Can we determine which variables are the most important to predicting which patients are at high risk of readmission?

I will use decision trees classification in the Python Scikit-learn package.

The business will be able to identify which variables are the most important to predicting which patients are at high risk of readmission with some measure of confidence. This process will provide weight for making improvements to hospital services and treatments.

The decision trees are a learning technique used for classification and regression, seen as a piecewise constant approximation. We use the decision trees to create a model that predicts the value of a target variable by learning simple decision rules indirect from the data features.

According to Nagesh Singh Chauhan, the decision trees classification method is based on the following assumptions:

- The whole training set is considered as the root.

- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.

- Records are distributed recursively on the basis of attribute values.

- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

(Nagesh Singh Chauhan. 2020)

I have many programming languages that I can use, such as R and Python, to achieve this process. In this assignment, I use Python to assess data quality, clean the data, and predict the data. Python is a multipurpose programming language with libraries that extend its capabilities to do statistical analysis. For the beginner, coding in Python is easy to read and easy to understand the flows of the program. Also, I will work with Python in the Jupyter notebooks as a convenient way to run code and visualizations and accessible to running documentation for my reference.

The libraries and packages used to clean the data run in python environments such as Panda, NumPy, Scipy, Matplotlib, and Seaborn. These libraries and packages provide functionalities like reading large datasets like statistical functions like Zscore, creating visualization models like box plots and histograms. I also use Scikit-learn for decision trees classifier.

We can see how the tree is split, what are the gini for the nodes, the records in those nodes, and their labels.

\*\*Pruning tree\*\*

The tree looks enormous and complex. We can optimize the decision tree classifier only by pre-pruning the control variable - the maximum depth of the tree. In the plot tree above, we can plot a decision tree on the same data with max-depth=3.

Overfitting the training dataset also might be holding this classification decision tree back. So we need to prune the tree, which should solve the overfitting issue and give us better effects. Pruning a tree is all about finding the correct alpha value for the pruning parameter, which controls how little or much pruning happens.

Now let find the optimal value of alpha.

In the graph, we see that the accuracy for the testing dataset had maximum value when alpha = 0.001. After 0.001, the accuracy of the training dataset drops off, which suggests we should set ccp\_alpha = 0.001. (I will use ccp\_alpha = 0.001 to create Mean Squared Error and R-squared score.)

We can see how the tree is split, the gini for the nodes, the records in those nodes, and their labels. This pruned model is simple, and understandable than the previous decision tree model plot.

MSE is the mean square error. The larger the number, the larger the error. The lower the error, the better, and 0 means the perfect model. MSE = 0.256 is small, which means the difference between the observed value (y1, y2, ..) and the predicted one's pred(y1, y2,..) is small. R-squared score = 0.71 is ready good correlation. This decision trees model is good.

The first resultant tree is unpruned, unexplainable, and challenging to understand. The tree looks enormous and complex. We can optimize the decision tree classifier only by pre-pruning the control variable - the maximum depth of the tree. In the plot tree above, we can plot a decision tree on the same data with max-depth=3.

Overfitting the training dataset also might be holding this classification decision tree back. So we need to prune the tree, which should solve the overfitting issue and give us better effects. Pruning a tree is all about finding the correct alpha value for the pruning parameter, which controls how little or much pruning happens.

After optimizing it by pruning, pruned model is simple, and understandable than the previous decision tree model plot.

We obtained an accuracy of 0.983 at max-depth = 3

- New\_Medical\_Accuracy: 0.983

- New\_Medical\_Precision: 0.966804979253112

- New\_Medical\_Recall: 0.9858956276445698

The classification rate of 98% is considered good accuracy, which is better than the previous model. Precision is about being precise and how accurate the model is. For example, the classification model predicted that patients are at high risk of readmission 97% of the time. However, if there are patients at high risk of readmission in the test set, the KNN model can identify it 99% of the recall time.

In the accuracy vs. alpha for training and testing sets graph, we see that the accuracy for the testing dataset had maximum value when alpha = 0.001. After 0.001, the accuracy of the training dataset drops off, which suggests we should set ccp\_alpha = 0.001. With alpha = 0.001 we have:

- Mean squared error (MSE): 0.25644777940030317

- R-squared score: 0.7126009028137881

MSE - mean square error. The larger the number, the larger the error. The lower the error, the better, and 0 means the perfect model. MSE = 0.256 is small, which means the difference between the observed value (y1, y2, ..) and the predicted one's pred(y1, y2,..) is small. R-squared score = 0.71 is ready good correlation. This decision tree model is good.

Now let look at the metric model. The dimension of this cnf\_matrix is 2 by 2 because this model is binary classification. We have 0 and 1 for ReAdmis (value yes/no). The array from con\_matrix follows true negatives, false positives, false negatives, and true positives. Diagonal values represent accurate predictions, while non-diagonal elements are inaccurate predictions. For example, 13e+03 and 7e+02 are actual predictions in the output, and 10 and 24 are incorrect predictions.

The ROC curve plots the true positive (TP) rate against the false positive (FP) rate. In this ROC curve, the AUC score is 0.9968, which means close to the AUC score of 1 represents a perfect classifier.

The first limitation of the decision trees is that the slight variation(or variance) in data can result in a different decision tree. The second limitation of the decision tree is biased with an imbalance dataset, so I recommend that we balance out the dataset before creating the decision tree. Even though the model has the Pseudo R-square at 71%, which is high, MSE at 0.25 is low, and AUC at 0.99, the limitations of the data analysis are that we do not know the time frame of this dataset. The data set has an unbalanced data-target variable, ReAdmis (yes values is about 60% of no values). I also think we do not have enough data and need to collect more data for predicting which patients are at high risk of readmission.

Based on the results, I would advise the business to bag and boost algorithms and add the class\_weight parameters if the dataset is unbalanced. We need to reduce more variables in the data frame to predict better which patients are at high risk of readmission with the decision tree model. I also think we do not have enough data and need to collect more data to predict which patients are at high risk of readmission; we will need full features for each example to compute the space by filling the missing values with the average value of the variable across the whole dataset.