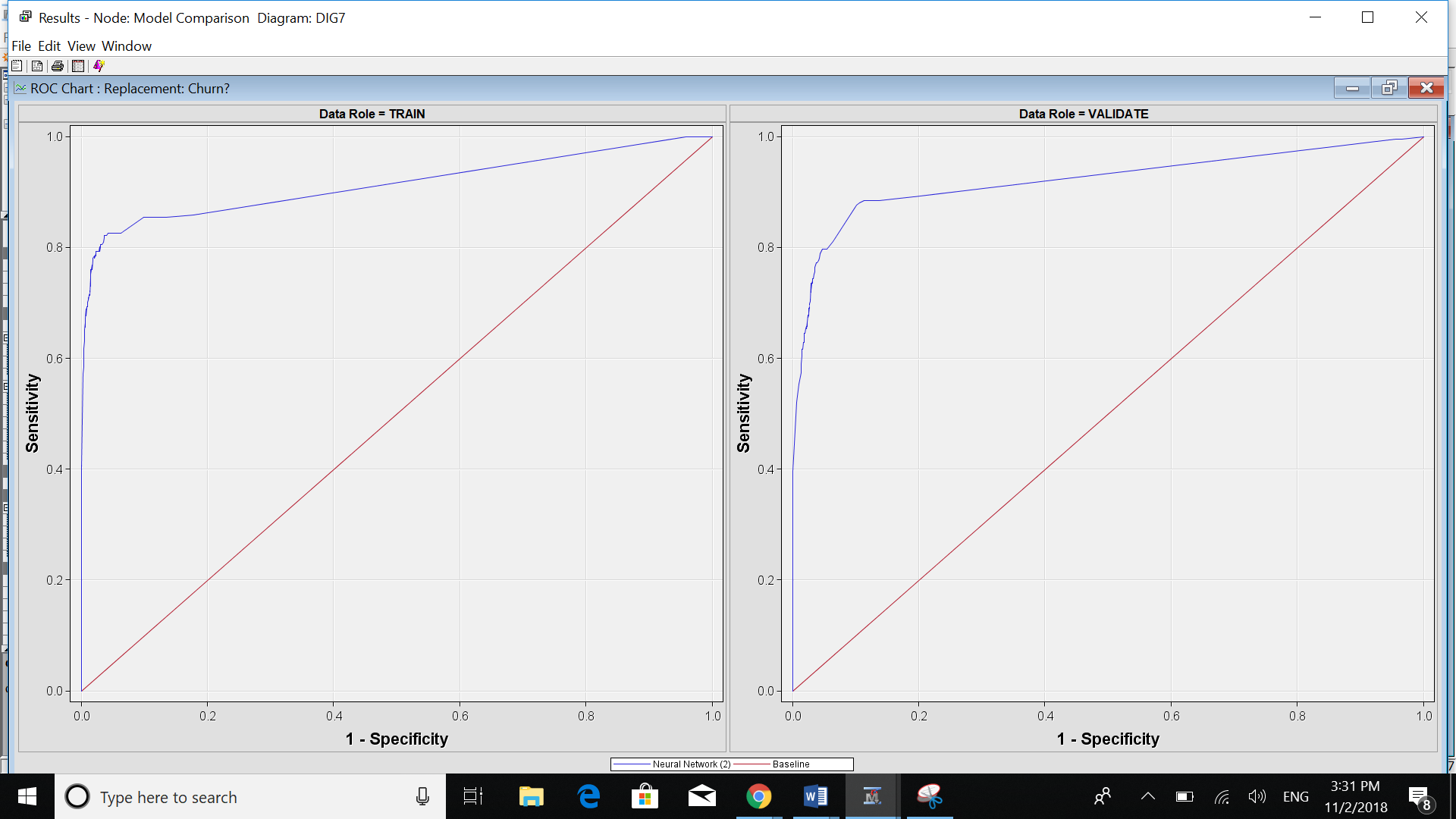
So, the value of False Negative is smaller in NN compared to DT. But at the same time, the value of False Positive is higher than DD. In this case, False Negative is costlier for us. As we may miss that specific group of customers. So, in this case NN did a bit better.

In term of Model Complexity, we know that Neural Network is more complex model. It needs further understanding for interpreting the result. But at the same time, Decision Tree is less complicated. We can illustrate the result very easier in compare to NN. Also, here out Neural Network model’s optimal iteration is 41 while for DT is just 15. As a result, we can see that our Neural Network model is way more complicated in compare to Decision Tree.

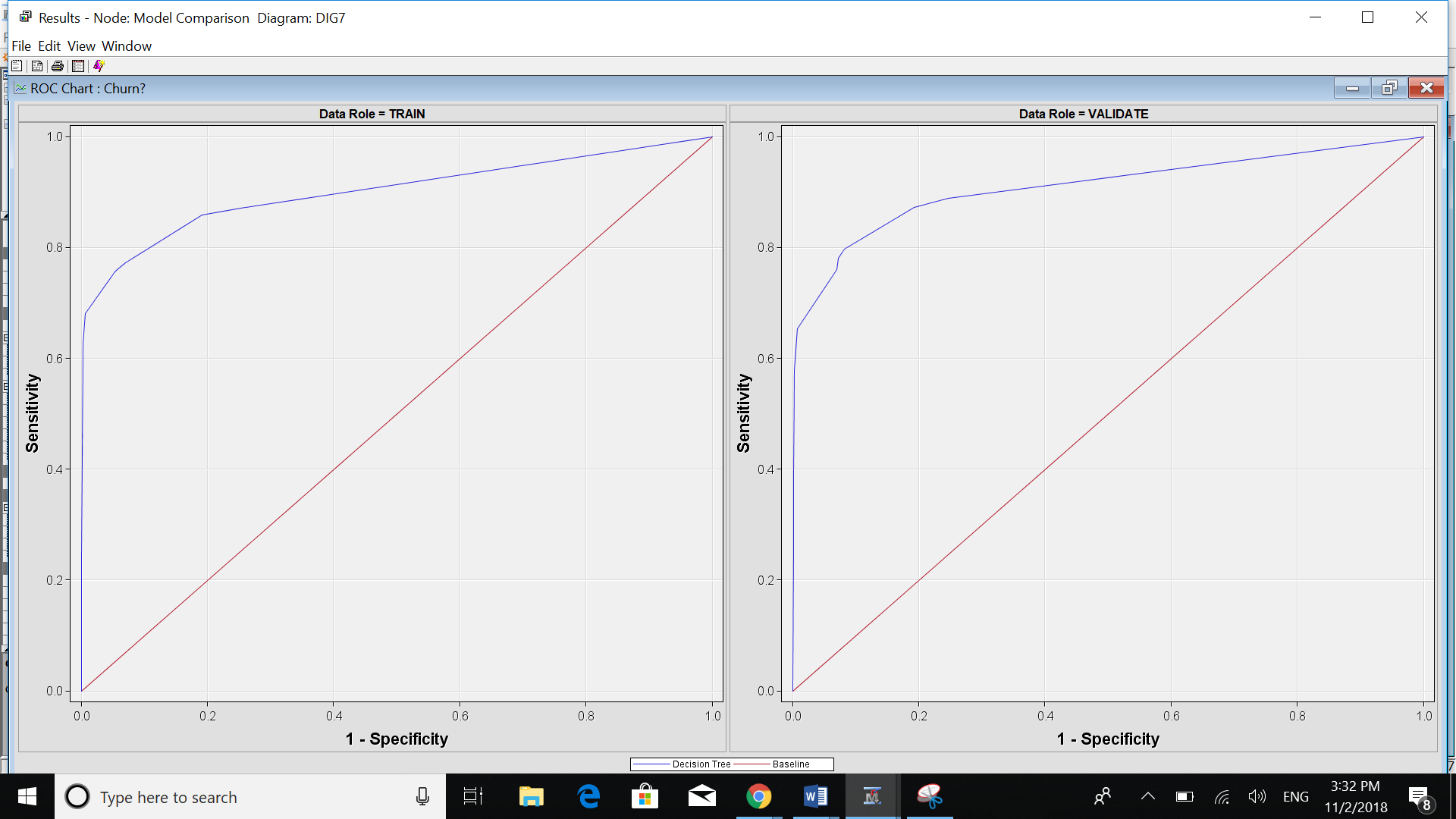
For the next step, I tried to run model comparison node for both NN and DT. But the problem I face to was the fact that we need interval data for NN while we have categorical data for DT. So, I used same Data Set with different Roles and Levels for variables. So, I couldn’t run the model comparison for these two models as they are using two different data sets.

Then, I tried to run model comparison for each of these models and see the result.

We have the ROC chart for NN as follows:



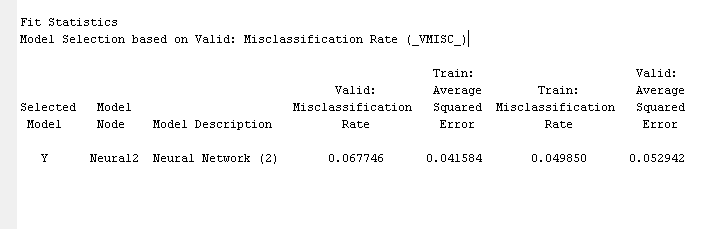
ROC chart for DT:



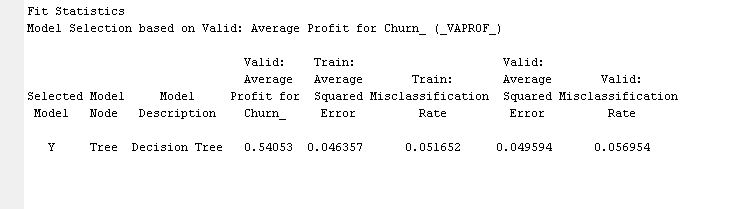
We can see DT ROC is pretty close to NN. But because DT is closer to the red line which is not good, it makes NN the better model in terms of Sensitivity and Specificity. But we should pay attention to the point that they are so close to each other.

Model Selection based on Misclassification Rate:

For NN we have:



For DT we have:



Here, the values are very close to each other but for NN the misclassification rate for training is a little bit smaller than DT.

The last point I would like to mention here is that Neural Network works with the interval variables while DT works with both categorical and interval variables which makes me a more flexible model.