ADS 503 Project

Hunter Blum, Brianne Bell, Eva Chow

2022-06-21

### Libraries

library(haven)  
library(caret)  
library(gridExtra)  
library(corrplot)  
library(e1071)  
library(car)  
library(lattice)  
library(doParallel)  
library(RANN)  
library(rpart)  
library(party)  
library(partykit)  
library(rpart.plot)  
library(randomForest)  
library(RWeka)  
library(gbm)  
library(Cubist)  
  
library(tidyverse)

### Set up Parallelization

cl <- makeCluster(6)  
registerDoParallel(cl)

# Creating Dataset

### Read in data

Demographic <- read\_xpt("P\_DEMO.XPT")  
BodySize <- read\_xpt("P\_BMX.XPT")  
Chol\_ldl <- read\_xpt("P\_TRIGLY.XPT")

### Chol-ldl

#Select Variables of interest  
Chol\_ldl <- Chol\_ldl %>% select(SEQN, LBDLDL)  
#NA in target feature won't be useful  
Chol\_ldl <- Chol\_ldl %>% drop\_na()

### Demographic

#Get rid of variables we don't need  
Drop\_col <- c('SDDSRVYR', 'RIDSTATR', 'RIDEXMON', 'SIAPROXY', 'SIAINTRP', 'FIAPROXY', 'FIAINTRP', 'MIAPROXY', 'MIAINTRP', 'WTINTPRP', 'WTMECPRP', 'SDMVPSU', 'SDMVSTRA')  
Demographic <- Demographic %>% select(-one\_of(Drop\_col))

### BodySize

Drop\_col <- c('BMIWT', 'BMIRECUM', 'BMIHEAD', 'BMIHT', 'BMILEG', 'BMIARML', 'BMIARMC', 'BMIWAIST', 'BMIHIP', 'BMDSTATS')  
BodySize <- BodySize %>% select(-one\_of(Drop\_col))

### Join

J1 <- Chol\_ldl %>% left\_join(Demographic, by = "SEQN")  
Chol <- J1 %>% left\_join(BodySize, by = "SEQN")  
Chol <- Chol %>% select(!SEQN)

# Cleaning

## Changing factors for EDA

### Changing Variables to the Correct Type

Chol\_2 <- Chol  
factors <- c("RIAGENDR", "RIDRETH1", "RIDRETH3", "DMDBORN4", "DMDEDUC2", "DMDMARTZ", "RIDEXPRG", "SIALANG", "FIALANG", "MIALANG", "AIALANGA")  
Chol\_2[,factors] <- lapply(Chol\_2[,factors], factor)

### Change factor levels to be more interpretable

levels(Chol\_2$RIAGENDR) <- c("Male", "Female")  
levels(Chol\_2$RIDRETH1) <- c("Mex", "OHis", "White", "Black", "Oth")  
levels(Chol\_2$RIDRETH3) <- c("Mex", "OHis", "White", "Black", "Asian", "Oth")  
levels(Chol\_2$DMDBORN4) <- c("USA", "Oth", "Ref", "DK")  
levels(Chol\_2$DMDYRUSZ) <- c("<5", "5-15", "15-30", ">30", "Ref", "DK")  
levels(Chol\_2$DMDEDUC2) <- c("<9", "9-11", "HS", "AA", "BS+", "Ref", "DK")  
levels(Chol\_2$DMDMARTZ) <- c("Mar", "Sep", "Nev", "Ref", "DK")  
levels(Chol\_2$RIDEXPRG) <- c("Yes", "No", "DK")  
levels(Chol\_2$SIALANG) <- c("English", "Spanish")  
levels(Chol\_2$FIALANG) <- c("English", "Spanish")  
levels(Chol\_2$MIALANG) <- c("English", "Spanish")  
levels(Chol\_2$AIALANGA) <- c("English", "Spanish", "Asian")

# EDA

## NAs

### By Variable - Removed variables with over 3000 observations missing

Variable\_na <- Chol\_2 %>% select(everything()) %>% summarise\_all(funs(sum(is.na(.)))) %>% pivot\_longer(cols = c(colnames(Chol\_2[,1:ncol(Chol\_2)])), names\_to = "Variable", values\_to = "Missing") %>% arrange(desc(Missing))  
Drop\_col <- c("RIDAGEMN", "BMXRECUM", "BMXHEAD", "BMDBMIC", "RIDEXPRG", "DMDYRUSZ")  
Chol\_2 <- Chol\_2 %>% select(-one\_of(Drop\_col))

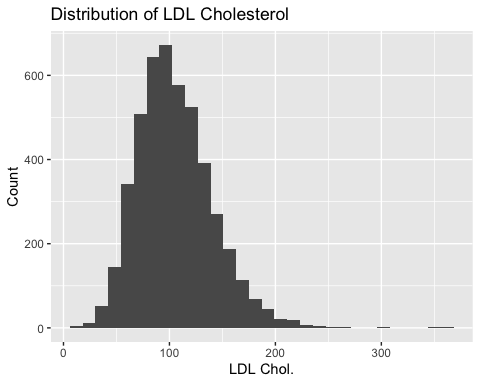
### By Row

row\_na <- rowSums(is.na(Chol\_2))  
row\_na <- data.frame(row\_na, Row = c(1:length(row\_na)))  
row\_na <- row\_na %>% arrange(desc(row\_na))  
#Most missing values in a row is 12, not bad

## Distributions

### Response - LDL Cholesterol

#Looks like a fairly normal distribution, maybe a little skewed to the right.   
ggplot(Chol\_2, aes(x = LBDLDL)) + geom\_histogram() + ggtitle("Distribution of LDL Cholesterol") + xlab("LDL Chol.") + ylab("Count")



skewness(Chol\_2$LBDLDL)

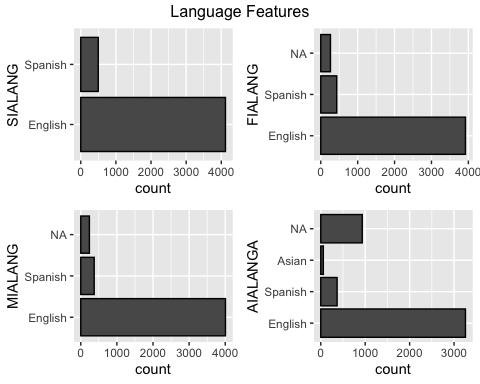
## [1] 0.7886403

#skewness value .7886403 confirms very mild skewness to the right

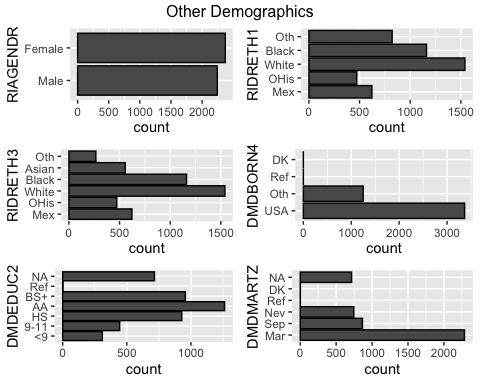
### Predictors

Factors

Chol\_fact <- Chol\_2 %>% select\_if(is.factor)  
Chol.bar <- function(xvar){  
 ggplot(Chol\_fact, aes\_(x = as.name(xvar))) +  
 geom\_bar(color = "black") + coord\_flip()  
}  
Lang\_barplots <- lapply(names(Chol\_fact[,7:10]), Chol.bar)  
Oth\_barplots <- lapply(names(Chol\_fact[,1:6]), Chol.bar)  
grid.arrange(grobs = Lang\_barplots, top = "Language Features")

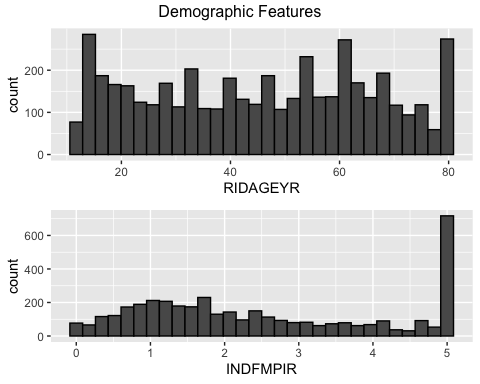


grid.arrange(grobs = Oth\_barplots, top = "Other Demographics")

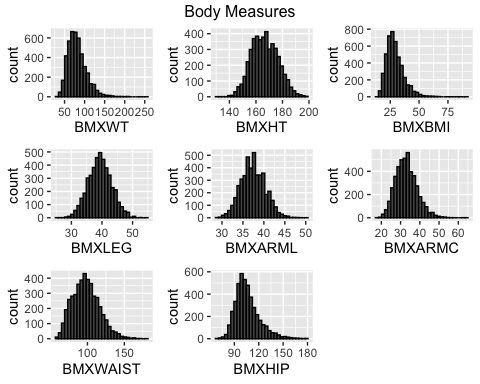


Numeric

Chol\_num <- Chol\_2 %>% select\_if(is.numeric) %>% select(!LBDLDL)  
Chol.hist <- function(xvar){  
 ggplot(Chol\_num, aes\_(x = as.name(xvar))) +  
 geom\_histogram(color = "black")   
}  
Dem\_hist <- lapply(names(Chol\_num[,1:2]), Chol.hist)  
Body\_hist <- lapply(names(Chol\_num[,3:10]), Chol.hist)  
grid.arrange(grobs = Dem\_hist, top = "Demographic Features")



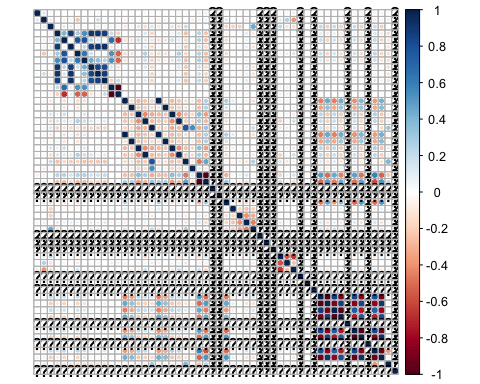
grid.arrange(grobs = Body\_hist, top = "Body Measures")



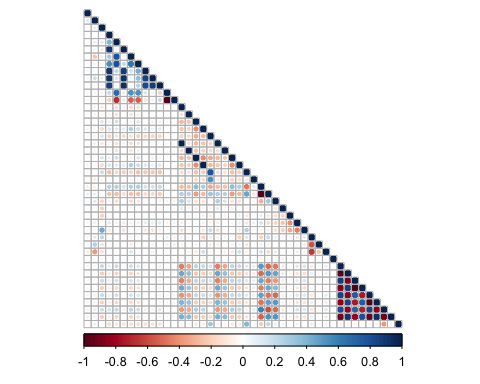
## Correlations

Heatmap

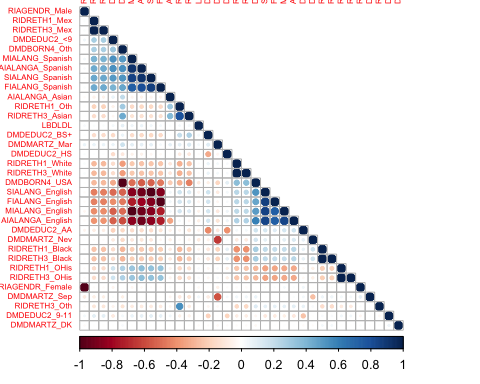
Chol\_dummy <- fastDummies::dummy\_cols(Chol\_2)  
Chol\_dummy <- Chol\_dummy %>% select\_if(~!is.factor(.))  
Chol\_dummy[] <- lapply(Chol\_dummy, as.numeric)  
Chol\_cor <- cor(Chol\_dummy, use = "complete.obs")  
Chol\_corplot <- corrplot(cor(Chol\_dummy, use = "complete.obs"), tl.pos = 'n')



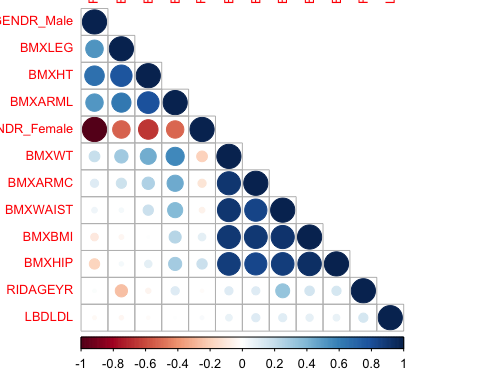
#Looks like some dummy variables that are refusal could be messing up correlations  
Drop\_col <- c("DMDBORN4\_Ref", "DMDBORN4\_DK", "DMDEDUC2\_Ref", "DMDEDUC2\_DK", "DMDEDUC2\_NA", "DMDMARTZ\_Ref", "DMDMARTZ\_NA", "FIALANG\_NA", "MIALANG\_NA", "AIALANGA\_NA")  
Chol\_dummy\_2 <- Chol\_dummy %>% select(-one\_of(Drop\_col))  
invisible(cor(Chol\_dummy\_2, use = "complete.obs")) # using invisible() to reduce extensive output  
corrplot(cor(Chol\_dummy\_2, use = "complete.obs"), tl.pos = 'n', type = 'lower')

 ### Smaller plots for easier interpretation.

# Mini Correlations: Sociological Measures:  
socio <- Chol\_dummy\_2[,c("RIAGENDR\_Male","RIAGENDR\_Female",  
 "RIDRETH1\_Mex", "RIDRETH1\_OHis", "RIDRETH1\_White",   
 "RIDRETH1\_Black", "RIDRETH1\_Oth", "RIDRETH3\_Mex", "RIDRETH3\_OHis",  
 "RIDRETH3\_White", "RIDRETH3\_Black", "RIDRETH3\_Asian", "RIDRETH3\_Oth",  
 "DMDBORN4\_USA", "DMDBORN4\_Oth", "DMDEDUC2\_<9", "DMDEDUC2\_9-11",  
 "DMDEDUC2\_HS", "DMDEDUC2\_AA", "DMDEDUC2\_BS+","DMDMARTZ\_Mar",  
 "DMDMARTZ\_Sep", "DMDMARTZ\_Nev", "DMDMARTZ\_DK", "SIALANG\_English",   
 "SIALANG\_Spanish", "FIALANG\_English", "FIALANG\_Spanish",  
 "MIALANG\_English","MIALANG\_Spanish", "AIALANGA\_English",  
 "AIALANGA\_Spanish", "AIALANGA\_Asian", "LBDLDL")]  
  
invisible(cor(socio, use = "complete.obs")) # using invisible() to reduce extensive output  
corrplot(cor(socio, use = "complete.obs"), tl.pos = 'y', type = 'lower',   
 order = "hclust", tl.cex = 0.5)



# Mini Correlations: Biological Measures:  
biologic <- Chol\_dummy\_2[,c("RIDAGEYR", "BMXWT", "BMXHT", "BMXBMI",   
 "BMXLEG", "BMXARML", "BMXARMC", "BMXWAIST",  
 "BMXHIP", "RIAGENDR\_Male", "RIAGENDR\_Female","LBDLDL")]  
invisible(cor(biologic, use = "complete.obs")) # using invisible() to reduce extensive output  
corrplot(cor(biologic, use = "complete.obs"), tl.pos = 'y', type = 'lower',   
 order = "hclust", tl.cex = 0.8)



# correlations between certain biological measures make sense. BMI is derived from the MASS and height of an individual, so it makes sense that many of the BMI measurements correlate with each other. (i.e. hip, waist, and weight measurements correlate with a higher BMI. Being female correlates negatively with leg and arm length as well as height)

### Highly-correlated variables in cholesterol

Note: Highly correlated variables were removed in the model training process - this was just for EDA

# let's check for highly correlated predictors  
# we'll do this on our non factor transformed dataset  
dim(Chol)

## [1] 4617 27

# 27 variables. let's find correlations greater than 0.80 and see how the data looks if removed  
corr\_Chol <- cor(Chol)  
  
# if removed, how many variables are left  
high\_corr\_Chol <- findCorrelation(corr\_Chol, cutoff = 0.80)  
no\_corr\_Chol <- Chol[, -high\_corr\_Chol]  
dim(no\_corr\_Chol)

## [1] 4617 26

### Looking at a simple ols model to get an idea of important predictors.

# looking at a base linear model, to see significant variables   
model0 <- lm(LBDLDL~., Chol\_2)  
invisible(summary(model0)) # invisible() used to reduce output. Key observations will be noted below:  
 # significant contributors: variable (Pr(>|t|))   
 # RIDAGEYR (0.000132)  
 # (Intercept) (0.043474)  
 # MDBORN4Oth (0.076414)   
 # AIALANGASpanish (0.041803)  
 # BMXLEG (0.033051)  
 # BMXARMC (0.004507)   
 # BMXWAIST (0.071888)

Looking at VIF of simple model showed aliased coefficients. Removed them in the model training process.

# looking at VIF for baseline linear:  
# vif(model0)  
# highly/perfectly correlated factors, we might need to drop some

### Degenerate Predictors in the non-dummy dataset

# Let's check for degenerate predictors from the original dataset  
nearZeroVar(Chol, saveMetrics = FALSE)

## [1] 4 11 16 18 19

deg\_chol <- subset(Chol, select=c(4,11,16,18,19))  
colnames(deg\_chol)

## [1] "RIDAGEMN" "RIDEXPRG" "INDFMPIR" "BMXRECUM" "BMXHEAD"

# Do it again on factor dataset  
nearZeroVar(Chol\_2, saveMetrics = FALSE)

## [1] 13

deg\_chol2 <- subset(Chol\_2, select=c(13))  
colnames(deg\_chol2)

## [1] "INDFMPIR"

# we may have to consider removing depending on the data used for modeling

# Preparing data for modeling

### Splitting dummy-variable data set and resampling

# set the seed and split the data. We'll do an 80/20 split  
set.seed(123)  
Chol\_split <- createDataPartition(Chol$LBDLDL, p=0.80, list=FALSE)  
  
# split into train and test  
Chol\_train <- Chol\_dummy[Chol\_split,]  
Chol\_test <- Chol\_dummy[-Chol\_split,]  
  
# split predictors from the target  
Chol\_train\_X <- as.data.frame(subset(Chol\_train, select=-c(LBDLDL)))   
Chol\_train\_y <- Chol\_train$LBDLDL  
  
Chol\_test\_X <- as.data.frame(subset(Chol\_test, select=-c(LBDLDL)))  
Chol\_test\_y <- Chol\_test$LBDLDL  
  
# Creating imputed data sets  
Chol\_imp <- preProcess(Chol\_train\_X, method = c("center", "scale", "knnImpute"))  
Chol\_train\_X\_imp <- predict(Chol\_imp, Chol\_train\_X)  
Chol\_test\_X\_imp <- predict(Chol\_imp, Chol\_test\_X)  
  
# Adding Resampling/Validation Set and Control   
set.seed(123)  
Chol\_folds <- createFolds(y = Chol\_train\_X, k = 10, returnTrain = T)  
Chol\_control <- trainControl(method = "cv", index = Chol\_folds)

### Numeric training data set with just the numeric variables and gender (Just going off a hunch that having too many dummy variables is hurting linear model performance).

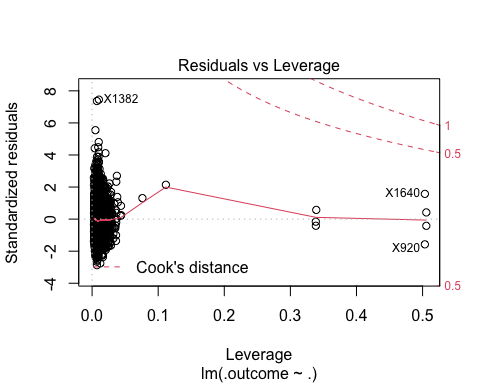
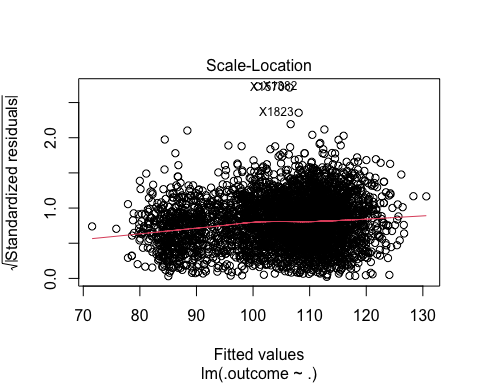
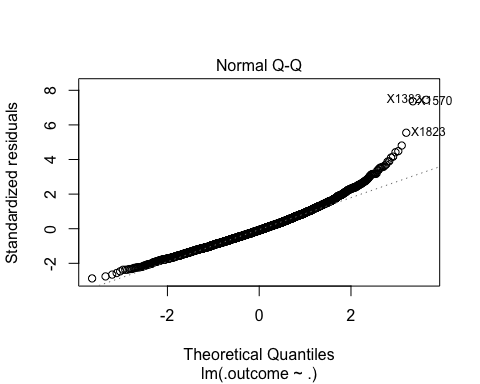
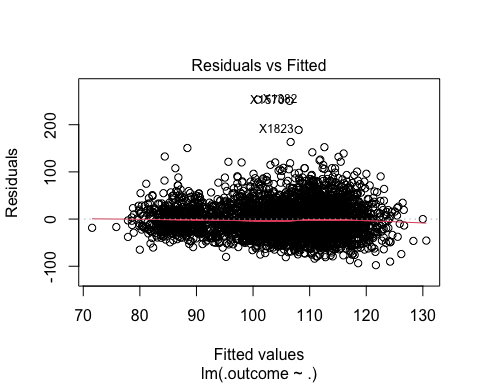
Drop\_col <- c('RIDRETH1', 'RIDRETH3', 'DMDBORN4', 'DMDEDUC2', 'DMDMARTZ', 'SIALANG', 'FIALANG', 'MIALANG', 'AIALANGA', 'LBDLDL')  
Chol\_num <- Chol\_2 %>% select(-one\_of(Drop\_col))  
Chol\_dummy <- fastDummies::dummy\_cols(Chol\_num)  
Chol\_num <- Chol\_dummy %>% select\_if(~!is.factor(.))  
Chol\_num[] <- lapply(Chol\_num, as.numeric)  
  
Chol\_num\_tr\_X <- as.data.frame(Chol\_num[Chol\_split, ])  
Chol\_num\_test\_X <- as.data.frame(Chol\_num[-Chol\_split, ])  
  
#Preprocess  
Chol\_imp <- preProcess(Chol\_num\_tr\_X, method = c("center", "scale", "knnImpute"))  
Chol\_num\_tr\_X <- predict(Chol\_imp, Chol\_num\_tr\_X)  
Chol\_num\_test\_X <- predict(Chol\_imp, Chol\_num\_test\_X)  
  
# Adding Resampling/Validation Set and Control   
set.seed(123)  
Chol\_folds\_num <- createFolds(y = Chol\_num\_tr\_X, k = 10, returnTrain = T)  
Chol\_control\_num <- trainControl(method = "cv", index = Chol\_folds\_num)

# Linear Models - Hunter

## OLS

### Create Initial Model

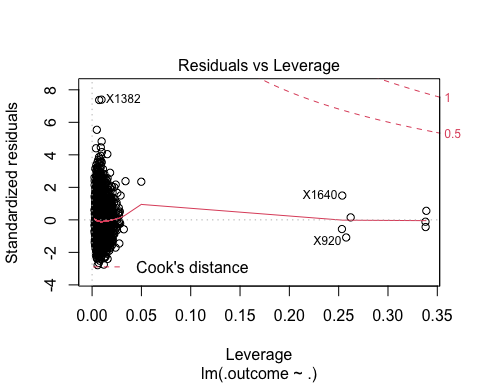
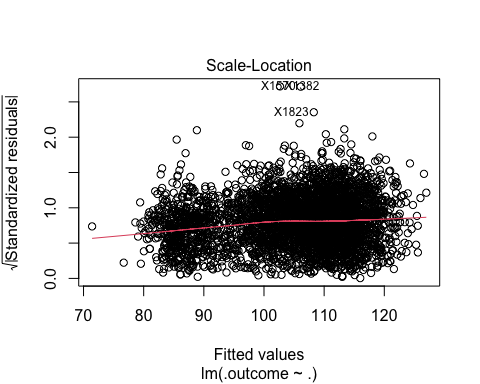
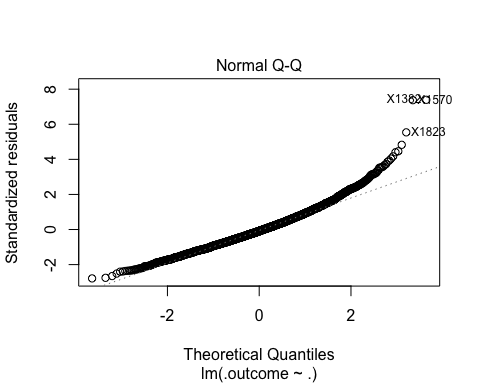
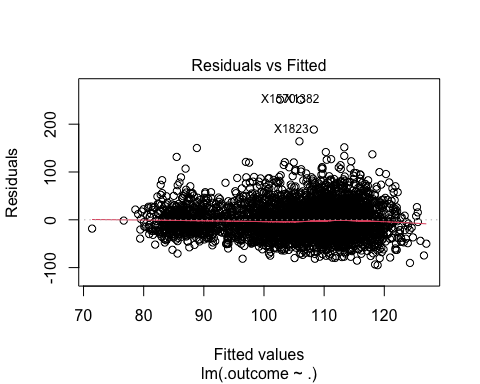
Chol\_ols\_tune <- train(x = Chol\_train\_X\_imp, y = Chol\_train\_y, method = "lm", trControl = Chol\_control)  
plot(Chol\_ols\_tune$finalModel)

 ### FIX TRAINING SET

# VIF shows aliased coefficients, need to get rid of those by removing high cor predictors  
test <- cor(Chol\_train\_X\_imp)  
# Also have an issue with DMDEDUC2\_DK all being zero so get rid of high var predictors  
Chol\_tr\_x\_imp\_vr <- Chol\_train\_X\_imp[, -nearZeroVar(Chol\_train\_X\_imp)]  
Chol\_tr\_X\_imp\_fin <- Chol\_tr\_x\_imp\_vr[, -findCorrelation(cor(Chol\_tr\_x\_imp\_vr), cutoff = 0.9)]

### Tune Another Model

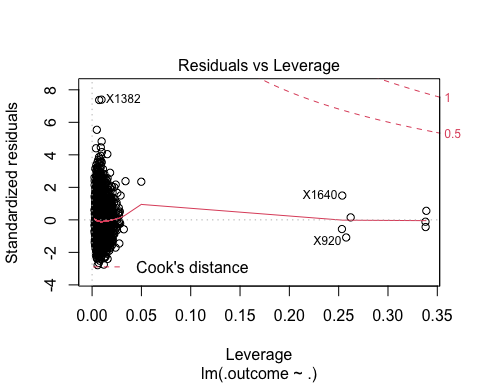
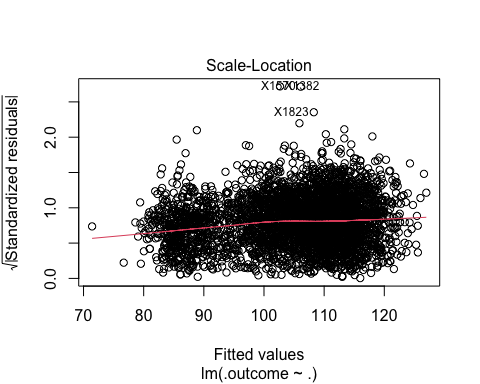
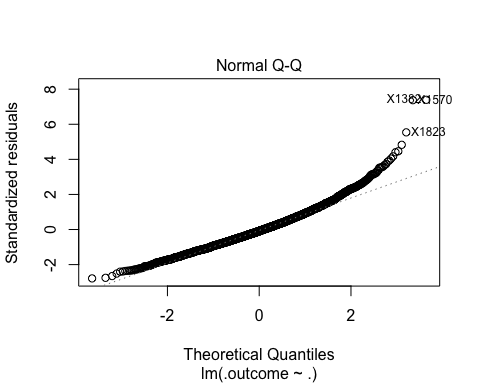
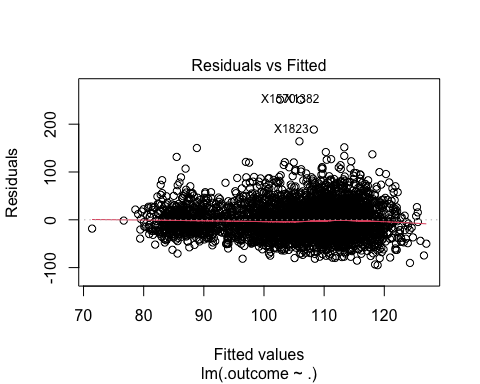
Chol\_ols\_tune2 <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "lm", trControl = Chol\_control)  
plot(Chol\_ols\_tune2$finalModel)



invisible(summary(Chol\_ols\_tune2$finalModel)) # invisible used to minimize output clutter. Key observations to be noted in final report

### Tune model with BoxCox to see if it will help normality issues - Didn’t do much, we’ll just stick with the non-transformed data. Also tried transformin LDL, didn’t work either.

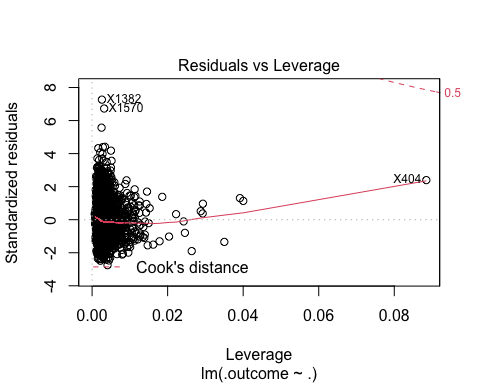
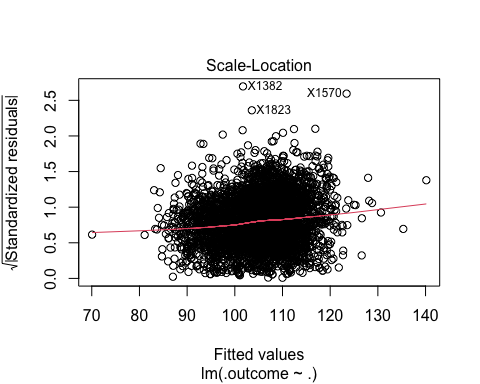
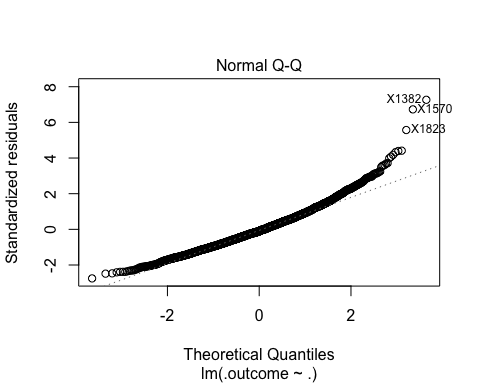
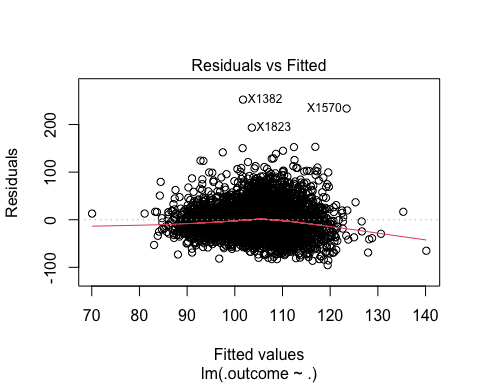
Chol\_bct <- preProcess(Chol\_tr\_X\_imp\_fin, method = "BoxCox")  
Chol\_tr\_boxcox <- predict(Chol\_bct, Chol\_tr\_X\_imp\_fin)  
  
Chol\_ols\_tune3 <- train(x = Chol\_tr\_boxcox, y = Chol\_train\_y, method = "lm", trControl = Chol\_control)  
plot(Chol\_ols\_tune3$finalModel)



invisible(summary(Chol\_ols\_tune3$finalModel)) #invisible used to reduce cluttered output. Key observations noted in final report

### Try reduced data - Didn’t really help our diagnostic plot, so we’ll go with the regular dummy data

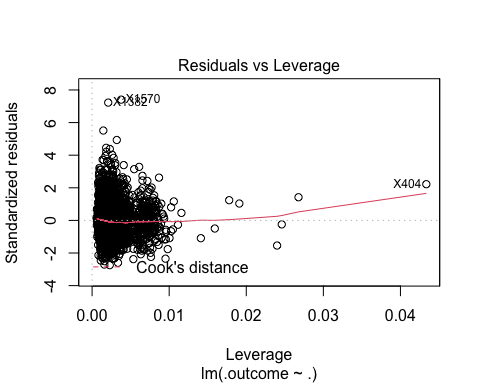
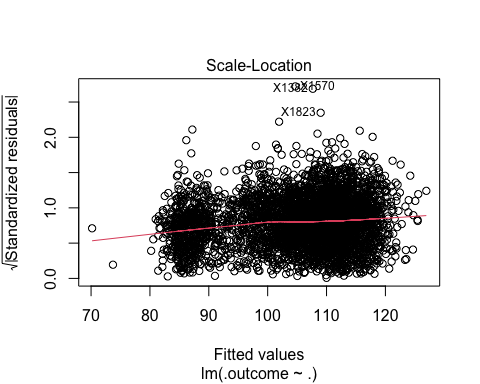
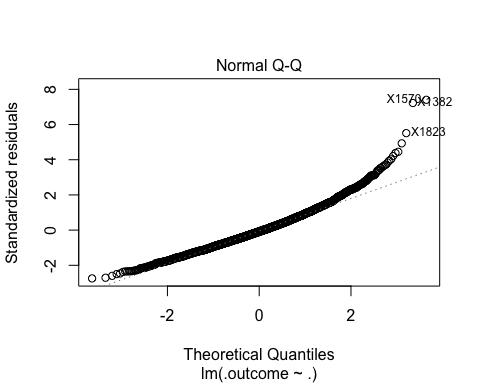
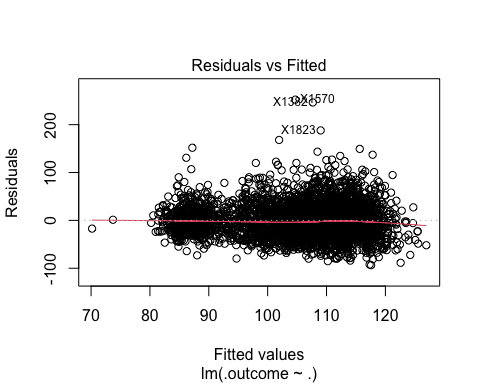
Chol\_ols\_tune\_num <- train(x = Chol\_num\_tr\_X, y = Chol\_train\_y, method = "lm", trControl = Chol\_control\_num)  
plot(Chol\_ols\_tune\_num$finalModel)



invisible(summary(Chol\_ols\_tune\_num$finalModel)) # hiding long output to reduce clutter. Key observations to be noted in final report

### Final OLS Model

Chol\_sig\_tr <- Chol\_tr\_X\_imp\_fin %>% select(RIDAGEYR, BMXHT, BMXBMI, BMXLEG, BMXARMC, DMDBORN4\_Oth, DMDEDUC2\_NA, MIALANG\_NA, AIALANGA\_NA)  
Chol\_ols <- train(x = Chol\_sig\_tr, y = Chol\_train\_y, method = "lm", trControl = Chol\_control)  
plot(Chol\_ols$finalModel)



invisible(summary(Chol\_ols$finalModel)) # hidden to reduce output clutter. Key observations to be noted  
  
#Predict on test data  
Chol\_ols\_res <- predict(Chol\_ols, Chol\_test\_X)

## PCR and PLS

### PCR

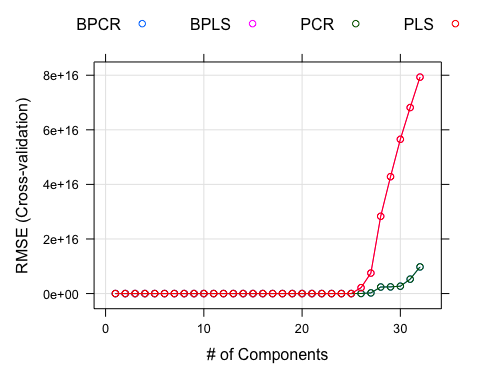
set.seed(123)  
Chol\_pcr <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "pcr", tuneGrid = expand.grid(ncomp=1:32), trControl = Chol\_control)  
invisible(Chol\_pcr) # output hidden in final report to reduce clutter  
  
set.seed(123)  
Chol\_pcr\_box <- train(x = Chol\_tr\_boxcox, y = Chol\_train\_y, method = "pcr", tuneGrid = expand.grid(ncomp=1:32), trControl = Chol\_control)  
invisible(Chol\_pcr\_box) # output hidden in final report to reduce clutter  
  
set.seed(123)  
Chol\_pcr\_num <- train(x = Chol\_num\_tr\_X, y = Chol\_train\_y, method = "pcr", tuneGrid = expand.grid(ncomp=1:8), trControl = Chol\_control\_num)  
invisible(Chol\_pcr\_num) # output hidden in final report to reduce clutter  
  
pcr\_resamp <- Chol\_pcr$results  
pcr\_resamp$Model <- "PCR"  
  
box\_pcr\_resamp <- Chol\_pcr\_box$results  
box\_pcr\_resamp$Model <- "BPCR"  
  
num\_pcr\_resamp <- Chol\_pcr\_num$results  
num\_pcr\_resamp$Model <- "PCR"  
# key observations and summary to be noted in final report

### PLS

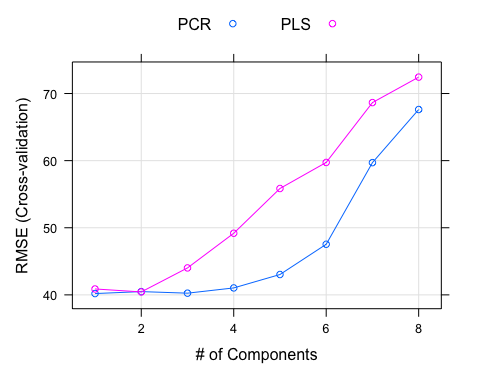
set.seed(123)  
Chol\_pls <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "pls", tuneGrid = expand.grid(ncomp = 1:32), trControl = Chol\_control)  
invisible(Chol\_pls) # output hidden to reduce clutter  
  
set.seed(123)  
Chol\_pls\_box <- train(x = Chol\_tr\_boxcox, y = Chol\_train\_y, method = "pls", tuneGrid = expand.grid(ncomp = 1:32), trControl = Chol\_control)  
invisible(Chol\_pls\_box) # output hidden to reduce clutter  
  
set.seed(123)  
Chol\_pls\_num <- train(x = Chol\_num\_tr\_X, y = Chol\_train\_y, method = "pls", tuneGrid = expand.grid(ncomp = 1:8), trControl = Chol\_control\_num)  
invisible(Chol\_pls\_num) # output hidden to reduce clutter  
  
pls\_resamp <- Chol\_pls$results  
pls\_resamp$Model <- "PLS"  
  
pls\_box\_resamp <- Chol\_pls\_box$results  
pls\_box\_resamp$Model <- "BPLS"  
  
pls\_num\_resamp <- Chol\_pls\_num$results  
pls\_num\_resamp$Model <- "PLS"

### Compare

plot\_data <- rbind(pcr\_resamp, box\_pcr\_resamp, pls\_resamp, pls\_box\_resamp)  
xyplot(RMSE ~ ncomp, data = plot\_data, xlab = "# of Components", ylab = "RMSE (Cross-validation)", auto.key = list(columns = 4), groups = Model, type = c("o", "g"))



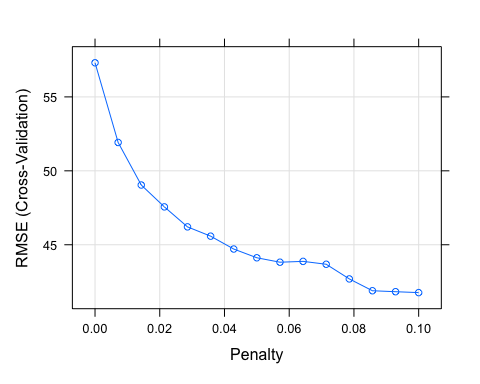
plot2\_data <- rbind(num\_pcr\_resamp, pls\_num\_resamp)  
xyplot(RMSE ~ ncomp, data = plot2\_data, xlab = "# of Components", ylab = "RMSE (Cross-validation)", auto.key = list(columns = 2), groups = Model, type = c("o", "g"))



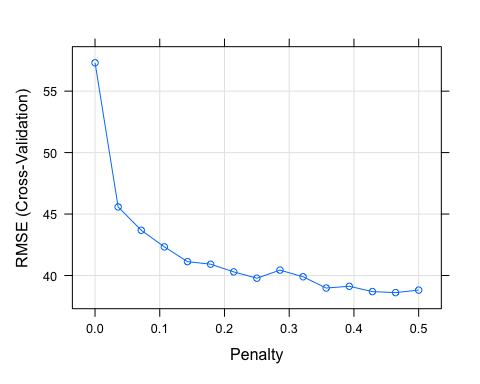
## Penalized Models

### Ridge

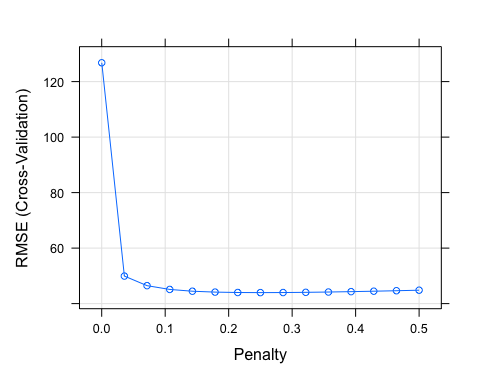
set.seed(123)  
Chol\_ridge <- train(x = Chol\_tr\_X\_imp\_fin, y= Chol\_train\_y, method = "ridge", tuneGrid = expand.grid(lambda = seq(0, .1, length = 15)), trControl = Chol\_control)  
invisible(Chol\_ridge) # model output hidden to reduce clutter  
  
set.seed(123)  
Chol\_ridge\_box <- train(x = Chol\_tr\_boxcox, y= Chol\_train\_y, method = "ridge", tuneGrid = expand.grid(lambda = seq(0, .5, length = 15)), trControl = Chol\_control)  
invisible(Chol\_ridge\_box) # model output hidden to reduce clutter  
  
set.seed(123)  
Chol\_ridge\_num <- train(x = Chol\_num\_tr\_X, y= Chol\_train\_y, method = "ridge", tuneGrid = expand.grid(lambda = seq(0, .5, length = 15)), trControl = Chol\_control\_num)  
invisible(Chol\_ridge\_num) # model output hidden to reduce clutter  
  
print(update(plot(Chol\_ridge), xlab = "Penalty"))



print(update(plot(Chol\_ridge\_box), xlab = "Penalty"))

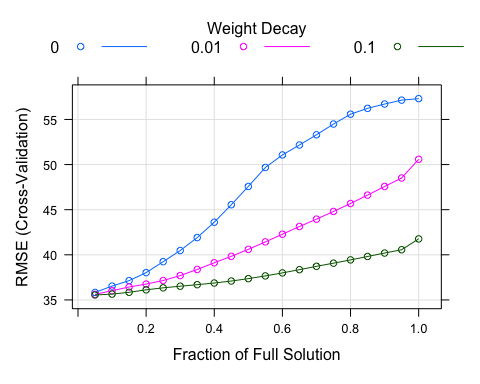


print(update(plot(Chol\_ridge\_num), xlab = "Penalty"))

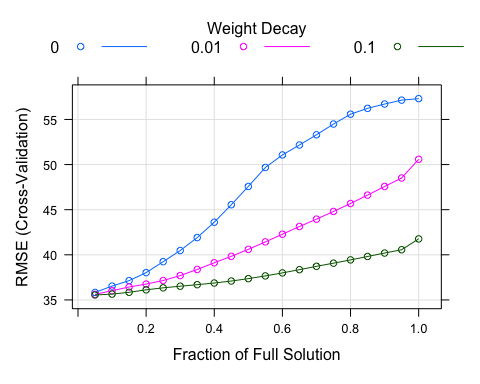


### Elastic Net

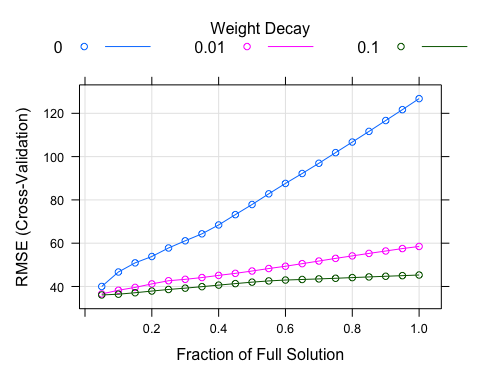
enet\_grid <- expand.grid(lambda = c(0, 0.01, 0.1), fraction = seq(0.05, 1, length = 20))  
  
Chol\_enet <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "enet", tuneGrid = enet\_grid, trControl = Chol\_control)  
invisible(Chol\_enet) # model output hidden to reduce clutter  
  
Chol\_enet\_box <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "enet", tuneGrid = enet\_grid, trControl = Chol\_control)  
invisible(Chol\_enet\_box) # model output hidden to reduce clutter  
  
Chol\_enet\_num <- train(x = Chol\_num\_tr\_X, y = Chol\_train\_y, method = "enet", tuneGrid = enet\_grid, trControl = Chol\_control\_num)  
  
plot(Chol\_enet)



plot(Chol\_enet\_box)



plot(Chol\_enet\_num)



## Gather Results from Linear Models

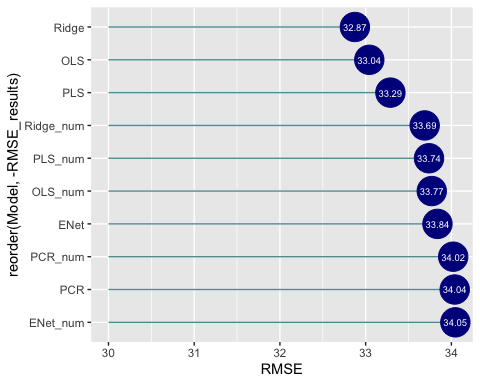
Note: No diagnostic plot showed much of a difference between boxcox vs. regular (not suprising given normality in histograms), so we’ll just use the regular constructed models

### Create Data set of predictions and observed

Res\_OLS <- predict(Chol\_ols, Chol\_test\_X\_imp)  
Res\_OLS\_num <- predict(Chol\_ols\_tune\_num, Chol\_num\_test\_X)  
Res\_PLS <- predict(Chol\_pls, Chol\_test\_X\_imp)  
Res\_PLS\_num <- predict(Chol\_pls\_num, Chol\_num\_test\_X)  
Res\_PCR <- predict(Chol\_pcr, Chol\_test\_X\_imp)  
Res\_PCR\_num <- predict(Chol\_pcr\_num, Chol\_num\_test\_X)  
Res\_Ridge <- predict(Chol\_ridge, Chol\_test\_X\_imp)  
Res\_Ridge\_num <- predict(Chol\_ridge\_num, Chol\_num\_test\_X)  
Res\_Enet <- predict(Chol\_enet, Chol\_test\_X\_imp)  
Res\_Enet\_num <- predict(Chol\_enet\_num, Chol\_num\_test\_X)  
  
Linear\_res <- cbind.data.frame(Observed = Chol\_test\_y, OLS = Res\_OLS, OLS\_num = Res\_OLS\_num, PLS = Res\_PLS, PLS\_num = Res\_PLS\_num, PCR = Res\_PCR, PCR\_num = Res\_PCR\_num ,Ridge = Res\_Ridge, Ridge\_num = Res\_Ridge\_num, ENet = Res\_Enet, ENet\_num = Res\_Enet\_num)

### Get RMSE and Plot

find\_rmse <- function(x){  
 caret::RMSE(x, Linear\_res[,"Observed"])  
}  
  
RMSE\_results <- apply(X = Linear\_res[,2:11], FUN = find\_rmse, MARGIN = 2)  
RMSE\_results <- data.frame(RMSE\_results)  
RMSE\_results$Model <- rownames(RMSE\_results)  
  
  
ggplot(RMSE\_results, aes(x=reorder(Model, -RMSE\_results), y=RMSE\_results)) + geom\_segment(aes(x=reorder(Model, -RMSE\_results), xend = reorder(Model, -RMSE\_results), y=30, yend=RMSE\_results), color = "cadetblue") + geom\_point(color = "darkblue", size = 10) + coord\_flip() + ylab("RMSE") + geom\_text(aes(label = round(RMSE\_results, 2)), color = "white", size = 2.5)

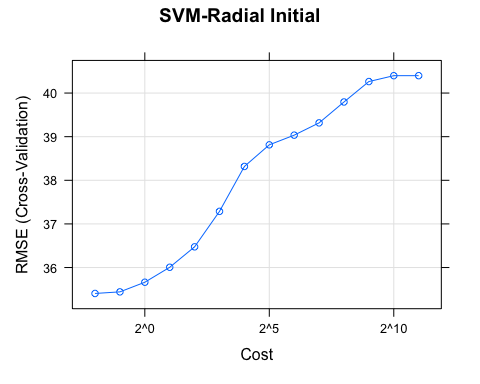


# Non-Linear Models - Brianne

## Support Vector Machine (SVM)

### Create Initial Radial Model

# initial SVM model with radial basis and processed Chol\_tr\_X\_imp\_fin and Chol\_train\_y  
set.seed(123)  
svmR0 <- train(x=Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y,  
 method = "svmRadial",  
 preProcess = c("center", "scale"),  
 tuneLength = 14,  
 trControl = Chol\_control)  
invisible(svmR0) # model output hidden to reduce clutter. Key observations noted below:  
 # final model uses: sigma = 0.0207376 and C = 0.25  
 # RMSE: 35.40559, Rsquared: 0.019455264  
plot(svmR0, scales = list(x = list(log = 2)), main="SVM-Radial Initial")



### final radial model

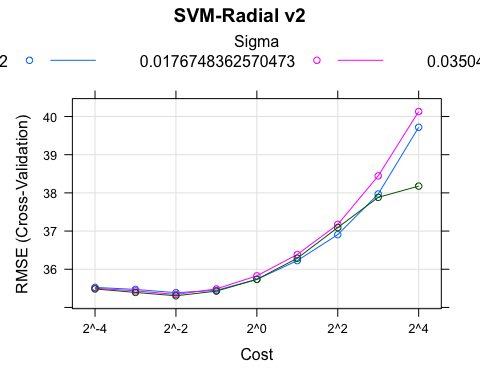
# svm radial model v2   
 # issue causing variables in X: MIALANG\_Spanish, DMDEDUC2\_<9, MIALANG\_NA (zero var)  
Chol\_tr\_X\_impfin\_drop <- c("MIALANG\_Spanish", "DMDEDUC2\_<9", "MIALANG\_NA")  
Chol\_tr\_X\_impfin\_sv <- subset(Chol\_tr\_X\_imp\_fin,   
 select = !(names(Chol\_tr\_X\_imp\_fin) %in% Chol\_tr\_X\_impfin\_drop))  
  
#making test X have same columns available  
Chol\_te\_X\_sv <- subset(Chol\_test\_X\_imp, select = c(names(Chol\_tr\_X\_impfin\_sv)))  
dim(Chol\_tr\_X\_impfin\_sv)

## [1] 3696 29

dim(Chol\_te\_X\_sv)

## [1] 921 29

# sigma grid instead of using tuneLength = 14  
sigmaEst <- kernlab::sigest(as.matrix(Chol\_tr\_X\_impfin\_sv[,1:29]))  
Csearch <- 2^seq(-4,+4)  
# sigma estimates using kernlab's sigest function  
svmgrid <- expand.grid(sigma = sigmaEst, C = Csearch)  
  
#model  
set.seed(123)  
svmR1 <- train(x=Chol\_tr\_X\_impfin\_sv, y = Chol\_train\_y,  
 method = "svmRadial",  
 preProcess = c("center", "scale"),  
 tuneGrid = svmgrid,  
 trControl = Chol\_control)  
invisible(svmR1) # model output hidden to reduce clutter. Key observations noted below:  
 # final model uses: sigma = 0.03504524 and C = 0.25.  
 # RMSE: 35.30443, Rsquared: 0.020245225   
plot(svmR1, scales = list(x = list(log = 2)), main="SVM-Radial v2")

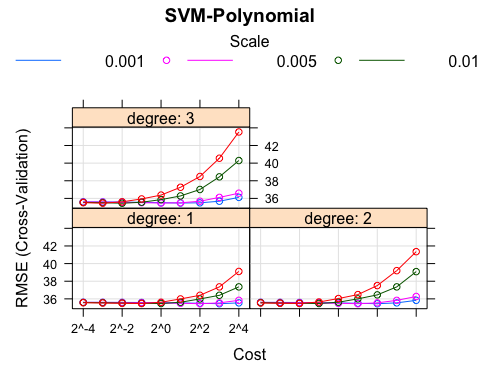


svmR1$finalModel

## Support Vector Machine object of class "ksvm"   
##   
## SV type: eps-svr (regression)   
## parameter : epsilon = 0.1 cost C = 0.25   
##   
## Gaussian Radial Basis kernel function.   
## Hyperparameter : sigma = 0.035045237945048   
##   
## Number of Support Vectors : 3377   
##   
## Objective Function Value : -557.6121   
## Training error : 0.82428

### Create Polynomial Model

# going to use the x training set from final svm-radial due to assuming there will be the same problem causing factors of nearZeroVar.  
set.seed(123)  
svmP <- train(x=Chol\_tr\_X\_impfin\_sv, y = Chol\_train\_y,  
 method = "svmPoly",  
 preProcess = c("center", "scale"),  
 tuneGrid = expand.grid(degree = 1:3,   
 scale = c(0.01, 0.005, 0.001, 0.0005),   
 C = Csearch),  
 trControl = Chol\_control)  
invisible(svmP) # model output hidden to reduce clutter. Key observations noted below:  
 # final model uses: degree = 2, scale = 0.001, offset = 1   
 # sigma = 0.02231109 and C = 2.  
 # RMSE: 35.44929, Rsquared: 0.011746076  
plot(svmP, scales = list(x = list(log = 2),  
 between=list(x=.5, y=1)), main="SVM-Polynomial")



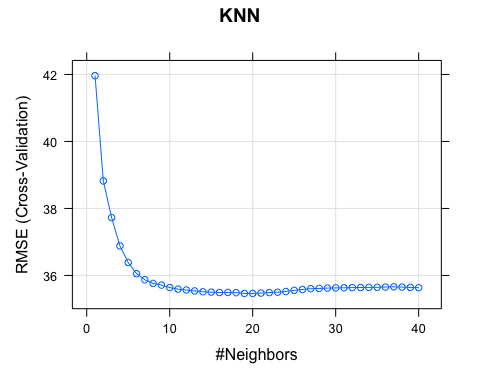
svmP$finalModel

## Support Vector Machine object of class "ksvm"   
##   
## SV type: eps-svr (regression)   
## parameter : epsilon = 0.1 cost C = 2   
##   
## Polynomial kernel function.   
## Hyperparameters : degree = 2 scale = 0.001 offset = 1   
##   
## Number of Support Vectors : 3366   
##   
## Objective Function Value : -4741.157   
## Training error : 0.932111

## K Nearest Neighbors (KNN)

### Create Initial KNN Model

# KNN Model needs to have NZV removed so again we are using the x=Chol\_tr\_X\_impfin\_sv to train and Chol\_te\_X\_sv to test  
set.seed(123)  
knnTune <- train(x=Chol\_tr\_X\_impfin\_sv, y = Chol\_train\_y,  
 method = "knn",  
 preProcess = c("center", "scale"),  
 tuneGrid = data.frame(k=1:40),  
 trControl = Chol\_control)  
invisible(knnTune) # model output hidden to reduce clutter. Key observations noted below:  
 # final model uses: k=40  
 # RMSE: 35.63394, Rsquared: 0.0006650456  
plot(knnTune, main="KNN")



knnTune$finalModel

## 20-nearest neighbor regression model

## Multivariate Adaptive Regression Splines (MARS)

### Create Initial MARS Model

# MARS model doesn't need preprocessing, so first rendition will be with Chol\_tr\_X\_imp\_fin   
set.seed(123)  
mars1 <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y,  
 method = "earth",  
 tuneGrid = expand.grid(degree = 1:3, nprune = 2:38),  
 trControl = Chol\_control)

## Loading required package: earth

## Loading required package: Formula

## Loading required package: plotmo

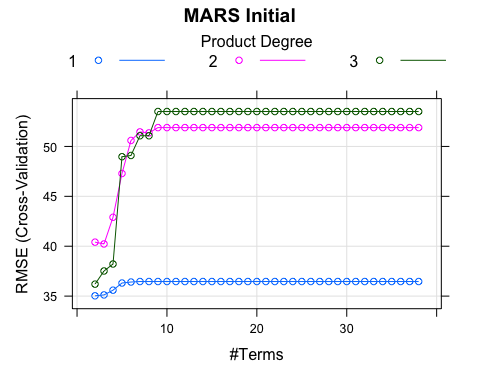
## Loading required package: plotrix

## Loading required package: TeachingDemos

mars1$finalModel

## Selected 2 of 18 terms, and 1 of 32 predictors (nprune=2)  
## Termination condition: RSq changed by less than 0.001 at 18 terms  
## Importance: RIDAGEYR, INDFMPIR-unused, BMXHT-unused, BMXBMI-unused, ...  
## Number of terms at each degree of interaction: 1 1 (additive model)  
## GCV 1192.459 RSS 4400176 GRSq 0.05246562 RSq 0.05349109

invisible(mars1$results) # model results hidden to reduce clutter. Key observations noted below:  
 #used 1 of 32 predictors, 2 of 18 terms (nprune =2) degree=1  
 # RMSE: 35.02993, Rsquared: 0.05354368   
plot(mars1, main="MARS Initial")



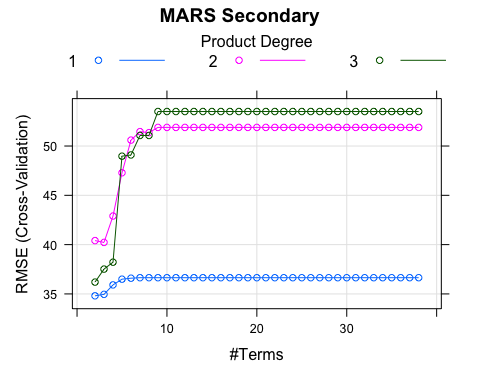
## Multivariate Adaptive Regression Splines (MARS)

### Create Secondary MARS Model

# MARS model using same X sets as SVM models:   
set.seed(123)  
mars2 <- train(x = Chol\_tr\_X\_impfin\_sv, y = Chol\_train\_y,  
 method = "earth",  
 tuneGrid = expand.grid(degree = 1:3, nprune = 2:38),  
 trControl = Chol\_control)  
mars2$finalModel

## Selected 2 of 18 terms, and 1 of 29 predictors (nprune=2)  
## Termination condition: RSq changed by less than 0.001 at 18 terms  
## Importance: RIDAGEYR, INDFMPIR-unused, BMXHT-unused, BMXBMI-unused, ...  
## Number of terms at each degree of interaction: 1 1 (additive model)  
## GCV 1192.459 RSS 4400176 GRSq 0.05246562 RSq 0.05349109

plot(mars2, main="MARS Secondary")

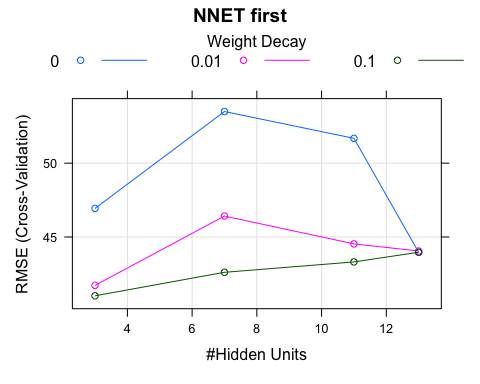


# No change between the two MARS models. Drops all but two factors for both.  
 #nprune=2, degree=1, RMSE: 34.80857, Rsquared: 0.05457991

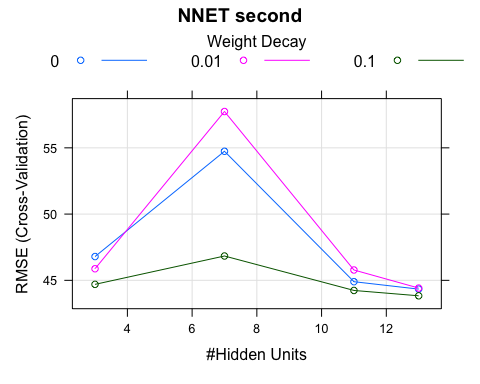
## Neural Network Model (nnet)

### Create Initial NNET Model

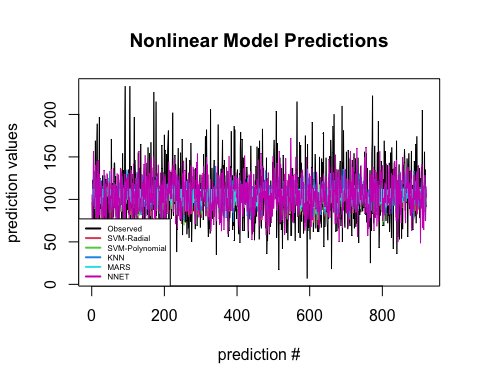
set.seed(123)  
nnetGrid <- expand.grid(decay = c(0, 0.01, .1), size = c(3, 7, 11, 13))  
# NNET first rendition will be with Chol\_tr\_X\_imp\_fin   
set.seed(100)  
nnet1 <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y,  
 method = "nnet",  
 tuneGrid = nnetGrid,  
 trControl = Chol\_control,  
 preProc = c("center", "scale"),  
 linout = TRUE,  
 trace = FALSE,  
 MaxNWts = 13 \* (ncol(Chol\_tr\_X\_imp\_fin) + 1) + 13 + 1,  
 maxit = 100)  
invisible(nnet1) # model output hidden to reduce clutter. Key observations noted below:  
 # size=3, decay=0.1  
 # RMSE: 40.97813, Rsquared: 0.0051328366  
plot(nnet1, main="NNET first")



set.seed(123)  
# NNET second rendition will be with Chol\_tr\_X\_impfin\_sv   
set.seed(100)  
nnet2 <- train(x = Chol\_tr\_X\_impfin\_sv, y = Chol\_train\_y,  
 method = "nnet",  
 tuneGrid = nnetGrid,  
 trControl = Chol\_control,  
 preProc = c("center", "scale"),  
 linout = TRUE,  
 trace = FALSE,  
 MaxNWts = 13 \* (ncol(Chol\_tr\_X\_impfin\_sv) + 1) + 13 + 1,  
 maxit = 100)  
invisible(nnet2) # model output hidden to reduce clutter. Key observations noted below:  
 # size=13, decay=0.1  
 # RMSE: 44.38061, Rsquared: 0.001743654  
plot(nnet2, main="NNET second")

 ## Comparing Nonlinear Models: ### Saving Results

NonLpred <- data.frame(obs=Chol\_test\_y)  
NonLpred$svmR <- predict(svmR1, Chol\_te\_X\_sv)  
NonLpred$svmP <- predict(svmP, Chol\_te\_X\_sv)  
NonLpred$KNN <- predict(knnTune, Chol\_te\_X\_sv)  
NonLpred$MARS2 <- predict(mars2, Chol\_te\_X\_sv)  
NonLpred$NNET2 <- predict(nnet2, Chol\_te\_X\_sv)  
plotpred <- data.frame(x=1:921, y1=NonLpred$obs, y2=NonLpred$svmR,   
 y3=NonLpred$svmP, y4=NonLpred$KNN,   
 y5=NonLpred$MARS2[,"y"], y6=NonLpred$NNET2)  
plot(plotpred$x, plotpred$y1, type = "l", col = 1,   
 xlab = "prediction #", ylab = "prediction values",   
 main = "Nonlinear Model Predictions")  
lines(plotpred$x, plotpred$y2, col = 2)  
lines(plotpred$x, plotpred$y3, col = 3)  
lines(plotpred$x, plotpred$y4, col = 4)  
lines(plotpred$x, plotpred$y5, col = 5)  
lines(plotpred$x, plotpred$y6, col = 6)  
legend("bottomleft", cex=0.5, legend = c("Observed", "SVM-Radial",  
 "SVM-Polynomial", "KNN", "MARS", "NNET"),  
 col = 1:6, lwd = 2)

 ### Getting RMSE and Plotting

# RMSE = sqrt(sum((obs-pred)^2)/n), n=921  
getRMSE <- function(x,y) {  
 sqrt(sum((x-y)^2)/length(x))  
}  
nonlin\_rmse <- data.frame(c("svmRad", "svmPoly", "KNN",  
 "MARS", "NNet"))  
nonlin\_rmse$RMSE <- c(getRMSE(NonLpred$obs, NonLpred$svmR),  
 getRMSE(NonLpred$obs, NonLpred$svmP),  
 getRMSE(NonLpred$obs, NonLpred$KNN),  
 getRMSE(NonLpred$obs, NonLpred$MARS2),  
 getRMSE(NonLpred$obs, NonLpred$NNET2))  
colnames(nonlin\_rmse)[1]<-"Model Type"  
nonlin\_rmse[order(nonlin\_rmse$RMSE),]

## Model Type RMSE  
## 1 svmRad 32.94845  
## 2 svmPoly 33.03741  
## 4 MARS 33.52618  
## 3 KNN 33.85140  
## 5 NNet 36.94634

# best non linear model is svm radial with   
 # sigma = sigma = 0.03504524 and C = 0.25. (RMSE is 32.95)

### Messing around with predictor set and log transforming the response

expDropC <- c("RIDRETH1\_Black", "RIDRETH1\_Mex", "RIDRETH1\_OHis",   
 "RIDRETH1\_Oth", "RIDRETH1\_White", "BMXBMI")  
Xtrial\_tr <- subset(Chol\_train\_X\_imp, select = !(names(Chol\_train\_X\_imp) %in% expDropC))  
# looking for near zero var:  
Xtri\_nzv <- nearZeroVar(Xtrial\_tr)  
 # "DMDBORN4\_Ref", "DMDBORN4\_DK", "DMDEDUC2\_Ref", "DMDEDUC2\_DK", "DMDMARTZ\_Ref",  
 # "DMDMARTZ\_DK", "AIALANGA\_Asian"  
Xtri\_tr\_nz <- subset(Xtrial\_tr, select = -c(Xtri\_nzv))  
print(paste("Xtrial\_tr ncol: ", ncol(Xtrial\_tr), " NZV removed, new ncol: ", ncol(Xtri\_tr\_nz)))

## [1] "Xtrial\_tr ncol: 47 NZV removed, new ncol: 40"

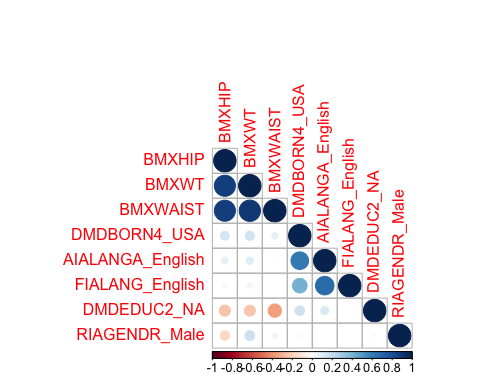
# dropping SIALANG groups (sample person interview instrument lang)  
expDropC <- c("SIALANG\_English", "SIALANG\_Spanish",   
 "MIALANG\_English", "MIALANG\_Spanish", "MIALANG\_NA")  
Xtri\_tr\_nz <- subset(Xtri\_tr\_nz, select = !(names(Xtri\_tr\_nz) %in% expDropC))  
print(paste("Xtrial\_tr ncol: ", ncol(Xtrial\_tr), " NZV removed, new ncol: ", ncol(Xtri\_tr\_nz)))

## [1] "Xtrial\_tr ncol: 47 NZV removed, new ncol: 35"

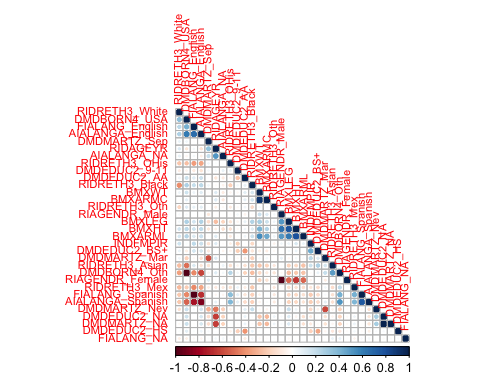
#looking for high corr:  
Xtritr\_hiC <- findCorrelation(cor(Xtri\_tr\_nz), cutoff = 0.8)  
Xtritr\_hiC

## [1] 33 18 3 30 8 9 25 10

# "AIALANGA\_English", "DMDBORN4\_USA", "BMXWT", "FIALANG\_English",   
 # "BMXWAIST", "BMXHIP", "DMDEDUC2\_NA", "RIAGENDR\_Male"  
  
Xtritr\_hiC <- subset(Xtri\_tr\_nz, select = c("AIALANGA\_English", "DMDBORN4\_USA",   
 "BMXWT", "FIALANG\_English", "BMXWAIST",  
 "BMXHIP", "DMDEDUC2\_NA", "RIAGENDR\_Male"))  
invisible(cor(Xtritr\_hiC)) # invisible used to reduce extensive output  
corrplot(cor(Xtritr\_hiC), order = "hclust", type="lower")



#dropping "BMXHIP", "BMXWAIST", DMDEDUC2\_<9 (recurring issues in model attempts)  
drophiC <- c("BMXHIP", "BMXWAIST", "DMDEDUC2\_<9")  
X\_trial\_train <- subset(Xtri\_tr\_nz, select = !(names(Xtri\_tr\_nz) %in% drophiC))  
corrplot(cor(X\_trial\_train), order = "hclust", type="lower", tl.cex = 0.7)

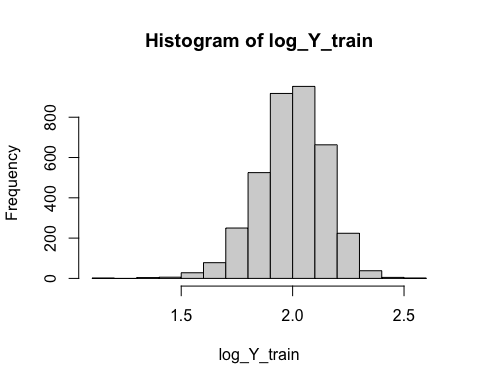


### Making test and train columns match

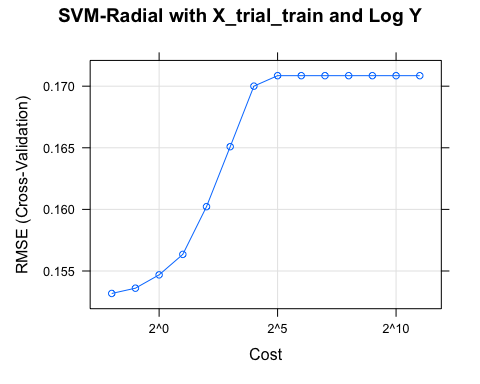
keepsies <- colnames(X\_trial\_train)  
X\_trial\_test <- subset(Chol\_test\_X\_imp, select = c(keepsies))

### Radial SVM with the matched data and log adjusted y

#svm radial with X\_trial\_train and log adjusted y  
log\_Y\_train <- log10(Chol\_train\_y)  
hist(log\_Y\_train)



log\_Y\_test <- log10(Chol\_test\_y)  
  
#model  
set.seed(123)  
svmR\_trial <- train(x=X\_trial\_train, y = log\_Y\_train,  
 method = "svmRadial",  
 preProcess = c("center", "scale"),  
 tuneLength = 14,  
 trControl = Chol\_control)  
invisible(svmR\_trial) # output hidden to reduce clutter. Key observations noted below:  
# issues in: DMDEDUC2\_<9,   
 # final model uses: sigma = 0.02019005 and C = 0.25.  
 # RMSE: 0.1531765, Rsquared: 0.021637157  
 # while these are the lowest values, the graph is identical to the non-log adjusted SVM radial model with just different RMSE values.   
plot(svmR\_trial, scales = list(x = list(log = 2)), main="SVM-Radial with X\_trial\_train and Log Y")



svmR\_trial$finalModel

## Support Vector Machine object of class "ksvm"   
##   
## SV type: eps-svr (regression)   
## parameter : epsilon = 0.1 cost C = 0.25   
##   
## Gaussian Radial Basis kernel function.   
## Hyperparameter : sigma = 0.0201900463201106   
##   
## Number of Support Vectors : 3367   
##   
## Objective Function Value : -574.0137   
## Training error : 0.850126

trialPred <- data.frame(obs=log\_Y\_test)  
trialPred$svmRad <- predict(svmR\_trial, X\_trial\_test)

# Regression Trees - Eva

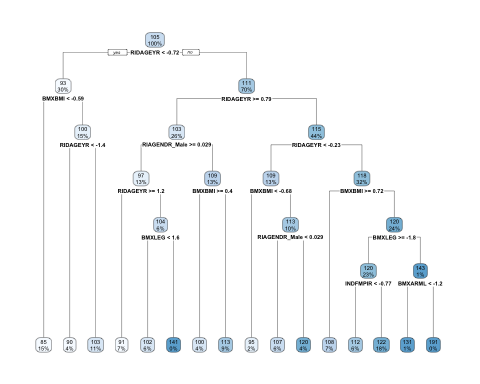
## Single Regression Trees

### Tree with CART based splits (rpart) and optimization with default parameters

set.seed(123)  
# let's see how it splits the training data  
Chol\_rpart <- rpart(Chol\_train\_y ~., data = Chol\_tr\_X\_imp\_fin, cp = 0.003)  
  
invisible(summary(Chol\_rpart)) # output hidden to reduce clutter. Key observations noted below:

## Call:  
## rpart(formula = Chol\_train\_y ~ ., data = Chol\_tr\_X\_imp\_fin, cp = 0.003)  
## n= 3696   
##   
## CP nsplit rel error xerror xstd  
## 1 0.054164414 0 1.0000000 1.0004973 0.03300310  
## 2 0.020563310 1 0.9458356 0.9554923 0.03188785  
## 3 0.013308632 2 0.9252723 0.9464258 0.03207678  
## 4 0.007112960 3 0.9119636 0.9311717 0.03205759  
## 5 0.005855210 4 0.9048507 0.9260955 0.03176146  
## 6 0.004972343 6 0.8931403 0.9256549 0.03150486  
## 7 0.004824224 7 0.8881679 0.9238887 0.03142018  
## 8 0.004142728 8 0.8833437 0.9264225 0.03141473  
## 9 0.003828087 10 0.8750582 0.9339156 0.03236363  
## 10 0.003774630 11 0.8712302 0.9329465 0.03238441  
## 11 0.003599490 12 0.8674555 0.9316994 0.03236427  
## 12 0.003384505 13 0.8638560 0.9351374 0.03241022  
## 13 0.003021559 14 0.8604715 0.9307203 0.03231526  
## 14 0.003000000 15 0.8574500 0.9348469 0.03254034  
##   
## Variable importance  
## RIDAGEYR BMXBMI DMDEDUC2\_NA DMDMARTZ\_Nev BMXARMC   
## 32 13 12 9 8   
## BMXARML AIALANGA\_NA BMXLEG BMXHT RIAGENDR\_Male   
## 5 5 4 4 4   
## INDFMPIR DMDMARTZ\_Sep DMDEDUC2\_BS+ DMDMARTZ\_Mar   
## 1 1 1 1   
##   
## Node number 1: 3696 observations, complexity param=0.05416441  
## mean=105.2411, MSE=1257.805   
## left son=2 (1098 obs) right son=3 (2598 obs)  
## Primary splits:  
## RIDAGEYR < -0.7177214 to the left, improve=0.05416441, (0 missing)  
## DMDEDUC2\_NA < 0.9305478 to the right, improve=0.04807959, (0 missing)  
## DMDMARTZ\_Nev < -0.2411516 to the right, improve=0.02845850, (0 missing)  
## BMXBMI < -0.7742706 to the left, improve=0.02378829, (0 missing)  
## BMXARMC < -0.8110234 to the left, improve=0.01974683, (0 missing)  
## Surrogate splits:  
## DMDEDUC2\_NA < 0.9305478 to the right, agree=0.862, adj=0.536, (0 split)  
## DMDMARTZ\_Nev < -0.2411516 to the right, agree=0.846, adj=0.482, (0 split)  
## BMXBMI < -0.9678924 to the left, agree=0.735, adj=0.108, (0 split)  
## BMXARMC < -1.151012 to the left, agree=0.730, adj=0.091, (0 split)  
##   
## Node number 2: 1098 observations, complexity param=0.01330863  
## mean=92.54463, MSE=805.3737   
## left son=4 (536 obs) right son=5 (562 obs)  
## Primary splits:  
## BMXBMI < -0.5935568 to the left, improve=0.06996470, (0 missing)  
## BMXARMC < -0.113152 to the left, improve=0.06346379, (0 missing)  
## RIDAGEYR < -1.398075 to the left, improve=0.04298442, (0 missing)  
## DMDEDUC2\_NA < 0.9305478 to the right, improve=0.03832009, (0 missing)  
## DMDEDUC2\_AA < -0.4815614 to the right, improve=0.02453409, (0 missing)  
## Surrogate splits:  
## BMXARMC < -0.4352465 to the left, agree=0.884, adj=0.763, (0 split)  
## BMXARML < -0.3506071 to the left, agree=0.622, adj=0.226, (0 split)  
## RIDAGEYR < -1.203688 to the left, agree=0.605, adj=0.190, (0 split)  
## DMDEDUC2\_NA < 0.9305478 to the right, agree=0.604, adj=0.188, (0 split)  
## DMDEDUC2\_BS+ < -0.3327601 to the right, agree=0.587, adj=0.153, (0 split)  
##   
## Node number 3: 2598 observations, complexity param=0.02056331  
## mean=110.607, MSE=1352.096   
## left son=6 (968 obs) right son=7 (1630 obs)  
## Primary splits:  
## RIDAGEYR < 0.7887756 to the right, improve=0.027213960, (0 missing)  
## AIALANGA\_NA < 0.7694033 to the right, improve=0.019766670, (0 missing)  
## BMXBMI < 0.6714394 to the right, improve=0.007005182, (0 missing)  
## DMDBORN4\_Oth < 0.5241397 to the left, improve=0.006384595, (0 missing)  
## BMXARML < 0.8611078 to the right, improve=0.005282776, (0 missing)  
## Surrogate splits:  
## AIALANGA\_NA < 0.7694033 to the right, agree=0.787, adj=0.428, (0 split)  
## DMDMARTZ\_Sep < 0.6618522 to the right, agree=0.653, adj=0.068, (0 split)  
## BMXLEG < -2.129812 to the left, agree=0.630, adj=0.007, (0 split)  
## BMXHT < -2.253535 to the left, agree=0.629, adj=0.005, (0 split)  
## BMXARML < -2.618245 to the left, agree=0.628, adj=0.001, (0 split)  
##   
## Node number 4: 536 observations  
## mean=84.85821, MSE=627.8344   
##   
## Node number 5: 562 observations, complexity param=0.003828087  
## mean=99.87544, MSE=864.6108   
## left son=10 (146 obs) right son=11 (416 obs)  
## Primary splits:  
## RIDAGEYR < -1.398075 to the left, improve=0.03662437, (0 missing)  
## BMXBMI < 0.32292 to the left, improve=0.02895393, (0 missing)  
## BMXARMC < 0.3163073 to the left, improve=0.02481897, (0 missing)  
## DMDEDUC2\_NA < 0.9305478 to the right, improve=0.02327611, (0 missing)  
## DMDEDUC2\_AA < -0.4815614 to the right, improve=0.01764382, (0 missing)  
## Surrogate splits:  
## DMDEDUC2\_NA < 0.9305478 to the right, agree=0.826, adj=0.329, (0 split)  
## INDFMPIR < -1.53041 to the left, agree=0.746, adj=0.021, (0 split)  
## BMXBMI < -0.5548324 to the left, agree=0.746, adj=0.021, (0 split)  
## BMXARMC < -1.043647 to the left, agree=0.746, adj=0.021, (0 split)  
## BMXHT < -2.298376 to the left, agree=0.744, adj=0.014, (0 split)  
##   
## Node number 6: 968 observations, complexity param=0.00711296  
## mean=102.7355, MSE=1416.277   
## left son=12 (495 obs) right son=13 (473 obs)  
## Primary splits:  
## RIAGENDR\_Male < 0.02868757 to the right, improve=0.02411971, (0 missing)  
## BMXHT < -0.4649097 to the right, improve=0.02055340, (0 missing)  
## BMXARML < 0.8611078 to the right, improve=0.01581797, (0 missing)  
## BMXBMI < 0.3706801 to the right, improve=0.01427778, (0 missing)  
## BMXARMC < -0.1310462 to the right, improve=0.01285898, (0 missing)  
## Surrogate splits:  
## BMXHT < -0.06633024 to the right, agree=0.853, adj=0.700, (0 split)  
## BMXLEG < -0.2429857 to the right, agree=0.773, adj=0.535, (0 split)  
## BMXARML < 0.2864087 to the right, agree=0.770, adj=0.529, (0 split)  
## DMDMARTZ\_Mar < -0.1600155 to the right, agree=0.616, adj=0.214, (0 split)  
## DMDMARTZ\_Sep < 0.6618522 to the left, agree=0.606, adj=0.195, (0 split)  
##   
## Node number 7: 1630 observations, complexity param=0.00585521  
## mean=115.2816, MSE=1255.334   
## left son=14 (464 obs) right son=15 (1166 obs)  
## Primary splits:  
## RIDAGEYR < -0.2317546 to the left, improve=0.011590290, (0 missing)  
## BMXBMI < 0.7617963 to the right, improve=0.011509240, (0 missing)  
## BMXLEG < -1.843139 to the right, improve=0.007419219, (0 missing)  
## DMDBORN4\_Oth < 0.5241397 to the left, improve=0.005611986, (0 missing)  
## BMXARMC < 1.085755 to the right, improve=0.004611308, (0 missing)  
## Surrogate splits:  
## BMXBMI < -1.613299 to the left, agree=0.718, adj=0.009, (0 split)  
## BMXARMC < -1.741519 to the left, agree=0.718, adj=0.009, (0 split)  
## BMXLEG < 2.27452 to the right, agree=0.717, adj=0.006, (0 split)  
## BMXARML < 2.817162 to the right, agree=0.717, adj=0.006, (0 split)  
##   
## Node number 10: 146 observations  
## mean=90.37671, MSE=687.0841   
##   
## Node number 11: 416 observations  
## mean=103.2091, MSE=884.1366   
##   
## Node number 12: 495 observations, complexity param=0.004824224  
## mean=97.02222, MSE=1260.83   
## left son=24 (257 obs) right son=25 (238 obs)  
## Primary splits:  
## RIDAGEYR < 1.177549 to the right, improve=0.03593447, (0 missing)  
## AIALANGA\_NA < 0.7694033 to the right, improve=0.03260307, (0 missing)  
## BMXLEG < 1.59693 to the left, improve=0.02390759, (0 missing)  
## BMXBMI < -0.2837618 to the right, improve=0.02065733, (0 missing)  
## BMXARMC < 1.318379 to the right, improve=0.01047926, (0 missing)  
## Surrogate splits:  
## AIALANGA\_NA < 0.7694033 to the right, agree=0.970, adj=0.937, (0 split)  
## RIDRETH3\_White < 0.3645385 to the right, agree=0.586, adj=0.139, (0 split)  
## BMXARMC < 0.2268366 to the left, agree=0.578, adj=0.122, (0 split)  
## RIDRETH3\_Black < 0.5542921 to the left, agree=0.570, adj=0.105, (0 split)  
## AIALANGA\_English < -2.469158 to the right, agree=0.570, adj=0.105, (0 split)  
##   
## Node number 13: 473 observations, complexity param=0.00377463  
## mean=108.7146, MSE=1509.045   
## left son=26 (156 obs) right son=27 (317 obs)  
## Primary splits:  
## BMXBMI < 0.3964963 to the right, improve=0.02458421, (0 missing)  
## INDFMPIR < -0.05039062 to the left, improve=0.02189095, (0 missing)  
## BMXARMC < 1.19312 to the right, improve=0.01510029, (0 missing)  
## BMXLEG < -2.520729 to the right, improve=0.01028420, (0 missing)  
## BMXARML < 0.6187648 to the right, improve=0.01009494, (0 missing)  
## Surrogate splits:  
## BMXARMC < 0.2304154 to the right, agree=0.854, adj=0.558, (0 split)  
## BMXARML < 0.7226261 to the right, agree=0.706, adj=0.109, (0 split)  
## INDFMPIR < -1.308407 to the left, agree=0.672, adj=0.006, (0 split)  
## BMXHT < 0.6610774 to the right, agree=0.672, adj=0.006, (0 split)  
##   
## Node number 14: 464 observations, complexity param=0.004972343  
## mean=109.2349, MSE=1026.456   
## left son=28 (91 obs) right son=29 (373 obs)  
## Primary splits:  
## BMXBMI < -0.6839137 to the left, improve=0.04853424, (0 missing)  
## RIAGENDR\_Male < 0.02868757 to the left, improve=0.03776510, (0 missing)  
## BMXARMC < -0.6499762 to the left, improve=0.03739263, (0 missing)  
## BMXLEG < 0.03326233 to the left, improve=0.01273680, (0 missing)  
## DMDBORN4\_Oth < 0.5241397 to the left, improve=0.01123883, (0 missing)  
## Surrogate splits:  
## BMXARMC < -0.8110234 to the left, agree=0.894, adj=0.462, (0 split)  
##   
## Node number 15: 1166 observations, complexity param=0.00585521  
## mean=117.6878, MSE=1326.074   
## left son=30 (265 obs) right son=31 (901 obs)  
## Primary splits:  
## BMXBMI < 0.7230719 to the right, improve=0.019870600, (0 missing)  
## RIDAGEYR < 0.6915823 to the right, improve=0.007605915, (0 missing)  
## BMXARMC < 1.085755 to the right, improve=0.006445164, (0 missing)  
## BMXLEG < -1.843139 to the right, improve=0.005717149, (0 missing)  
## BMXARML < 0.5841444 to the right, improve=0.004316487, (0 missing)  
## Surrogate splits:  
## BMXARMC < 1.085755 to the right, agree=0.886, adj=0.498, (0 split)  
## BMXLEG < -2.416484 to the left, agree=0.776, adj=0.015, (0 split)  
## BMXARML < 2.869092 to the right, agree=0.774, adj=0.004, (0 split)  
##   
## Node number 24: 257 observations  
## mean=90.54475, MSE=899.7889   
##   
## Node number 25: 238 observations, complexity param=0.003021559  
## mean=104.0168, MSE=1556.462   
## left son=50 (228 obs) right son=51 (10 obs)  
## Primary splits:  
## BMXLEG < 1.570869 to the left, improve=0.03791935, (0 missing)  
## DMDEDUC2\_9-11 < 1.224965 to the left, improve=0.03211565, (0 missing)  
## BMXBMI < -0.3483024 to the right, improve=0.02225825, (0 missing)  
## BMXHT < 1.478165 to the left, improve=0.02098695, (0 missing)  
## DMDMARTZ\_Nev < 0.7669317 to the right, improve=0.01875647, (0 missing)  
## Surrogate splits:  
## BMXHT < 2.235466 to the left, agree=0.966, adj=0.2, (0 split)  
##   
## Node number 26: 156 observations  
## mean=100.0321, MSE=1199.569   
##   
## Node number 27: 317 observations  
## mean=112.9874, MSE=1605.987   
##   
## Node number 28: 91 observations  
## mean=94.94505, MSE=776.6014   
##   
## Node number 29: 373 observations, complexity param=0.003384505  
## mean=112.7212, MSE=1025.44   
## left son=58 (215 obs) right son=59 (158 obs)  
## Primary splits:  
## RIAGENDR\_Male < 0.02868757 to the left, improve=0.04113595, (0 missing)  
## BMXLEG < 0.03326233 to the left, improve=0.01961291, (0 missing)  
## BMXBMI < 2.071971 to the right, improve=0.01668855, (0 missing)  
## DMDBORN4\_Oth < 0.5241397 to the left, improve=0.01517982, (0 missing)  
## BMXHT < -0.0165078 to the left, improve=0.01417552, (0 missing)  
## Surrogate splits:  
## BMXHT < 0.5016456 to the left, agree=0.831, adj=0.601, (0 split)  
## BMXARML < 0.2033197 to the left, agree=0.769, adj=0.456, (0 split)  
## BMXLEG < 0.2938737 to the left, agree=0.753, adj=0.418, (0 split)  
## INDFMPIR < -0.1527586 to the left, agree=0.619, adj=0.101, (0 split)  
## BMXARMC < -0.0952579 to the left, agree=0.590, adj=0.032, (0 split)  
##   
## Node number 30: 265 observations  
## mean=108.2226, MSE=1089.109   
##   
## Node number 31: 901 observations, complexity param=0.004142728  
## mean=120.4717, MSE=1361.67   
## left son=62 (867 obs) right son=63 (34 obs)  
## Primary splits:  
## BMXLEG < -1.843139 to the right, improve=0.014884930, (0 missing)  
## BMXARMC < 1.211014 to the left, improve=0.009461288, (0 missing)  
## RIDRETH3\_Black < 0.5542921 to the right, improve=0.008043446, (0 missing)  
## INDFMPIR < -0.7231829 to the left, improve=0.007061246, (0 missing)  
## BMXBMI < -0.2708537 to the left, improve=0.006776726, (0 missing)  
## Surrogate splits:  
## BMXHT < -1.954601 to the right, agree=0.971, adj=0.235, (0 split)  
## BMXARML < -2.323971 to the right, agree=0.966, adj=0.088, (0 split)  
##   
## Node number 50: 228 observations  
## mean=102.4079, MSE=1440.917   
##   
## Node number 51: 10 observations  
## mean=140.7, MSE=2786.21   
##   
## Node number 58: 215 observations  
## mean=107.1535, MSE=854.0927   
##   
## Node number 59: 158 observations  
## mean=120.2975, MSE=1159.019   
##   
## Node number 62: 867 observations, complexity param=0.00359949  
## mean=119.5802, MSE=1272.986   
## left son=124 (210 obs) right son=125 (657 obs)  
## Primary splits:  
## INDFMPIR < -0.7743669 to the left, improve=0.015161550, (0 missing)  
## BMXARMC < 1.211014 to the left, improve=0.011285510, (0 missing)  
## BMXBMI < -0.2708537 to the left, improve=0.008664953, (0 missing)  
## RIDAGEYR < 0.6915823 to the right, improve=0.006674608, (0 missing)  
## RIDRETH3\_Black < 0.5542921 to the right, improve=0.006344337, (0 missing)  
## Surrogate splits:  
## BMXARMC < -2.036772 to the left, agree=0.760, adj=0.010, (0 split)  
## BMXBMI < -1.548758 to the left, agree=0.759, adj=0.005, (0 split)  
##   
## Node number 63: 34 observations, complexity param=0.004142728  
## mean=143.2059, MSE=3085.987   
## left son=126 (27 obs) right son=127 (7 obs)  
## Primary splits:  
## BMXARML < -1.164187 to the left, improve=0.19305520, (0 missing)  
## BMXHT < -1.112601 to the left, improve=0.16323750, (0 missing)  
## INDFMPIR < -0.8033506 to the right, improve=0.11564990, (0 missing)  
## RIDAGEYR < 0.1084221 to the left, improve=0.07737871, (0 missing)  
## DMDEDUC2\_AA < 0.3718316 to the right, improve=0.07447832, (0 missing)  
## Surrogate splits:  
## BMXHT < -1.022921 to the left, agree=0.912, adj=0.571, (0 split)  
## BMXBMI < 0.4971797 to the left, agree=0.824, adj=0.143, (0 split)  
##   
## Node number 124: 210 observations  
## mean=111.8095, MSE=1186.621   
##   
## Node number 125: 657 observations  
## mean=122.0639, MSE=1275.122   
##   
## Node number 126: 27 observations  
## mean=130.7778, MSE=1924.099   
##   
## Node number 127: 7 observations  
## mean=191.1429, MSE=4673.837

# 127 nodes. means and MSE of the terminal nodes vary widely  
rpart.plot(Chol\_rpart)



# tune and predict  
Chol\_rpart\_tune <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "rpart", cp = 0.003)  
Chol\_rpart\_pred <- predict(Chol\_rpart\_tune, Chol\_test\_X\_imp)  
postResample(pred = Chol\_rpart\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 33.0113573 0.0659519 25.8550917

# Rsquared value of 0.066 isn't too great. RMSE of 33.0. Let's compare to other trees

### Tree with CART based splits (rpart2 to tune over max depth)

set.seed(123)  
Chol\_rpart2\_tune <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "rpart2", maxdepth = 6)  
Chol\_rpart2\_pred <- predict(Chol\_rpart2\_tune, Chol\_test\_X\_imp)  
postResample(pred = Chol\_rpart2\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 32.93397878 0.07117066 25.65328561

# minor improvement? Rsquared value of 0.071 and RMSE of 32.9

## Bagged Trees

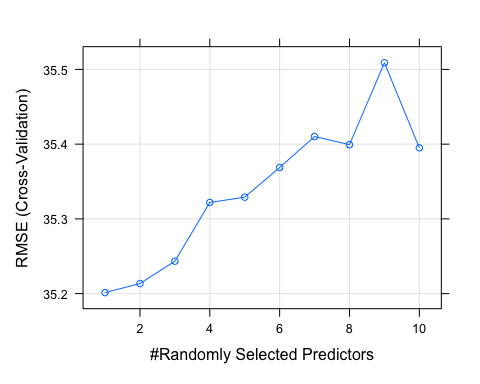
set.seed(123)  
Chol\_bagtree <- train(x=Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "treebag", nbagg = 70, cp = 0.003, trControl = Chol\_control)  
Chol\_bagtree

## Bagged CART   
##   
## 3696 samples  
## 32 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 51, 50, 48, 50, 46, 46, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 35.31841 0.03281359 27.35183

# Rsquared value of 0.033. RMSE is 35.3. Still not fantastic

## Random Forest

set.seed(123)  
  
rfmtryValues <- seq(1,10,1)  
  
Chol\_rf <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "rf", ntree = 300, tuneGrid = data.frame(mtry=rfmtryValues), trControl=Chol\_control) # 300 trees provides the highest Rsquared value when checking with test data  
invisible(Chol\_rf) # output hidden to reduce clutter  
plot(Chol\_rf)



Chol\_rf\_pred <- predict(Chol\_rf, Chol\_test\_X\_imp)  
postResample(pred = Chol\_rf\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 33.06977523 0.07717907 26.17119444

# Rsquared value of 0.077. RMSE of 33.1

## Boosted Trees

# some control parameters  
gbmGrid <- expand.grid(interaction.depth = c(1,3,5,7,9), n.trees=300, shrinkage = c(0.01, 0.1), n.minobsinnode=5)  
  
set.seed(123)  
Chol\_gbm <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "gbm", tuneGrid = gbmGrid, verbose = FALSE, trControl = Chol\_control)  
invisible(Chol\_gbm) # output hidden to reduce clutter. Key observations noted below:  
# shrinkage of 0.01 and a smaller interaction depth provided models with the lowest RMSE values  
  
Chol\_gbm\_pred <- predict(Chol\_gbm, Chol\_test\_X\_imp)  
postResample(pred = Chol\_gbm\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 32.86446826 0.08137564 25.82653202

# RMSE 32.9 and Rsquared 0.081

## Model Trees

### Model Trees (M5)

# decision tree with linear regression at terminal nodes to predict continuous variables  
set.seed(123)  
Chol\_M5 <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "M5", trControl = Chol\_control, control = Weka\_control(M=10))  
  
plot(Chol\_M5)



Chol\_M5\_pred <- predict(Chol\_M5, Chol\_test\_X\_imp)  
postResample(pred = Chol\_M5\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 33.61012224 0.07641278 26.16774336

# Rsquared value of 0.076 and RMSE of 33.6

### Model Tree (Rule Based M5)

# decision tree with linear regression at terminal nodes to predict continuous variables  
set.seed(123)  
Chol\_M5rules <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "M5Rules", trControl = Chol\_control, control = Weka\_control(M=10))  
  
plot(Chol\_M5rules)



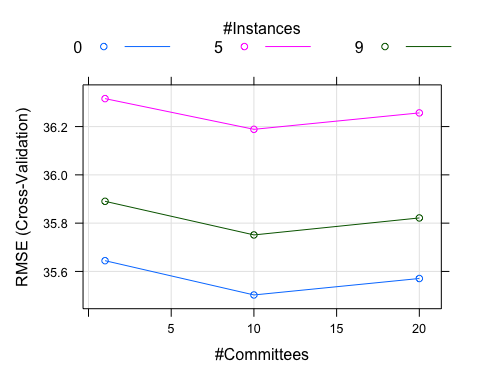
Chol\_M5rules\_pred <- predict(Chol\_M5rules, Chol\_test\_X\_imp)  
postResample(pred = Chol\_M5rules\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 32.75281564 0.08668215 25.26610200

# Slight improvement, Rsquared value of 0.087 and RMSE of 32.8

## Cubist

set.seed(123)  
Chol\_cube <- train(x = Chol\_tr\_X\_imp\_fin, y = Chol\_train\_y, method = "cubist", trControl = Chol\_control)  
  
plot(Chol\_cube)



Chol\_cube\_pred <- predict(Chol\_cube, Chol\_test\_X\_imp)  
postResample(pred = Chol\_cube\_pred, obs = Chol\_test\_y)

## RMSE Rsquared MAE   
## 32.63815970 0.09218122 25.11596950

# "Best performing" so far, but ever so slightly over the other models. Rsquared of 0.092 and RMSE of 32.6

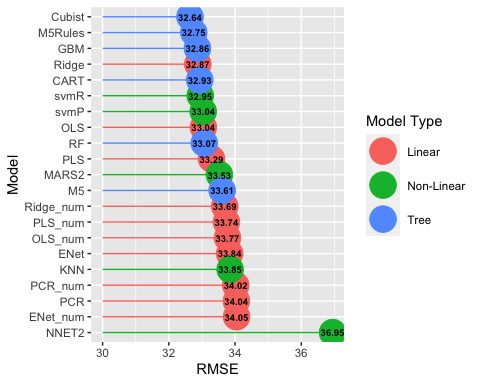
# Results

### Create Data set with all of the models

NonLpred <- NonLpred %>% select(!obs)  
Trees <- cbind.data.frame(Cubist = Chol\_cube\_pred, GBM = Chol\_gbm\_pred, M5 = Chol\_M5\_pred, M5Rules = Chol\_M5rules\_pred, RF = Chol\_rf\_pred, CART = Chol\_rpart2\_pred)  
  
Results <- cbind(Linear\_res, NonLpred, Trees)

### Get RMSE and Plot

find\_rmse <- function(x){  
 caret::RMSE(x, Results[,"Observed"])  
}  
  
RMSE\_results <- apply(X = Results[,2:22], FUN = find\_rmse, MARGIN = 2)  
RMSE\_results <- data.frame(RMSE\_results)  
RMSE\_results$Model <- rownames(RMSE\_results)  
RMSE\_results$Model\_Type <- "Linear"  
RMSE\_results$Model\_Type[11:15] <- "Non-Linear"  
RMSE\_results$Model\_Type[16:21] <- "Tree"  
  
ggplot(RMSE\_results, aes(x=reorder(Model, -RMSE\_results), y=RMSE\_results)) + geom\_segment(aes(x=reorder(Model, -RMSE\_results), xend = reorder(Model, -RMSE\_results), y=30, yend=RMSE\_results, color = Model\_Type)) + geom\_point(aes(color=Model\_Type), size = 9) + coord\_flip() + ylab("RMSE") + geom\_text(aes(label = round(RMSE\_results, 2)), color = "black", size = 2.5, fontface = "bold") + labs(color = "Model Type") + xlab("Model")



### Plot best model (cubist) against the original data

cubist\_plot <- Results %>% select(Observed, Cubist)  
cubist\_plot$x <- c(1:921)  
cubist\_plot <- cubist\_plot %>% gather(key = "Data Source", value = "LDL", -x)  
# cubist\_plot$alpha <- ifelse(cubist\_plot$Observed == "Observed", 0.8, 1)  
ggplot(cubist\_plot, aes(x = x, y = LDL)) + geom\_line(aes(color = `Data Source`, alpha = `Data Source`)) + scale\_color\_manual(values = c("royalblue1", "royalblue4")) + scale\_alpha\_manual(values = c(1,.3)) + theme\_classic()

