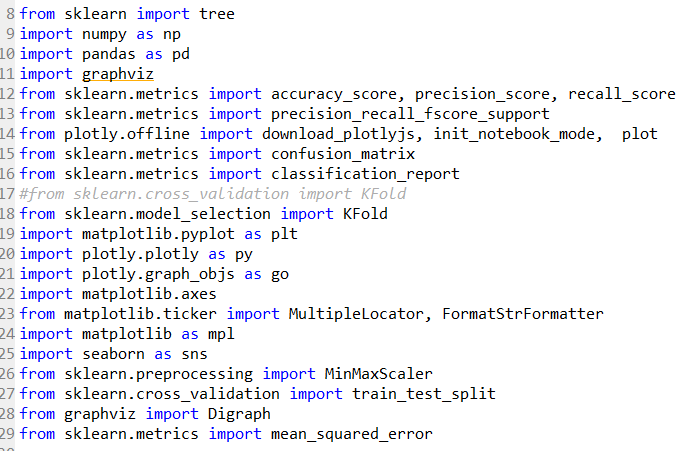
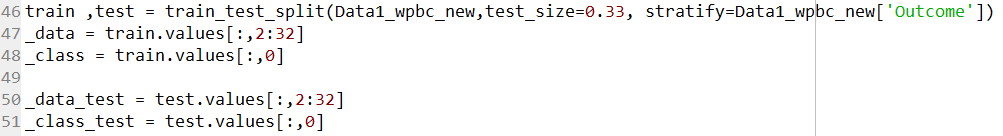
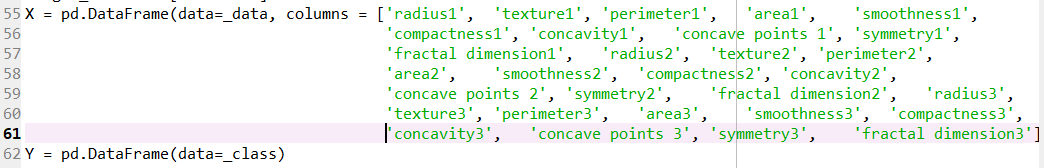
**Regression Analysis - Report**

Import necessary packages.

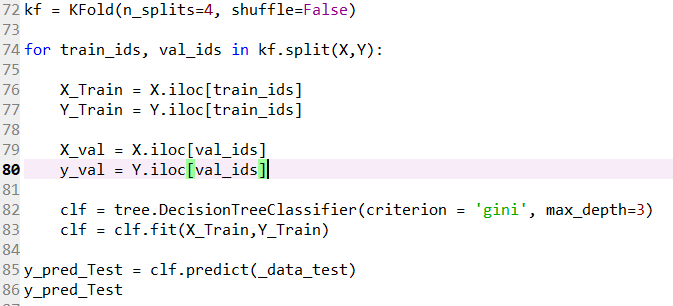


Split the dataset into train and test. Also, separate the features from the target class.

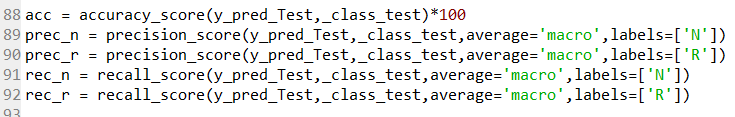




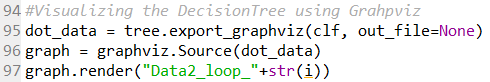
Perform 4-fold cross validation. Train the dataset



Calculate performance metrics.



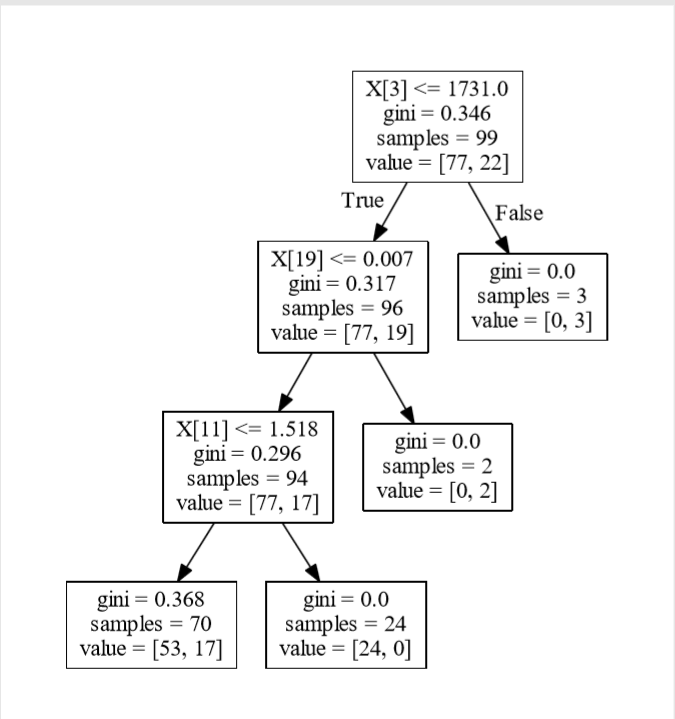
Visualize the decision tree using graphviz.



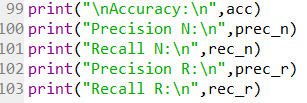
**The selected parameter values and the resulting optimal tree:**

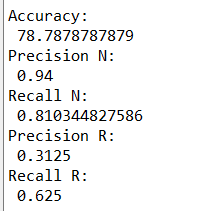
The criteria used for splitting nodes is gini. Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

max\_depth: The maximum depth of the tree. It is set to 3.



**For this decision tree finding the precision and recall values of each of the two classes.**

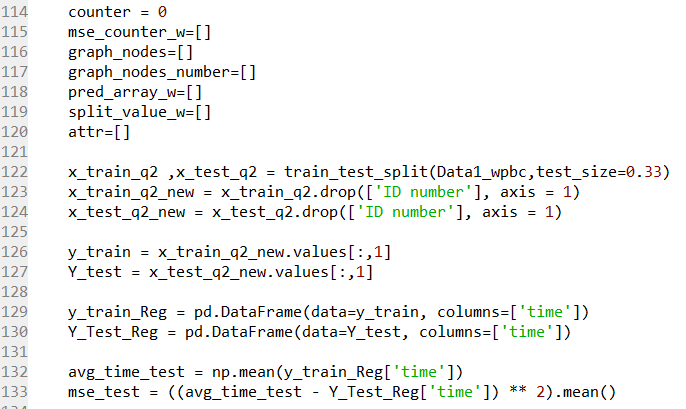




**An analysis of the performance metrics in terms of the data’s domain:**

The data has high class imbalance. The ‘N’ class is high in number compared to the ‘R’ class. This results in a skewed tree. The maximum attainable accuracy is 78%. It can also be seen that the precision and recall for class ‘N’ are relatively high compared to class ’R’. This is due to small number of instances in class ‘R’ compared to that of class ‘N’.

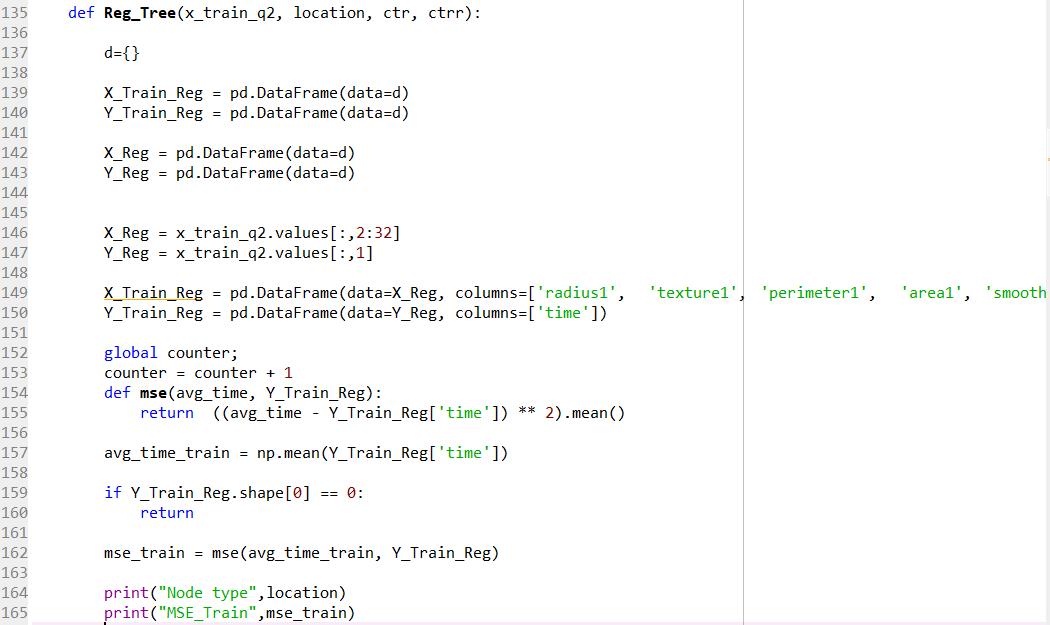
**Splitting the dataset into training (2/3rd) and testing dataset (1/3rd). Assuming the average number of disease-free years for the entire training dataset as the initial predicted value for each record of the dataset. Finding the mean square error (MSE) of making this prediction for all the training and the test set records (separately for training and test sets).**

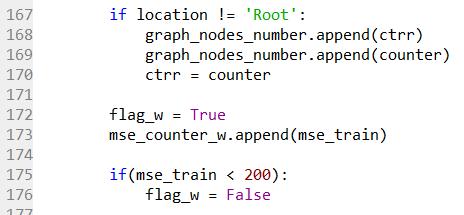


Splitting the data into train (2/3rd) and test(1.3rd). Separating the attributes from target column. For train and test datasets. Initializing the average test target class to compute mean squared error for test data set.

**Finding the attribute most closely correlated to the number of disease-free years in the training data; taking the median value of this attribute as the split-point and partitioning the dataset into two parts.**

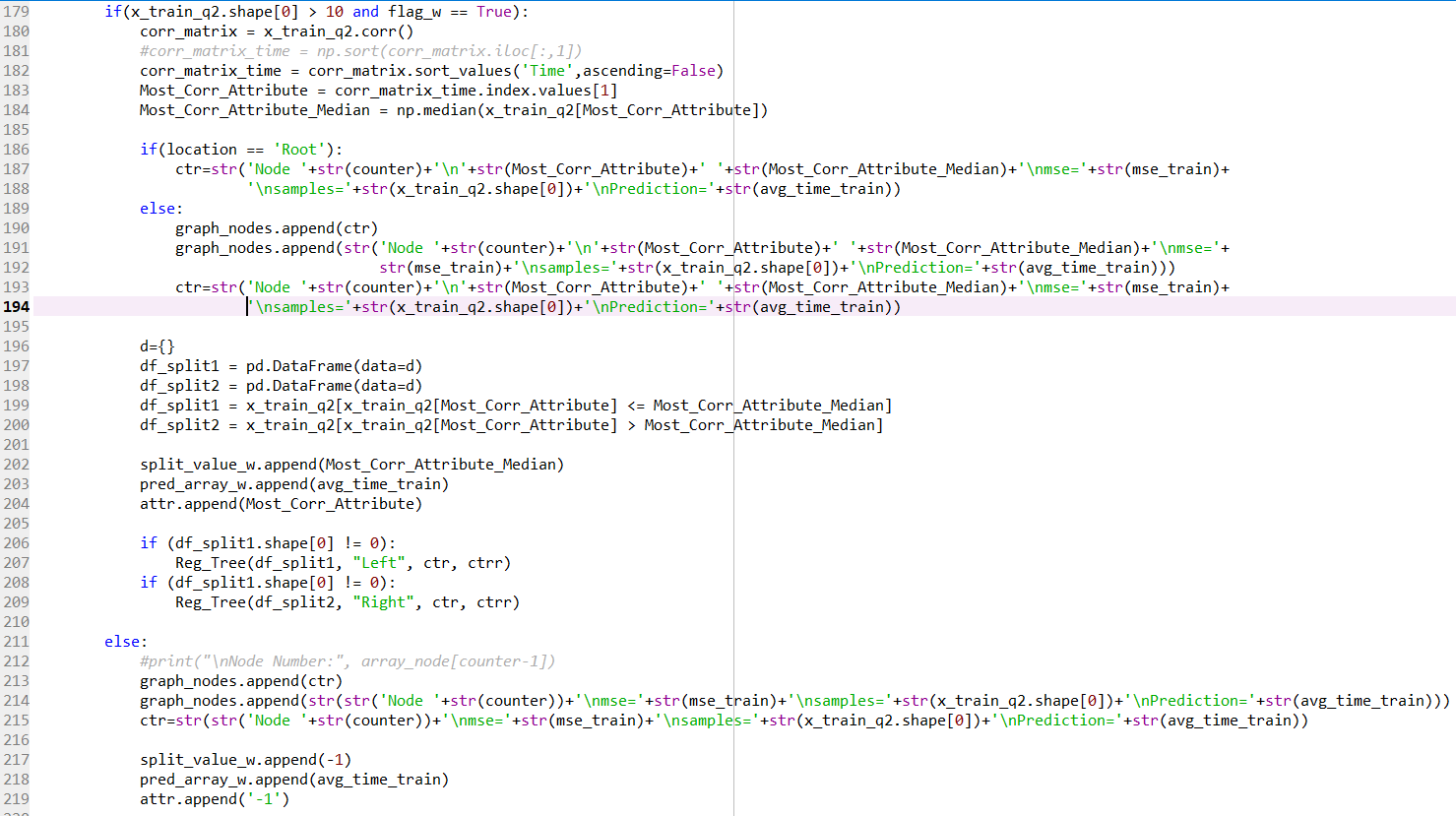
Defining a method for Regression tree building. This is a recursive function that takes in the train dataset, type of node (root, left or right) and flag variables to pass on the parent nodes for each iteration. Taking the median value of the most correlated column and splitting the dataset into two parts based on the median value.





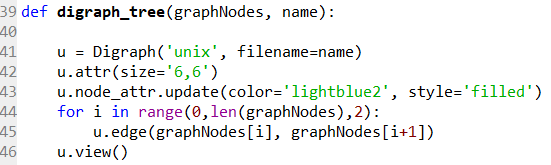
**Repeating above steps recursively for each leaf node until all the leaf nodes have their MSE values below some threshold value.**

Using the correlation matrix to obtain correlation between all the attributes with each other. Picking the attribute highly correlated with the target column, finding the median of the value and splitting the dataset into two parts. Recursively repeating the steps until there is no further split is possible.



Appending the leaf node details into a structure to build the regression tree structure.

The node values are stored in a structure to use it for testing and visualizing the tree.



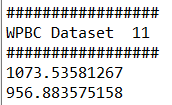
**Digraph** from **graphviz** is used to draw the regression tree. The edges are defined with parent and child node values. This constructs a tree similar to the one created in decision tree classification.

**Deciding on a suitable threshold to stop growing the regression tree.**

The stopping criteria used here is that if the mse at a node is below **200**, stop splitting the dataset. One more stopping criteria is that the number of instances in each node must be at least be **10** for proceeding with splitting the dataset. Tried a few different mse values as the stopping criteria and got the best result (minimum mse) for mse value as 200. Similarly, minimum number of instances at node for proceeding with split was best at 10. Since the number of instances are just 198, there has to be sufficient number of instances at the leaf nodes for the model to perform better.

**The resulting tree and details about each leaf node including its population, predicted value, and MSE values for the training and the test datasets:**

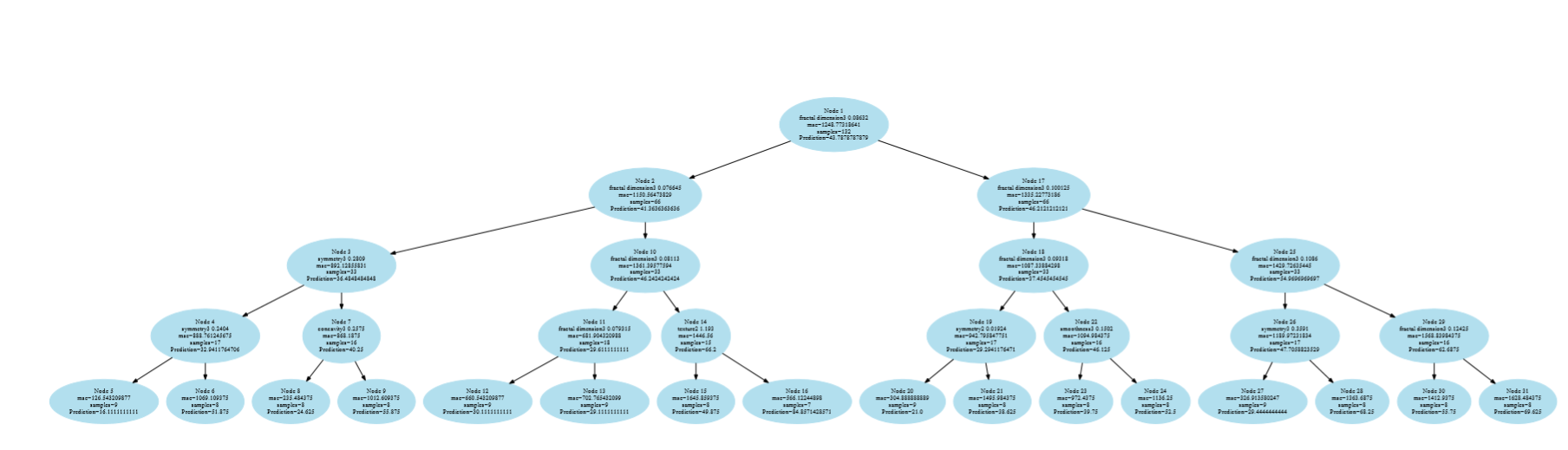
MSE values of training and the test dataset. The test set has lower mse compared to the train mse. This is the best possible regression model trained.



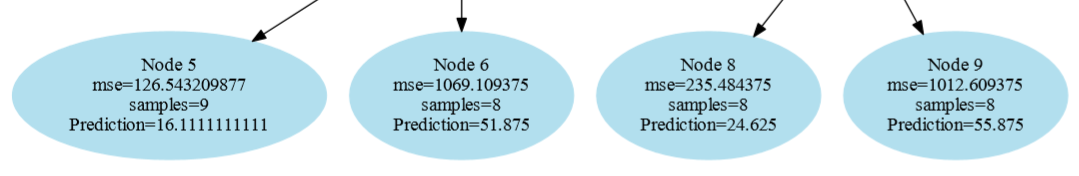
<-------- MSE before building Regression tree

<-------- MSE after building Regression tree

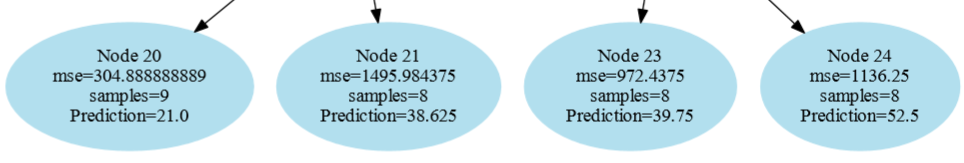
Resulting Tree:

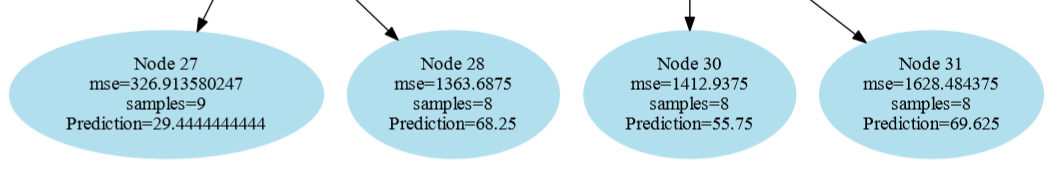


Leaf nodes with population, mse and prediction value details.



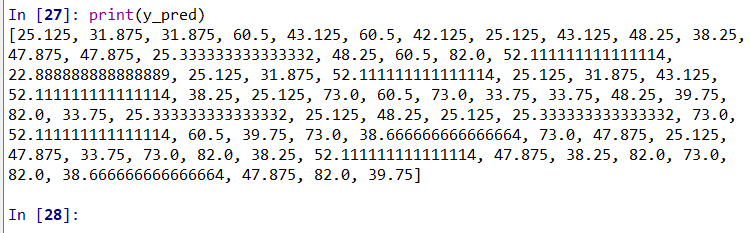






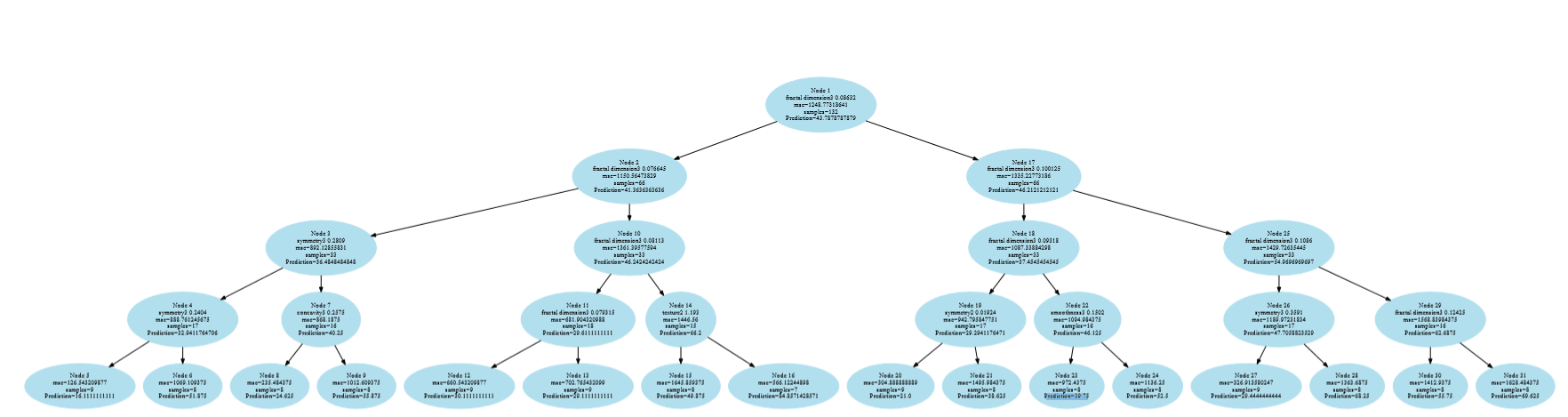
**Testset Predictions:**

The predicted values for test set are as mentioned below.



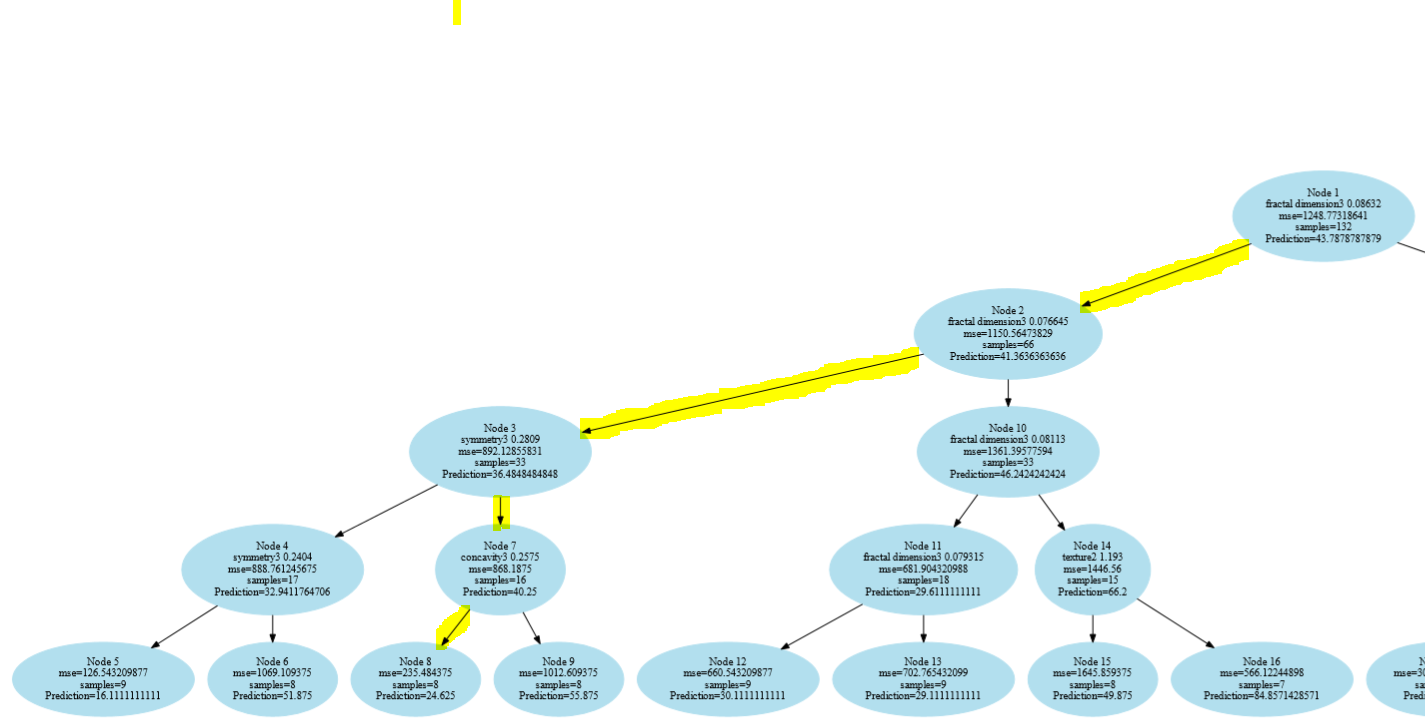
**An analysis of the results obtained by the regression tree and also of the structure of the regression tree:**

The regression tree obtained has 5 levels. It uses median of highly correlated attribute as the splitting criteria. The level or the depth of the tree is maintained with minimum mse and number of instances at each node. The prediction values are the average of the target column of instances at that particular node. The prediction with test dataset yields an overall mse less than the training mse. The tree thus obtained is complete binary in nature. After training the tree for many iterations, this is the best possible tree with least attainable mse values for train and test set.

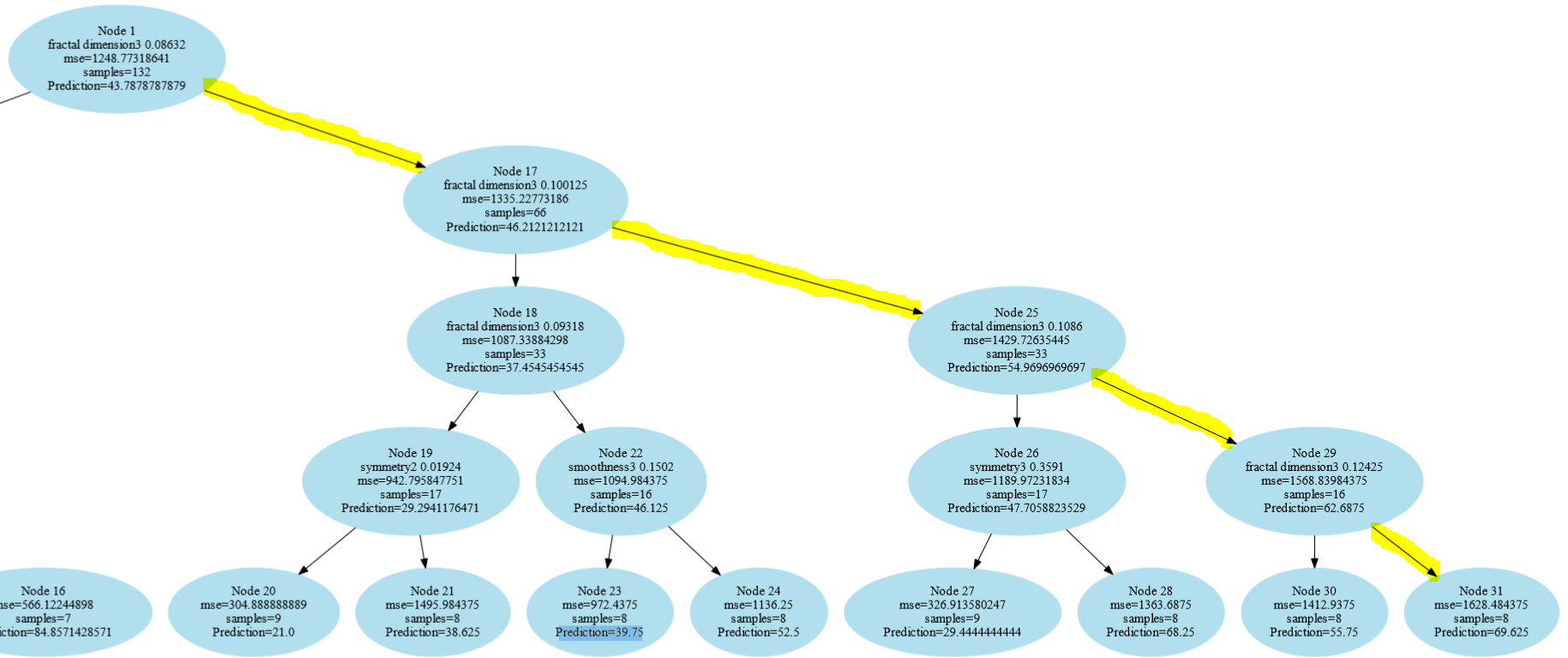


**The best and the worst paths of this regression tree:**

**Best path** has leads to the lowest mse along its path to the leaf node.



**Worst path** has larger mse along its path to the leaf node.



**The attributes the predictions depend on:**

[Fractal dimension3, symmetry3, symmetry2, concavity3, texture2, smoothness3]

**Similarities and differences between the decision tree obtained and this regression tree:**

**Similarities:**

Both approaches split the dataset at each node based on a criterion.

They have some common hyperparameters.

They are not restricted for binary classification.

**Differences:**

* Decision Tree:

Decision trees predict the target class. They are used to separate the dataset into classes belonging to the response variable.

Decision trees are used when the response or target variable is categorical in nature.

In classification tree, the idea is to split the dataset based on homogeneity of data.

The decision tree obtained is left skewed.

Split criterion: Gini index

* Regression Tree:

Regression trees are used when the target variable is numeric or continuous.

Regression trees are applicable for prediction type of problems as opposed to classification.

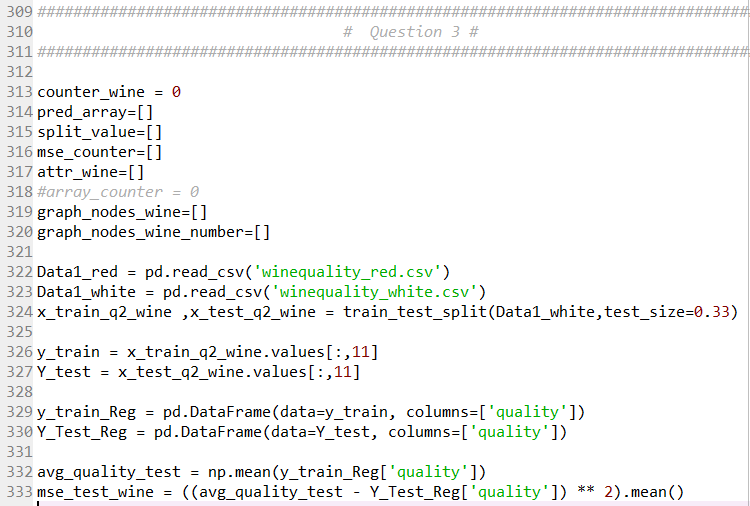
In a regression tree the idea is to fit a regression model to the target variable using each of the independent variables.

The regression tree is fully balanced.

Split criterion: Mean squared error

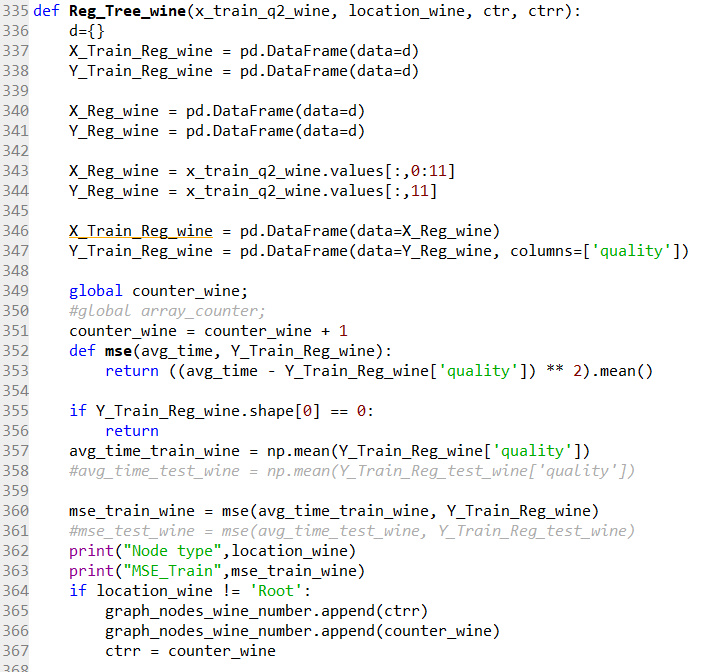
**Considering the White Wine Quality (WWQ) dataset. The target is the quality of wine which can take a value between 0 and 10. Building the best possible regression tree that can be obtained by following all the steps as described above:**

Loading the whitewine data and splitting the data into train(2/3rd) and test(1/3rd).

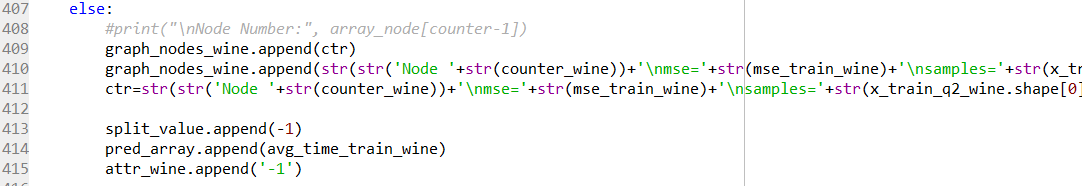


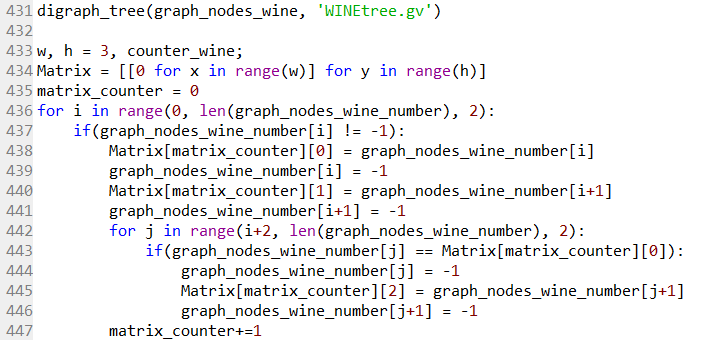
Constructing the regression tree by initially computing the mse of the train data with respect to the average of the target column values. Evaluating the correlation of attributes with the target column and selecting the most correlated attribute to obtain its median value. Now, with this median value as the splitting criterion, we split the dataset at that node into two parts based on whether or not the instance’s correlated attribute value is greater or lesser than the criterion.

This way the regression tree is built.

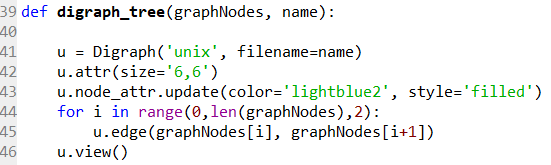








The node values are stored in a structure to use it for testing and visualizing the tree.

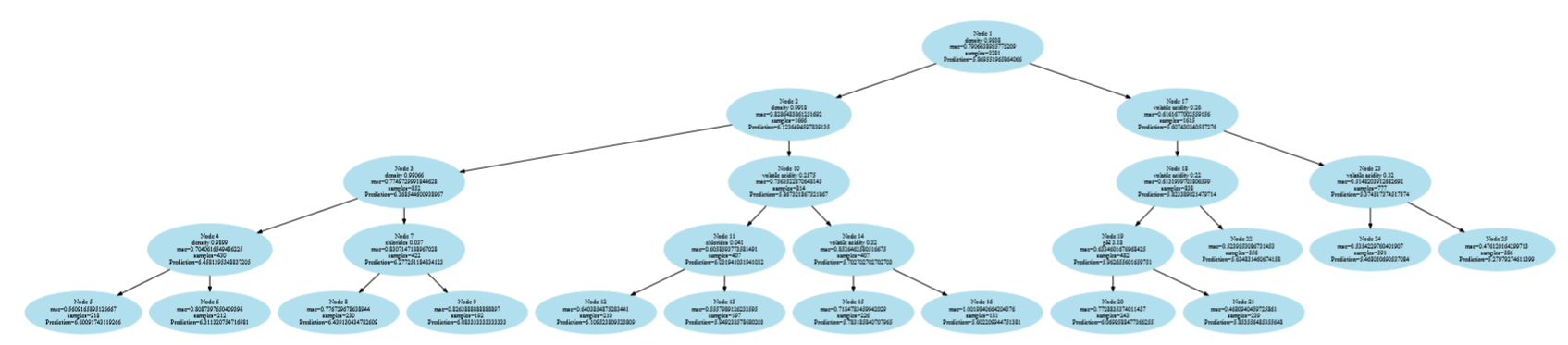


**Digraph** from **graphviz** is used to draw the regression tree. The edges are defined with parent and child node values. This constructs a tree similar to the one created in decision tree classification.

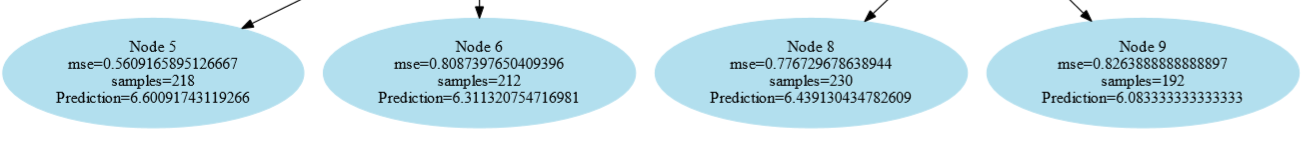
**Deciding the point at which to stop growing the regression tree:**

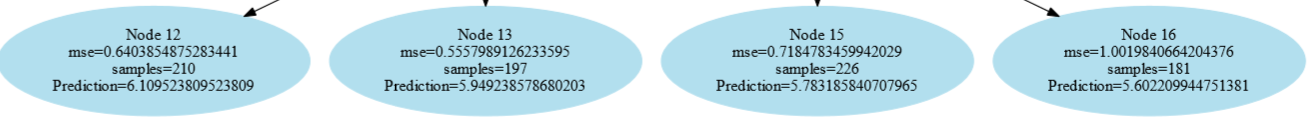
The stopping criteria is taken to be the minimum number of instances to be present for splitting to take place. The value is taken as 200 as this gives the best possible result with well-pruned tree. Also, the minimum mse value to be present to carry out with splitting is assumed to be 0.4. These together gives the optimum tree. After trying out different values for mse and minimum number if instances, this gave the best result.

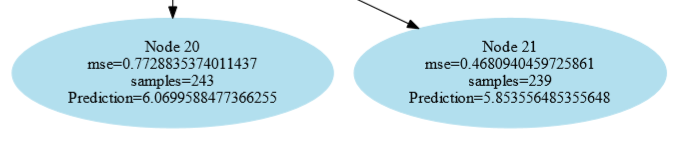
**The structure of the resulting regression tree and the details of each leaf node including its population, predicted value and the MSE values for the training and the test datasets.**

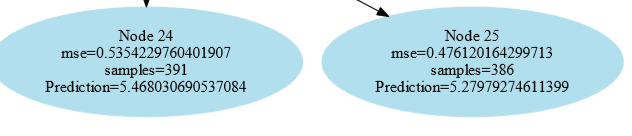


**Details of leaf node (training data):**

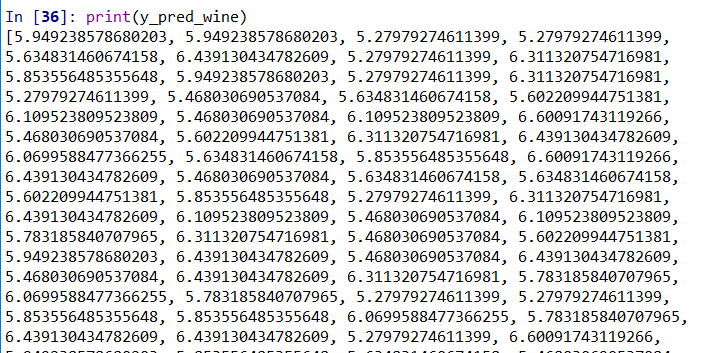








**Some of the test prediction values:**

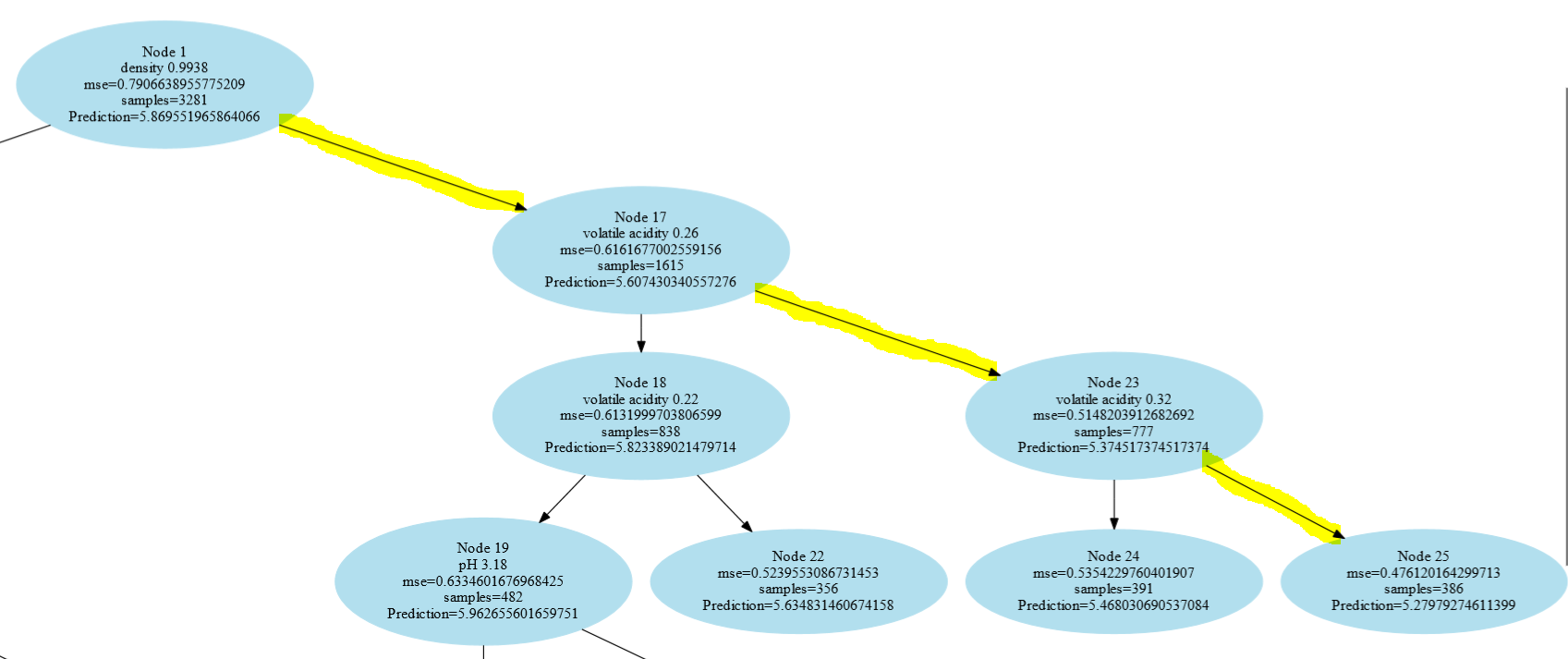


**Total MSE values for train and test**

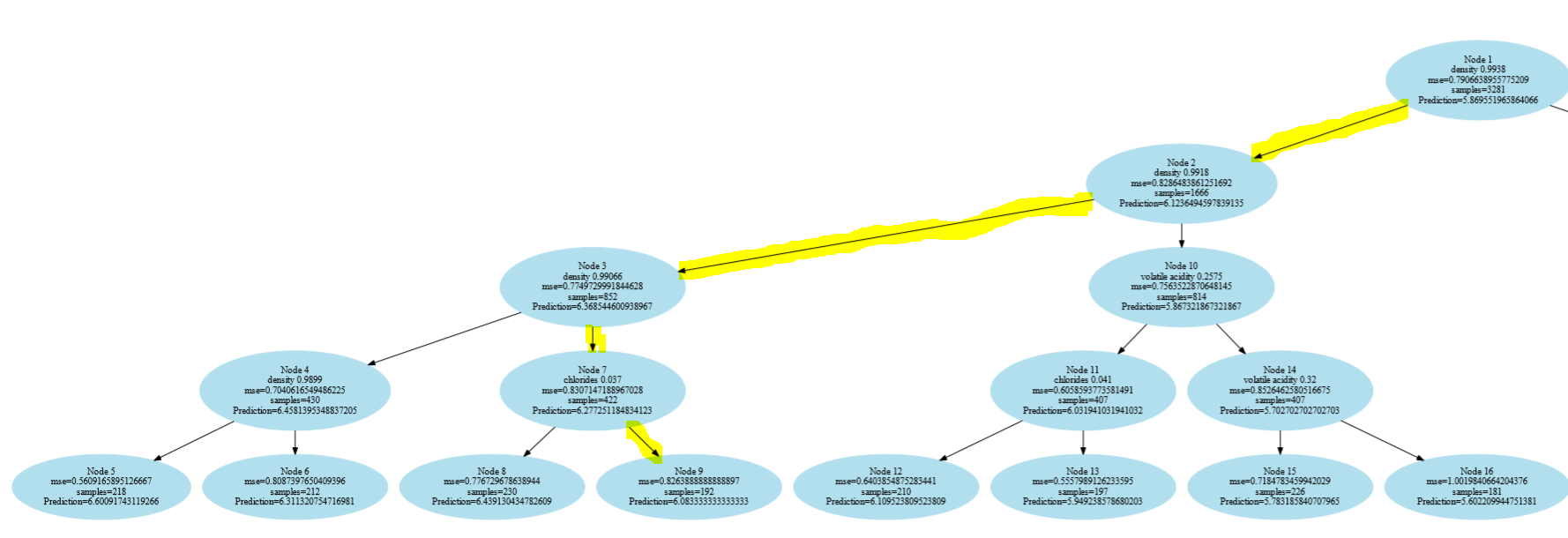


**The best and the worst predictors of wine quality:**

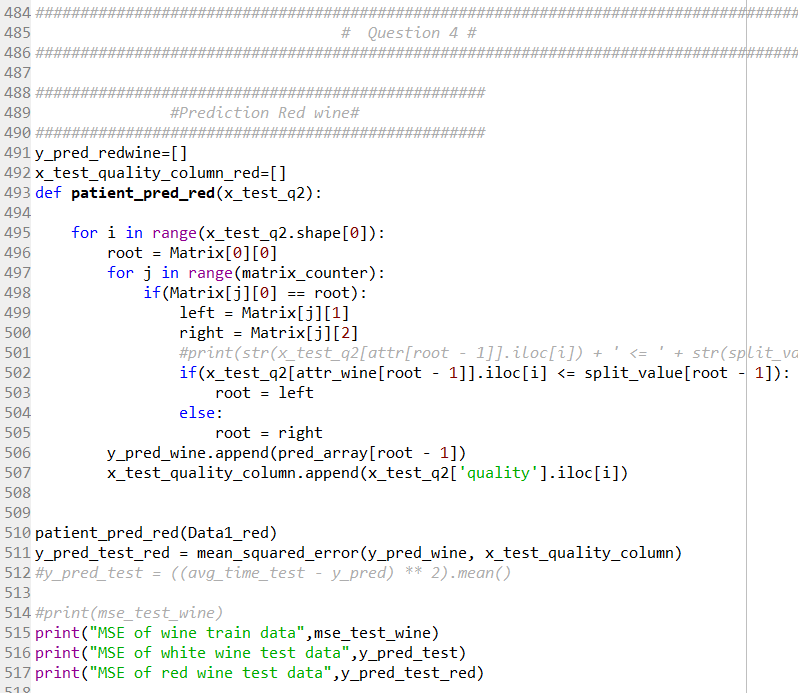
**Best Path:** mse reduces along this path from 0.9 to as low as 0.4. This appears to be the best path as the predicted values appear to be closer to the actual value. Also, there are considerable number of samples at each node to base this result on. This will act as a strong predictor. The depth at with the target is reached is low making it computationally efficient. The highmy correlated volatile acidity attribute is selected to do the prediction.



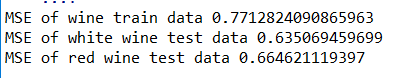
**Worst path:** mse along this path seem to increase till it reaches the leaf node 0.79 to 0.82. The depth of the tree along this path is more compared to the best path. This path has attributes density, chlorides which do not show much correlation with the target value.



**Using the red wine dataset as the test set on the regression tree obtained for white wines and computing the MSE for the records of the red wine dataset.**



**Comparing this performance to that for the white wine:**



It can be seen that the model works well for White wine data but performs not so well on Red wine data. The mse of red wine test is greater than the white wine test mse. This model is not good to work with red wine data.