Statistical Learning, Homework #2

Sofia Pietrini

27/04/2025

Introduction

The data analyzed in this study comes from a research conducted to investigate the association between diabetes progression after 1 year from a baseline and a number of clinical predictors, measured in 442 diabetic patients. The disease progression is explained by the variable `progr` (the higher the value, the worse the progression), the other explanatory variables are: age, sex, BMI (body mass index), BP (average blood pressure, in mm Hg), TC (total cholesterol, mg/dl), LDL (low-density lipoproteins, mg/dl), HDL (high-density lipoproteins, mg/dl), TCH (ratio between total cholesterlol and HDL), TG (triglycerides level, mg/dl, log-scaled), GC (blood glucose, mg/dl).

Loading data set, data exploration

The libraries needed for the study are: `tidyverse`, `tree`, `randomForest`, `gbm`, `caret` and `conflicted`.

```
dataf <- read.table("db.txt", header = TRUE)
dataf <- as.tibble(dataf)
head(dataf, 5)
```

```
## # A tibble: 5 x 11
##     age   sex   BMI    BP    TC   LDL   HDL   TCH    TG    GC progr
##   <int> <int> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <dbl> <int> <int>
## 1    59     2  32.1   101   157  93.2    38     4  4.86    87   151
## 2    48     1  21.6    87   183 103.     70     3  3.89    69    75
## 3    72     2  30.5    93   156  93.6    41     4  4.67    85   141
## 4    24     1  25.3    84   198 131.     40     5  4.89    89   206
## 5    50     1  23     101   192 125.     52     4  4.29    80   135
```

```
dataf$sex <- factor(dataf$sex)
levels(dataf$sex) <- c("Male", "Female")
attach(dataf)
summary(dataf)
```

```
##       age            sex           BMI              BP               TC
##  Min.   :19.00   Male  :235   Min.   :18.00   Min.   : 62.00   Min.   : 97.0
##  1st Qu.:38.25   Female:207   1st Qu.:23.20   1st Qu.: 84.00   1st Qu.:164.2
##  Median :50.00                Median :25.70   Median : 93.00   Median :186.0
##  Mean   :48.52                Mean   :26.38   Mean   : 94.65   Mean   :189.1
##  3rd Qu.:59.00                3rd Qu.:29.27   3rd Qu.:105.00   3rd Qu.:209.8
##  Max.   :79.00                Max.   :42.20   Max.   :133.00   Max.   :301.0
##       LDL              HDL             TCH              TG
##  Min.   : 41.60   Min.   :22.00   Min.   :2.00   Min.   :3.258
##  1st Qu.: 96.05   1st Qu.:40.25   1st Qu.:3.00   1st Qu.:4.277
##  Median :113.00   Median :48.00   Median :4.00   Median :4.620
##  Mean   :115.44   Mean   :49.79   Mean   :4.07   Mean   :4.641
##  3rd Qu.:134.50   3rd Qu.:57.75   3rd Qu.:5.00   3rd Qu.:4.997
##  Max.   :242.40   Max.   :99.00   Max.   :9.09   Max.   :6.107
##        GC             progr
##  Min.   : 58.00   Min.   : 25.0
##  1st Qu.: 83.25   1st Qu.: 87.0
##  Median : 91.00   Median :140.5
##  Mean   : 91.26   Mean   :152.1
##  3rd Qu.: 98.00   3rd Qu.:211.5
##  Max.   :124.00   Max.   :346.0
```

The data set is composed of 442 observations of 11 variables. All the variables are numerical, with the majority being of type `double` except for `age`, `sex`, `TC`, `GC`. To facilitate the analysis, the variable `sex` was transformed into a factor.
From the summary, we can notice the absence of missing values in the data set.

Decision tree

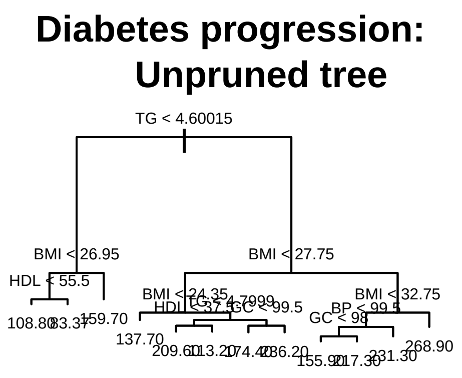
In this section we are going to use decision trees to analyze the data set. In particular, we will fit a decision tree on the whole data to predict the patients' diabetes progression.
Binning continuous variables like `BMI`, `LDL` and `TG` would make our model easier to understand. For example, it can help generate more intuitive rules, such as labeling BMI as "high," rather than splitting at a specific numerical value.
A first choice was made by binning the three continuous input variables. The thresholds used align with commonly accepted clinical standards: 25 for `BMI`, to separate normal from overweight and obese, 130 mg/dL for `LDL` to distinguish between optimal and higher levels, 5 for `TG` (log-scaled), which corresponds roughly to the clinical cutoff of 150 mg/dL.

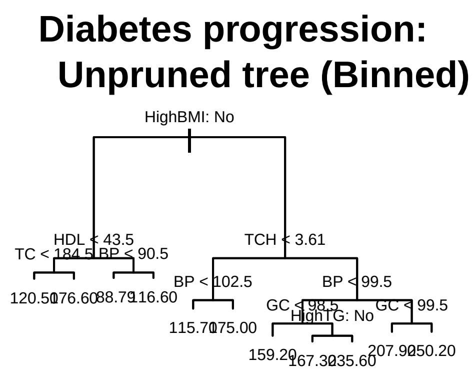
```r
tmp <- as_tibble(dataf) %>%
    mutate(HighBMI = cut(BMI, c(-Inf, 25, Inf), c("No", "Yes"))) %>%
    select(-BMI)
tmp <- as_tibble(tmp) %>%
    mutate(HighLDL = cut(LDL, c(-Inf, 130, Inf), c("No", "Yes"))) %>%
    select(-LDL)
x <- as_tibble(tmp) %>%
    mutate(HighTG = cut(TG, c(-Inf, 5, Inf), c("No", "Yes"))) %>%
    select(-TG)  #x is final data set resulting from the binning process
```

```r
tree.progression <- tree(progr ~ ., data = dataf)  #tree fitted to original data set
tree.progression_binned <- tree(progr ~ ., data = x)  #tree fitted to binned data set
```

The summaries and the plots of the trees produced above were, then, displayed side by side for comparison.

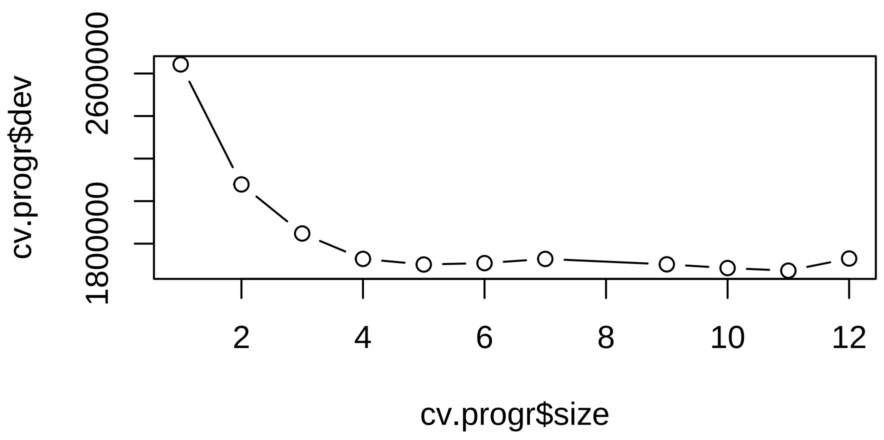
Summary 1	Summary 2
Regression tree:	Regression tree:
tree(formula = progr ~ ., data = dataf)	tree(formula = progr ~ ., data = x)
Variables actually used in tree construction:	Variables actually used in tree construction:
[1] ”TG” ”BMI” ”HDL” ”GC” ”BP”	[1] ”HighBMI” ”HDL” ”TC” ”BP” ”TCH” ”GC” ”HighTG”
Number of terminal nodes: 12	Number of terminal nodes: 11
Residual mean deviance: 2674 = 1150000 / 430	Residual mean deviance: 3183 = 1372000 / 431
Distribution of residuals:	Distribution of residuals:
Min. 1st Qu. Median Mean 3rd Qu. Max.	Min. 1st Qu. Median Mean 3rd Qu. Max.
-140.900 -35.830 -4.805 0.000 33.540 154.100	-150.200 -35.790 -4.787 0.000 37.730 186.800





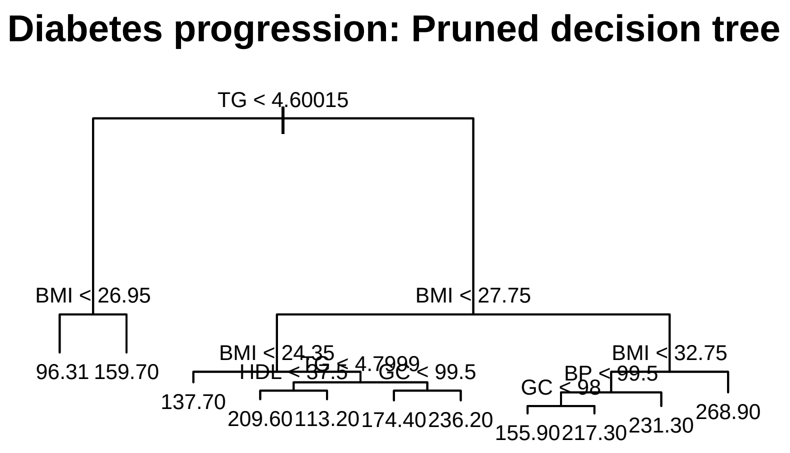
Notice that the output of `summary()` indicates that only 5, in the first case, and 7 of the variables have been used in constructing the tree. Additionally, the most important predictors change between the two cases.
Interestingly, fitting a tree to the original data set `dataf` resulted in a smaller value of residual mean deviance (2674 compared to 3183), which suggests it was better at capturing variation and structure in the data. Although the binned version made the decision rules more straightforward, it came at the cost of predictive precision. We then proceeded by using the original continuous inputs, since the resulting model offered better overall performance while still maintaining a reasonable level of interpretability.
Subsequently, cross-validation of the tree was performed to decide tree complexity.

```r
set.seed(1)
cv.progr <- cv.tree(tree.progression)
plot(cv.progr$size, cv.progr$dev, type = "b")
```



The plot indicates that the tree with minimum cross-validation error rate (`cv.progr$dev`) has 11 terminal nodes (`cv.progr$size`). Based on this result, we then proceeded with the pruning of the tree.

```r
prune_progr <- prune.tree(tree.progression, best = 11)
# generates tree with 11 terminal nodes starting from the original tree
plot(prune_progr)
text(prune_progr, pretty = 0, cex = 0.7)
title(main = "Diabetes progression: Pruned decision tree")
```



```r
summary(prune_progr)
```

```
##
## Regression tree:
## snip.tree(tree = tree.progression, nodes = 4L)
## Variables actually used in tree construction:
## [1] "TG"  "BMI" "HDL" "GC"  "BP"
## Number of terminal nodes:  11
## Residual mean deviance:  2732 = 1177000 / 431
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -140.90  -36.31   -6.31    0.00   35.34  156.70
```

The decision tree plot shows that triglyceride levels (TG) emerge as the primary factor in predicting diabetes progression, forming the first and most significant split. Among individuals with lower TG levels, BMI serves as the next major discriminating variable, where a lower BMI is associated with better predicted outcomes. Within these groups, additional splits on BMI, HDL cholesterol, and GC values further differentiate risk, with higher HDL and lower GC linked to more favorable progression scores. For those with higher TG levels, BMI continues to be an important divider: individuals with lower BMI tend to have better outcomes, while higher BMI combined with higher blood glucose concentrations (GC) and blood pressure (BP) indicate worse progression.
Examination of the summary statistics reveals that the residual mean deviance increased to 2732, exceeding the value obtained with the unpruned tree. This increase occurs because the pruned tree possesses a simpler structure, which results in a slightly less precise fit to the training data compared to the potentially overfit, larger, unpruned tree.

Random forest

Since the performance of the decision tree was not fully satisfactory, we attempted to improve the model by using random forests.
The library `randomForest` provides the function `randomForest()` which automatically builds a number of decision trees on bootstrapped training samples. At each split in each tree, only a random sample of m predictors, out of the total p predictors, is considered as split candidates. This number of predictors to sample is specified using the `mtry` parameter.
By default, `randomForest()` uses p/3 variables when building a random forest of regression trees. In our case, due to having 10 predictors, we chose `mtry`=3. The number of trees was set to 500, a value frequently employed in practice for robust model generation.

```r
set.seed(1)
bag.progr <- randomForest(progr ~ ., data = dataf, mtry = 3, ntree = 500, importance = TRUE)
bag.progr
```

```
##
## Call:
##  randomForest(formula = progr ~ ., data = dataf, mtry = 3, ntree = 500,      importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 3267.769
##                     % Var explained: 44.89
```

The random forest model fitted on the data results in a mean of squared residuals of approximately 3268, explaining about 45% of the variability in the diabetes progression.
The regression tree fitted on the same data set as the random forest produced a residual mean deviance of 2674, which is noticeably lower.
This can be due to the fact that random forests, by building an ensemble of de-correlated trees, tend to capture more subtle patterns and noise, while regression trees might be more prone to overfitting when compared to random forest.
The mean of squared residual just reflects the model performance in fitting the training set, the generalization on new data of the two model can be compared only by doing predictions on a test set.
Following the generation of the final random forest model, variable importance was assessed. Although several methods exist for representing it, an ordered table format based on the percentage increase in Mean Squared Error (%IncMSE) was chosen for presentation.

```r
importance_rf <- importance(bag.progr)
# creation of the ordered table with knitr library
knitr::kable(importance_rf[order(importance_rf[, "%IncMSE"], decreasing = TRUE),
    ])
```

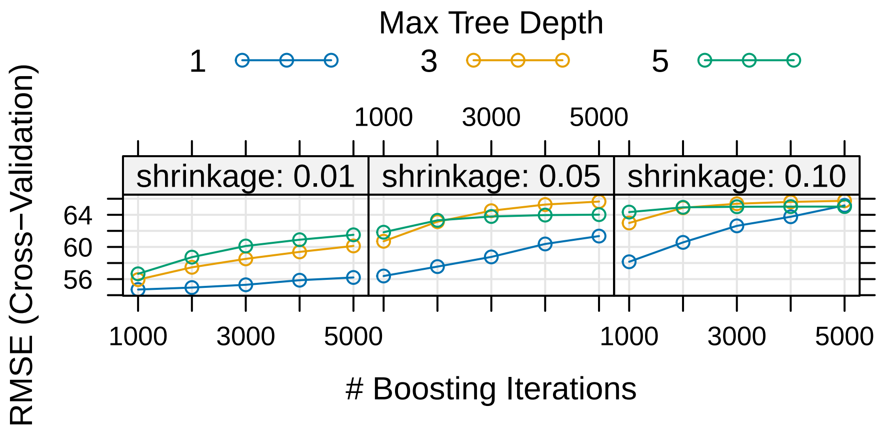
	%IncMSE	IncNodePurity
TG	34.347734	527350.5
BMI	33.941309	587133.3
BP	17.835972	292862.4
TCH	10.508242	173394.4
HDL	10.026117	213942.4
GC	7.333789	218336.8
TC	7.017659	143262.9
LDL	6.644723	154704.5
sex	4.601480	32589.4
age	2.901837	144147.6

According to this metric, the largest increases in MSE are associated with the variables `TG` and `BMI`, indicating their high predictive importance (both exhibiting %IncMSE values exceeding 30).

Boosted random forest

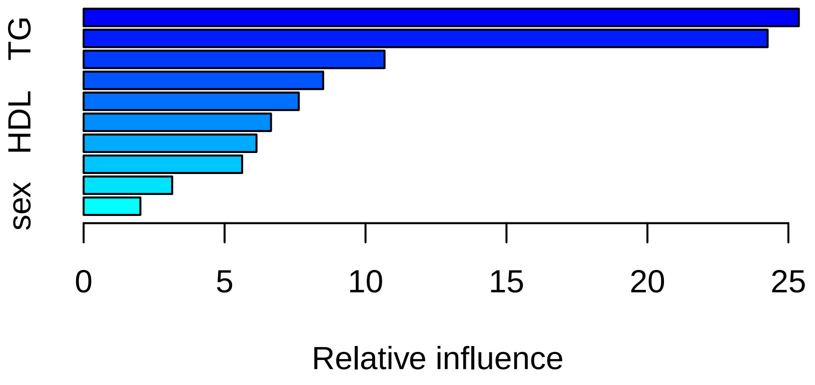
An alternative way to further improve our prediction accuracy would be the use of boosted random forest. Boosting works similarly to random forest, but the trees are grown sequentially using information from previously grown trees. Additionally, boosting does not involve bootstrap sampling.
Firstly, we use cross validation to tune the values for key parameters such as the number of trees, interaction depth and shrinkage. Values with lowest RMSE were selected.

```r
set.seed(1)
boosted_control <- trainControl(method = "cv", number = 10)
boosted_par_tuning <- expand.grid(n.trees = c(1000, 2000, 3000, 4000, 5000), interaction.depth = c(1,
    3, 5), shrinkage = c(0.01, 0.05, 0.1), n.minobsinnode = 10)

diab_gbm_tuned <- train(progr ~ ., data = dataf, method = "gbm", distribution = "gaussian",
    trControl = boosted_control, tuneGrid = boosted_par_tuning, verbose = F)

plot(diab_gbm_tuned)
```



Since we are looking for the smallest value of RMSE, the values chosen for the key parameters are 0.01 for `shrinkage` and 1 for `interaction.depth`. As for the tuning of `n.trees`, one would look for a plateau in the plot, choosing the value after which there is no significant increase in RMSE. The plot corresponding to the previously tuned parameters does not allow for an easy choice of the number of trees, consequently the default value of 5000 was used.
`gbm()` was consequently run with the option `distribution = "gaussian"`, since this is a regression problem and the parameters tuned as previously illustrated.

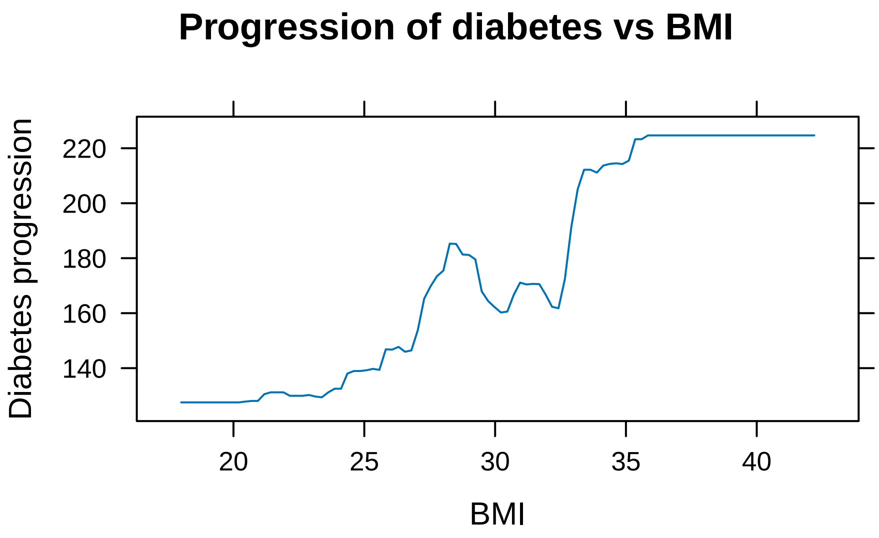
```r
set.seed(1)
boost.progr <- gbm(progr ~ ., data = dataf, distribution = "gaussian", n.trees = 5000,
    interaction.depth = 1, shrinkage = 0.01, cv.folds = 10, verbose = FALSE)
summary(boost.progr)
```



```
##     var   rel.inf
## BMI BMI 25.371908
## TG   TG 24.264451
## BP   BP 10.674463
## GC   GC  8.497782
## LDL LDL  7.632266
## HDL HDL  6.647249
## age age  6.133314
## TC   TC  5.622898
## TCH TCH  3.142075
## sex sex  2.013594
```

The `summary()` function produces a relative influence plot, as well as the relative influence statistics. Since `BMI` is shown to be the most important predictor for diabetes progression, we can plot it against `progr`.

```
plot(boost.progr, i = "BMI", ylab = "Diabetes progression", main = "Progression of diabetes vs BMI")
```



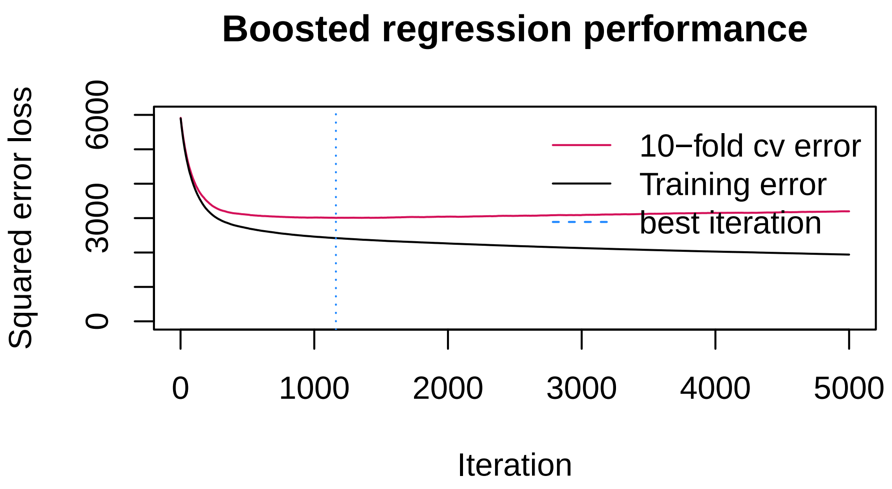
The provided partial dependence plot illustrates the estimated effect of BMI on predicted diabetes progression, holding all other variables constant. Generally, diabetes progression tends to increase with higher BMI. The progression shows a sharp rise around a BMI of 33, before reaching a plateau at higher BMI.

Since the `n.trees` parameter was set to the default value, we employed cross-validation to try to find an optimal value for the parameter. The function `gbm.perf` was employed, since it estimates the optimal number of boosting iterations for a `gbm` object.

```
set.seed(1)
best_treenum_cv <- gbm.perf(boost.progr, method = "cv", plot.it = FALSE)  #1162
best_treenum_cv
```

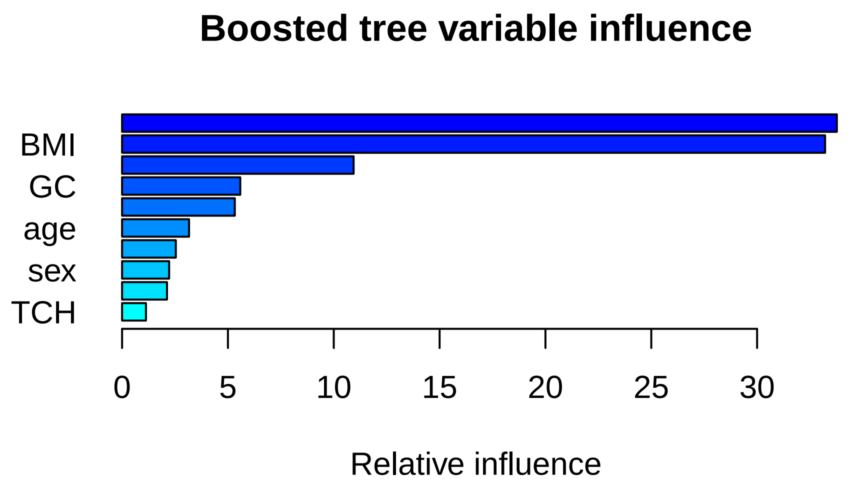
```
## [1] 1162
```

```
cv_plot <- boost.progr$cv.error
plot(cv_plot, type = "l", col = "#D41159", xlab = "Iteration", ylab = "Squared error loss",
    ylim = c(0, 6000), main = "Boosted regression performance")
lines(boost.progr$train.error, col = "black")
abline(v = best_treenum_cv, col = "#1A85FF", lty = 3)
legend("topright", legend = c("10-fold cv error", "Training error", "best iteration"),
    col = c("#D41159", "black", "#1A85FF"), lty = c(1, 1, 2), bty = "n")
```



This plot displays the training error and 10-fold cross-validation error against the number of boosting iterations for the model. While training error decreases continually, the CV error reaches a minimum at approximately 1162 iterations (dashed line), indicating the optimal number of trees before overfitting begins.
Following the determination of the optimal number of trees, variable influence was examined. The summary indicates that TG and BMI are again identified as the most influential predictors.

```r
summary(boost.progr, n.trees = best_treenum_cv, main = "Boosted tree variable influence",
    las = 1)  #summary of boosted tree generated with optimal number of iterations
```



```
##     var   rel.inf
## TG   TG 33.767542
## BMI BMI 33.211681
## BP   BP 10.938206
## GC   GC  5.584000
## HDL HDL  5.324822
## age age  3.163537
## LDL LDL  2.537877
## sex sex  2.222285
## TC   TC  2.121960
## TCH TCH  1.128090
```

Performance of the three models

Model performance was compared using a 10-fold nested cross-validation approach. The outer 10 folds were utilized for evaluating the final predictive performance. Within each outer training fold, model complexity was re-optimized independently: internal cross-validation was employed for decision trees and boosting models, while the Out-of-Bag error was used for random forests. This nested methodology ensures an unbiased performance assessment by preventing the outer test data from influencing complexity selection.

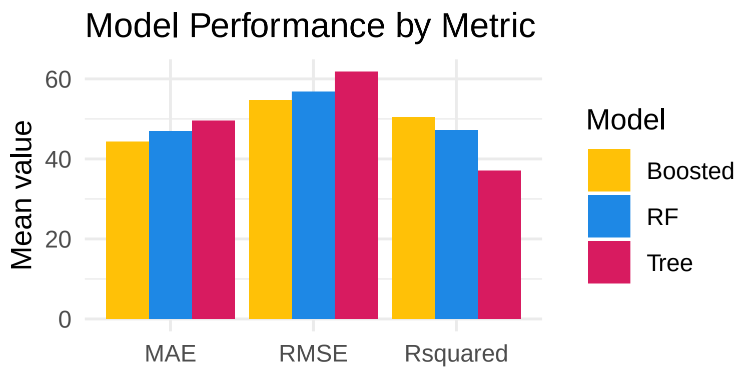
```r
set.seed(1)
ctrl <- trainControl(method = "cv", number = 10)  #10-fold cv

ctrl_diab_pruned_tree <- train(progr ~ ., data = dataf, method = "rpart", trControl = ctrl,
    tuneLength = 10)
ctrl_diab_random_forest <- train(progr ~ ., data = dataf, method = "rf", trControl = ctrl,
    tuneLength = 5, importance = T)
ctrl_diab_boosted <- train(progr ~ ., data = dataf, method = "gbm", trControl = ctrl,
    verbose = FALSE, tuneLength = 5)

results <- resamples(list(Tree = ctrl_diab_pruned_tree, RF = ctrl_diab_random_forest,
    Boosted = ctrl_diab_boosted))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: Tree, RF, Boosted
## Number of resamples: 10
##
## MAE
##             Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## Tree    42.41508 45.08509 46.97142 48.41002 50.94850 58.55047    0
## RF      42.12633 45.08762 46.94556 46.54626 47.77329 50.06712    0
## Boosted 38.42067 42.70510 44.31168 44.75354 46.57653 52.90543    0
##
## RMSE
##             Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## Tree    52.98221 55.33539 60.28714 60.55358 63.70180 70.61052    0
## RF      49.99604 55.40215 57.39803 56.39678 58.15165 59.17600    0
## Boosted 47.78437 51.80051 54.05434 55.07468 57.55156 65.80744    0
##
## Rsquared
##              Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Tree    0.1941628 0.3268416 0.3986645 0.3960310 0.4731795 0.6152249    0
## RF      0.3956995 0.4383136 0.4749509 0.4817397 0.5074833 0.6314498    0
## Boosted 0.3466679 0.4707196 0.5038334 0.4902557 0.5276428 0.6130912    0
```

It can be observed from the plot that the boosted model generally exhibits lower error metrics (MAE, RMSE) and higher R-squared values compared to the pruned tree and random forest models. Lower values of Mean Absolute Error and Root Mean Squared Error indicate that the predictions made by the boosted model are, on average, closer to the actual observed outcome. A higher R-squared value means that a larger proportion of the variance in the target variable is explained by the model, suggesting that the boosted model is better at capturing the underlying relationships in the data.

```r
library(ggplot2)
model_means <- data.frame(Model = rep(c("Tree", "RF", "Boosted"), times = 3), Metric = rep(c("MAE",
    "RMSE", "Rsquared"), each = 3), MeanValue = c(49.48875, 46.96492, 44.30927, 61.79852,
    56.80963, 54.68502, 0.3708766 * 100, 0.4717574 * 100, 0.504364 * 100))
```

```r
ggplot(model_means, aes(x = Metric, y = MeanValue, fill = Model)) + geom_col(position = "dodge") +
    scale_fill_manual(values = c(Tree = "#D81B60", RF = "#1E88E5", Boosted = "#FFC107")) +
    labs(title = "Model Performance by Metric", y = "Mean value", x = NULL) + theme_minimal()
```



Conclusions

In this study, we applied three tree-based regression methods to predict diabetes progression using clinical data: pruned decision trees, random forests, and boosted trees. A 10-fold cross-validation procedure was used for evaluation, with parameter tuning performed separately within each training fold to ensure an unbiased comparison. This approach allowed for a fair and consistent assessment of the predictive performance of each model across different samples of the data.

The pruned decision tree provided a simple and interpretable model; however, its predictive performance was lower compared to the ensemble methods. Random forests improved upon this by combining many decision trees, each trained on bootstrapped samples and with random feature selection at each split. Boosted trees further enhanced predictive accuracy through their sequential learning process, which focuses on correcting the residuals of previous trees. As a result, boosting achieved the best results according to the evaluation metrics, showing the lowest mean absolute error (MAE) and root mean squared error (RMSE), as well as the highest R-squared values among the models.

Regarding variable importance, all models consistently highlighted body mass index (BMI) and triglyceride levels (TG) as the most influential predictors of diabetes progression. These were followed by blood pressure (BP) and glycemia (GC), suggesting that metabolic and cardiovascular indicators play a central role in the development of the disease. HDL cholesterol also showed a potentially protective effect, which aligns with existing clinical knowledge. Overall, while decision trees offer simplicity and ease of interpretation, ensemble methods, particularly gradient boosting, provided greater predictive accuracy and deeper insights into the key factors driving disease progression.