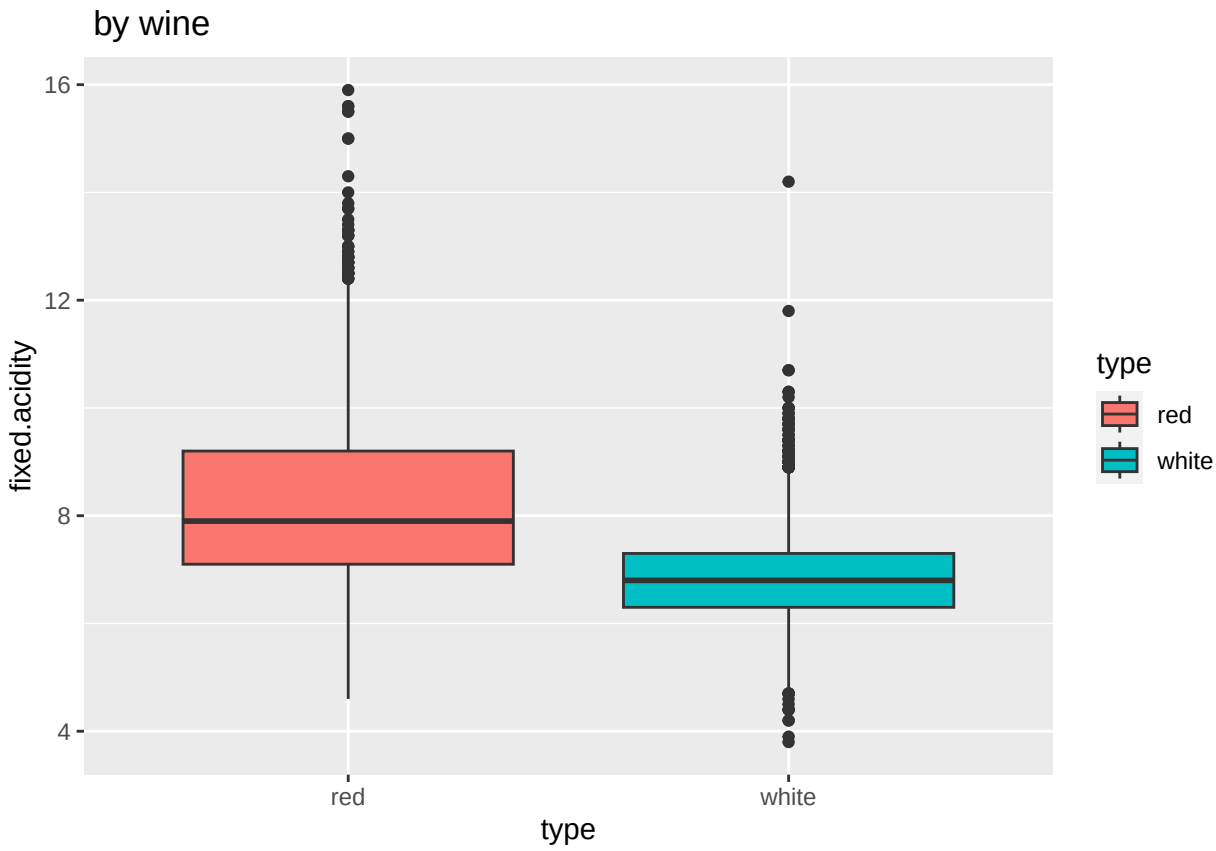
Final Project (STSCI 4740)

2023-04-24

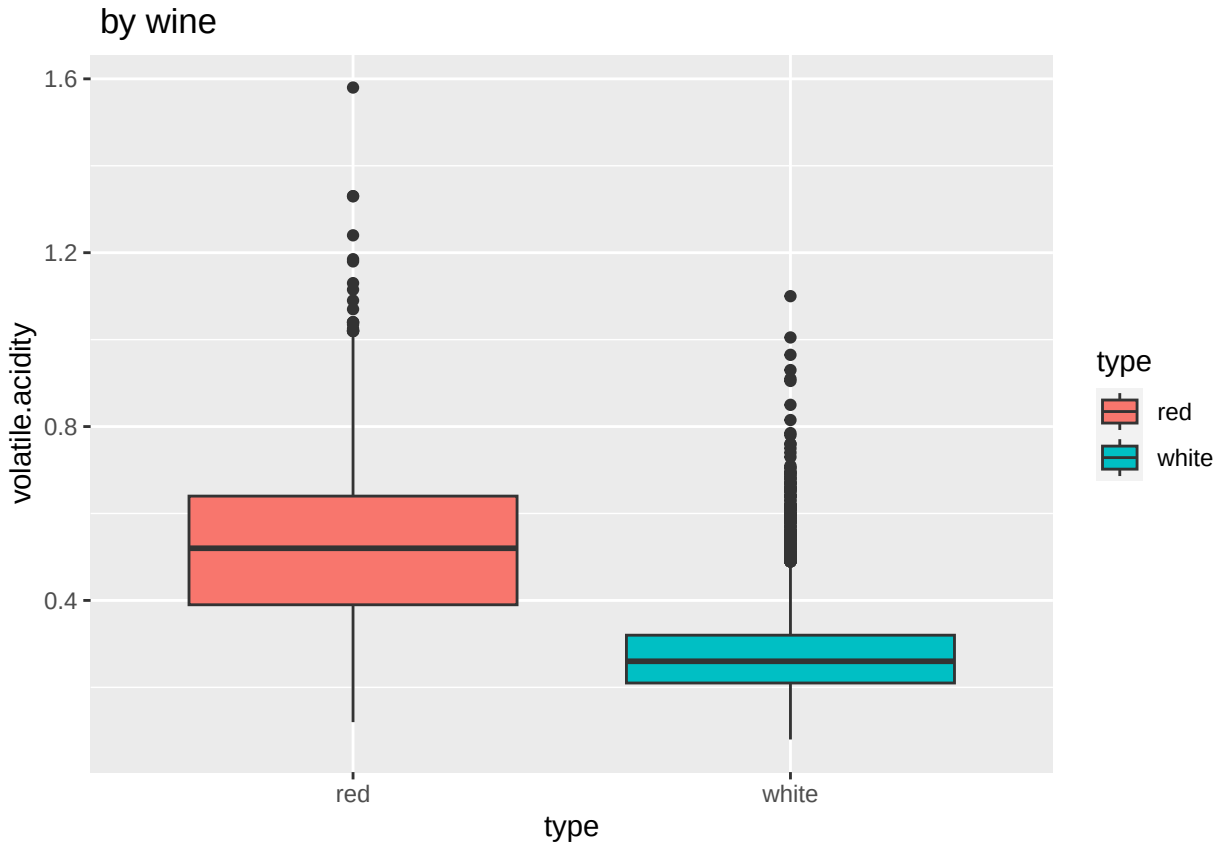
Part 1: Data Cleaning

EDA after for all the side by side boxplot

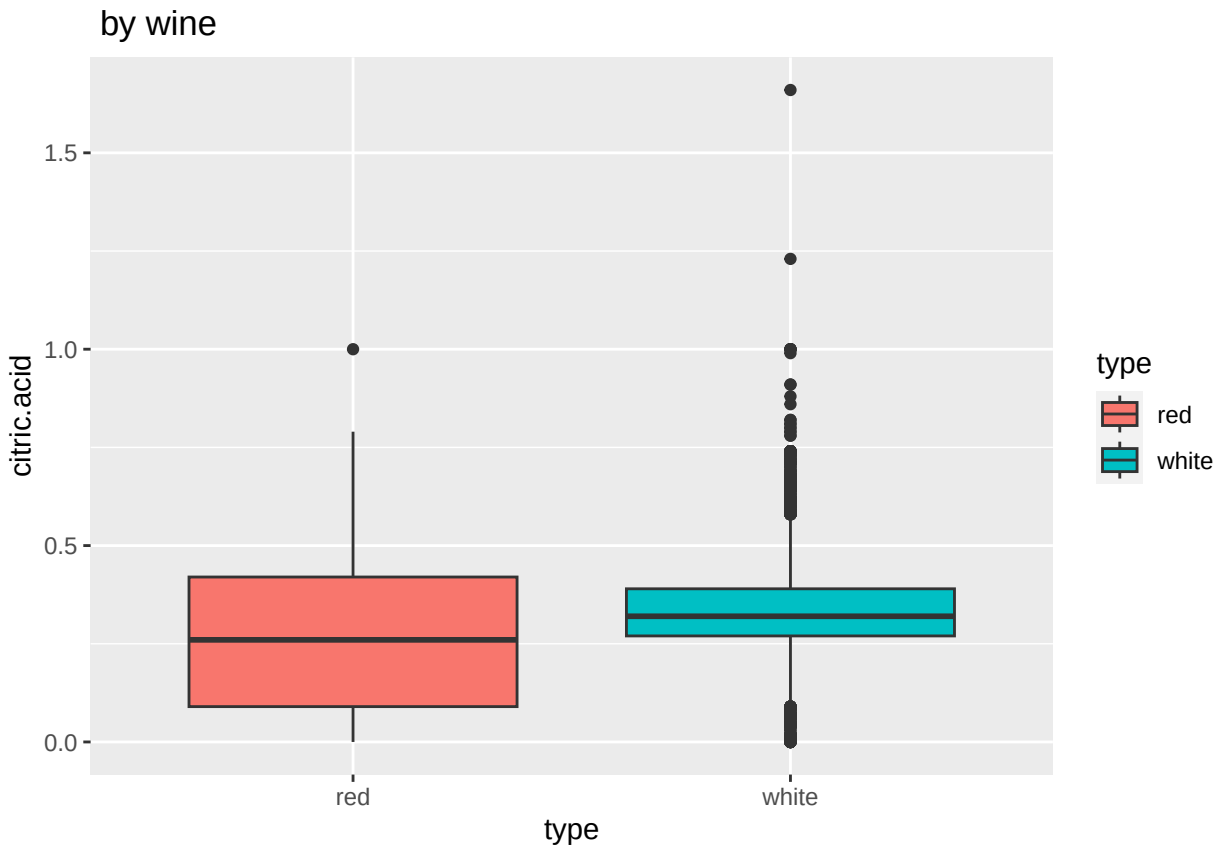
```
library(ggplot2)
par(mfrow=c(1,2))
#create vertical side-by-side boxplots for red and white wine
ggplot(wine, aes(x= type, y=fixed.acidity, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



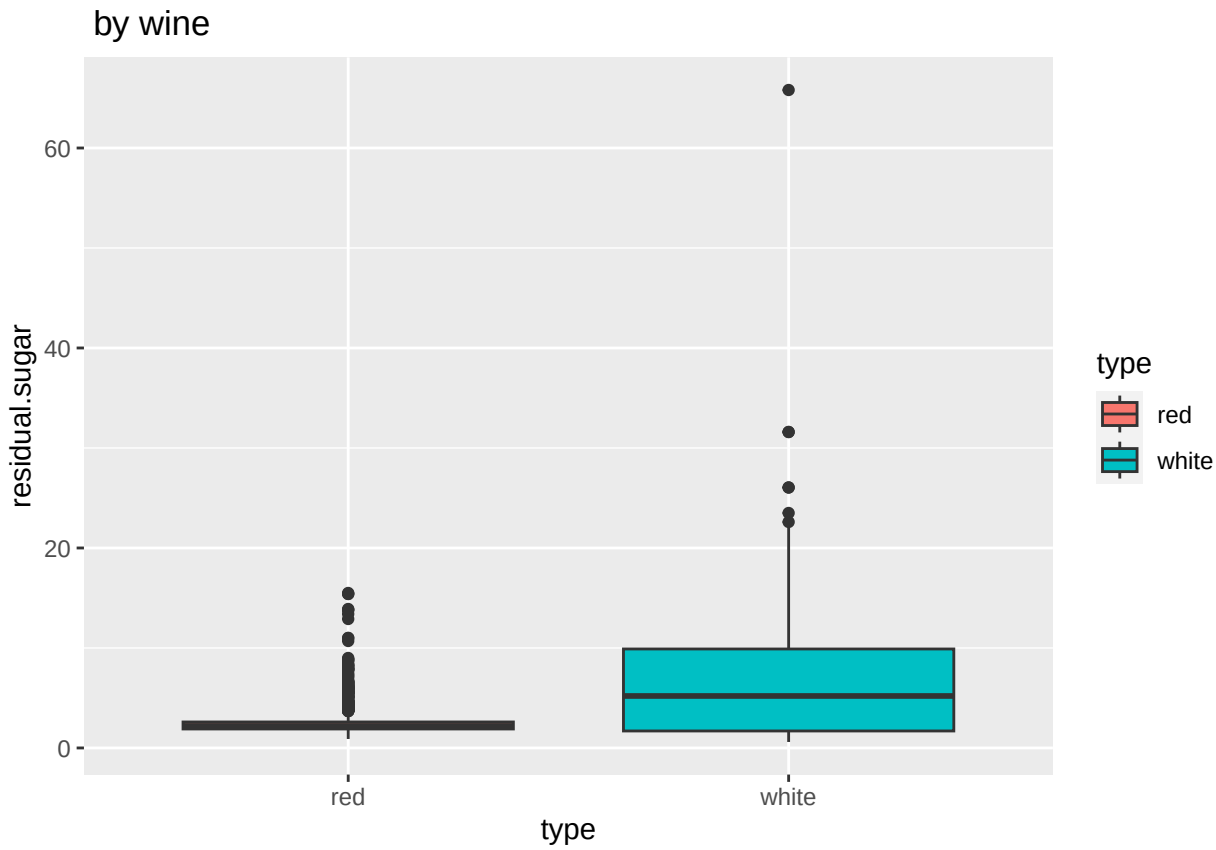
```
ggplot(wine, aes(x= type, y=volatile.acidity, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



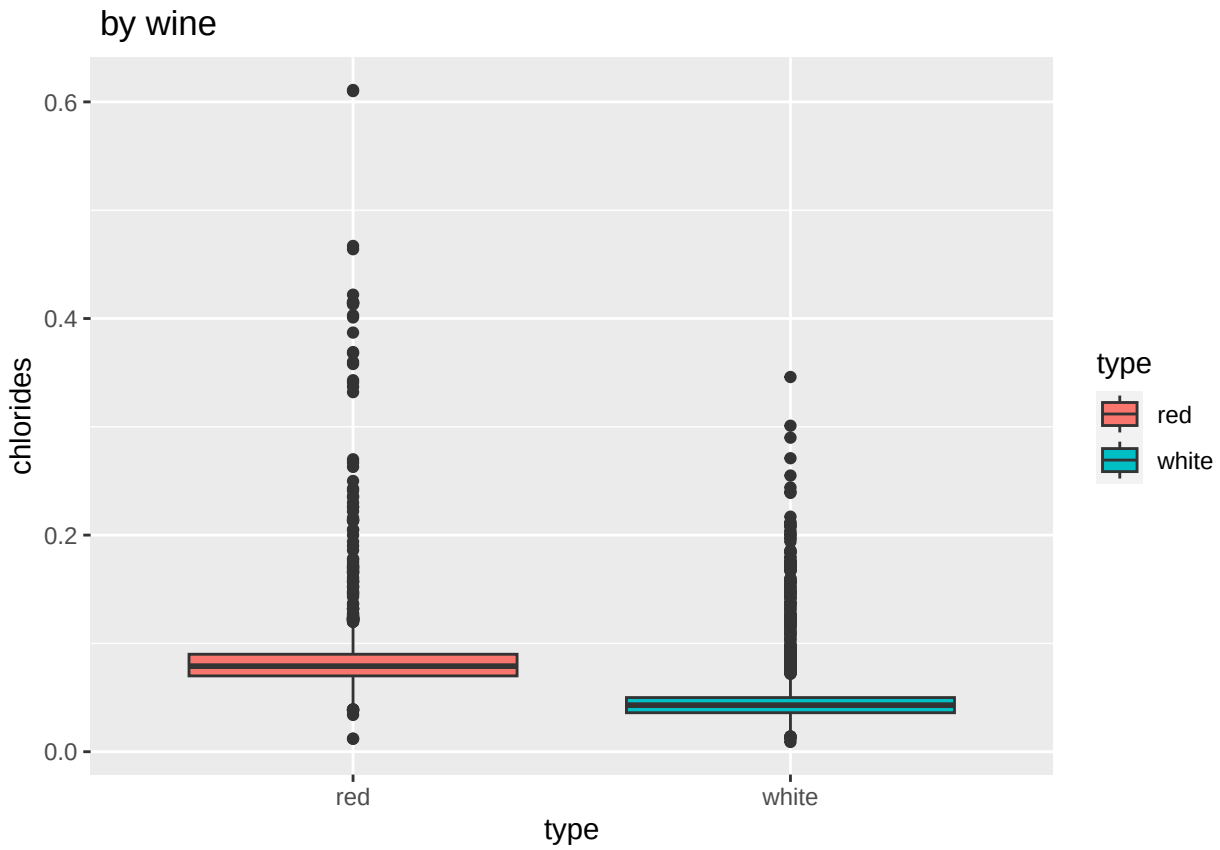

```
ggplot(wine, aes(x= type, y=citric.acid, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



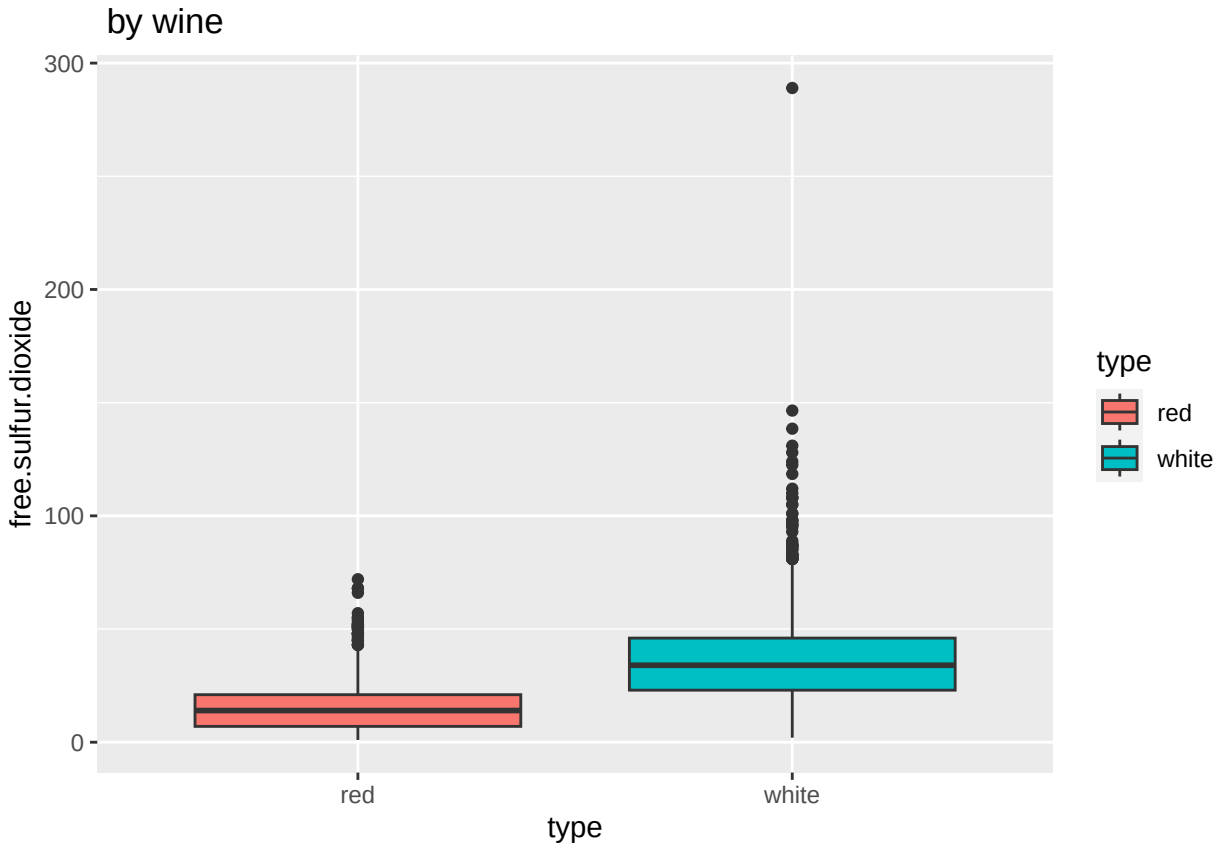

```
ggplot(wine, aes(x= type, y=residual.sugar, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



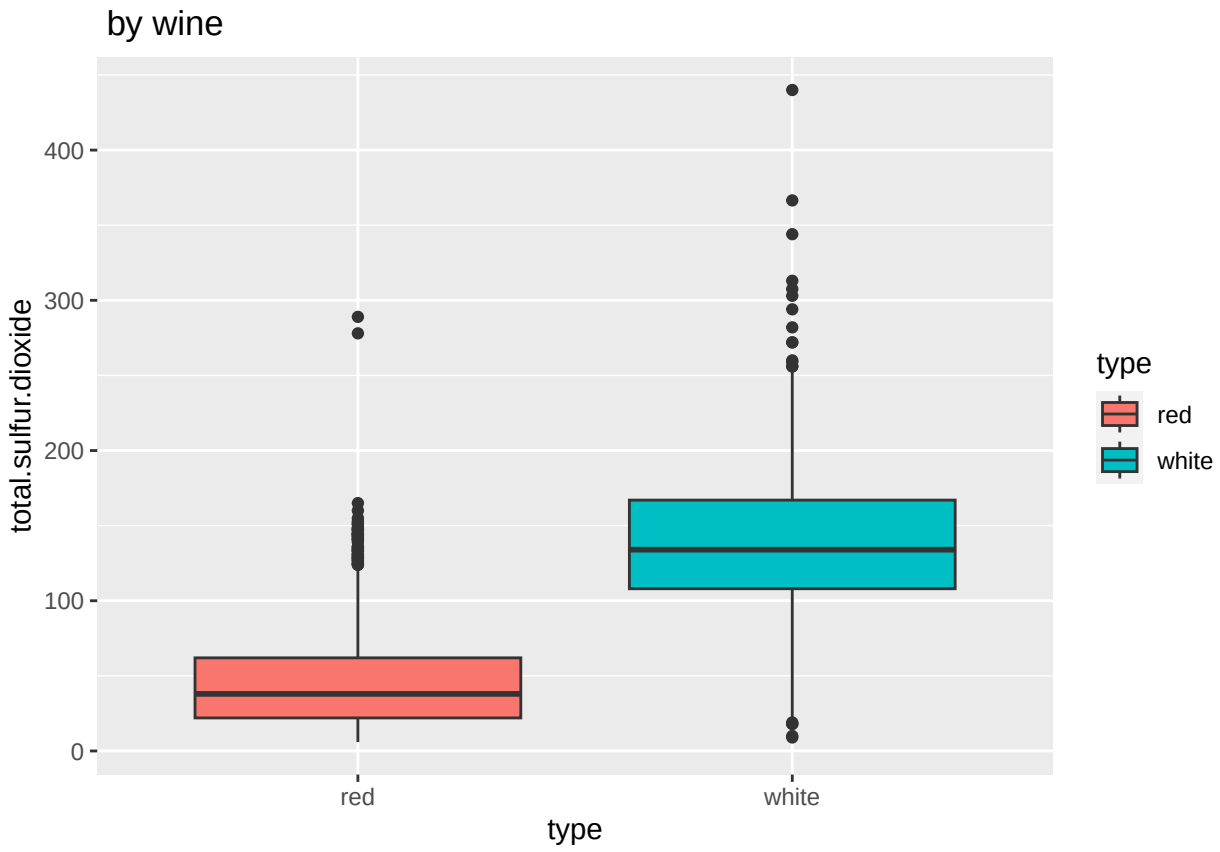

```
ggplot(wine, aes(x= type, y=chlorides, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



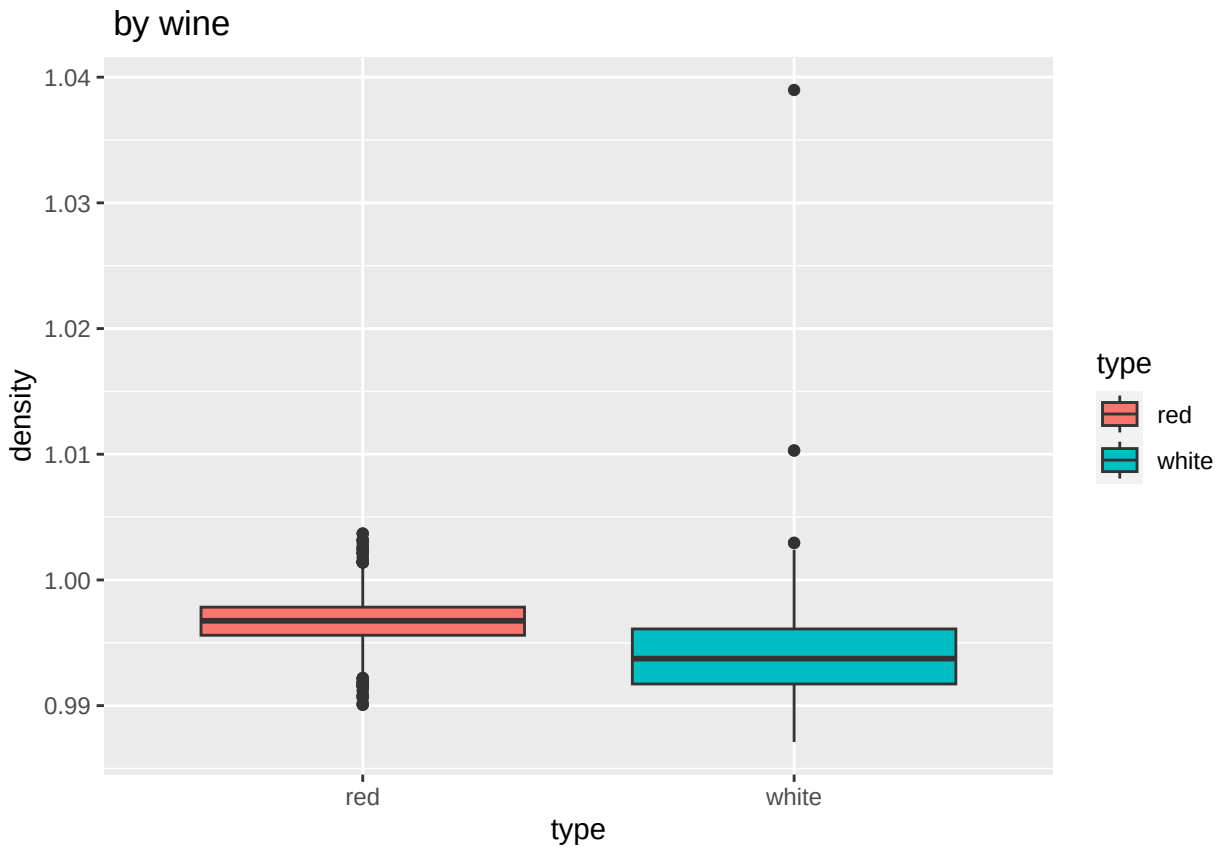

```
ggplot(wine, aes(x= type, y=free.sulfur.dioxide, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



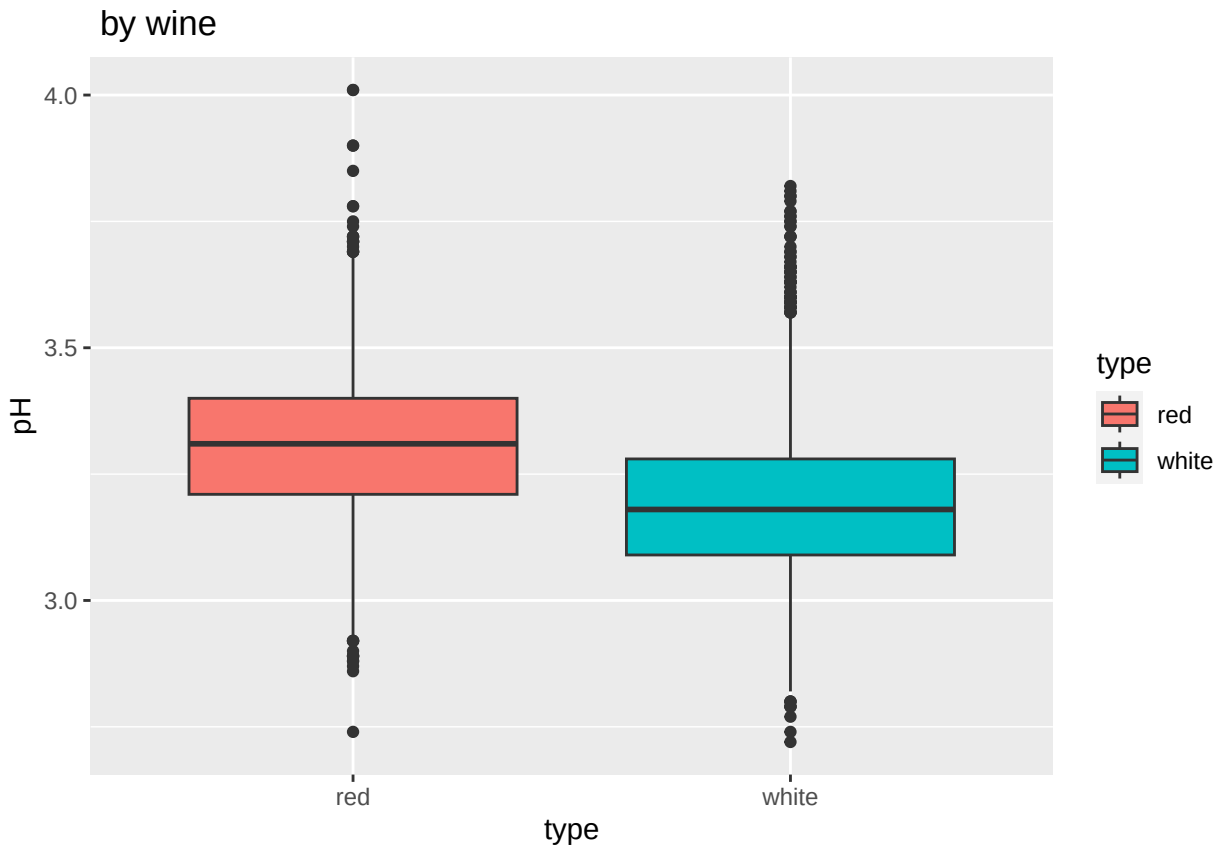

```
ggplot(wine, aes(x= type, y=total.sulfur.dioxide, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



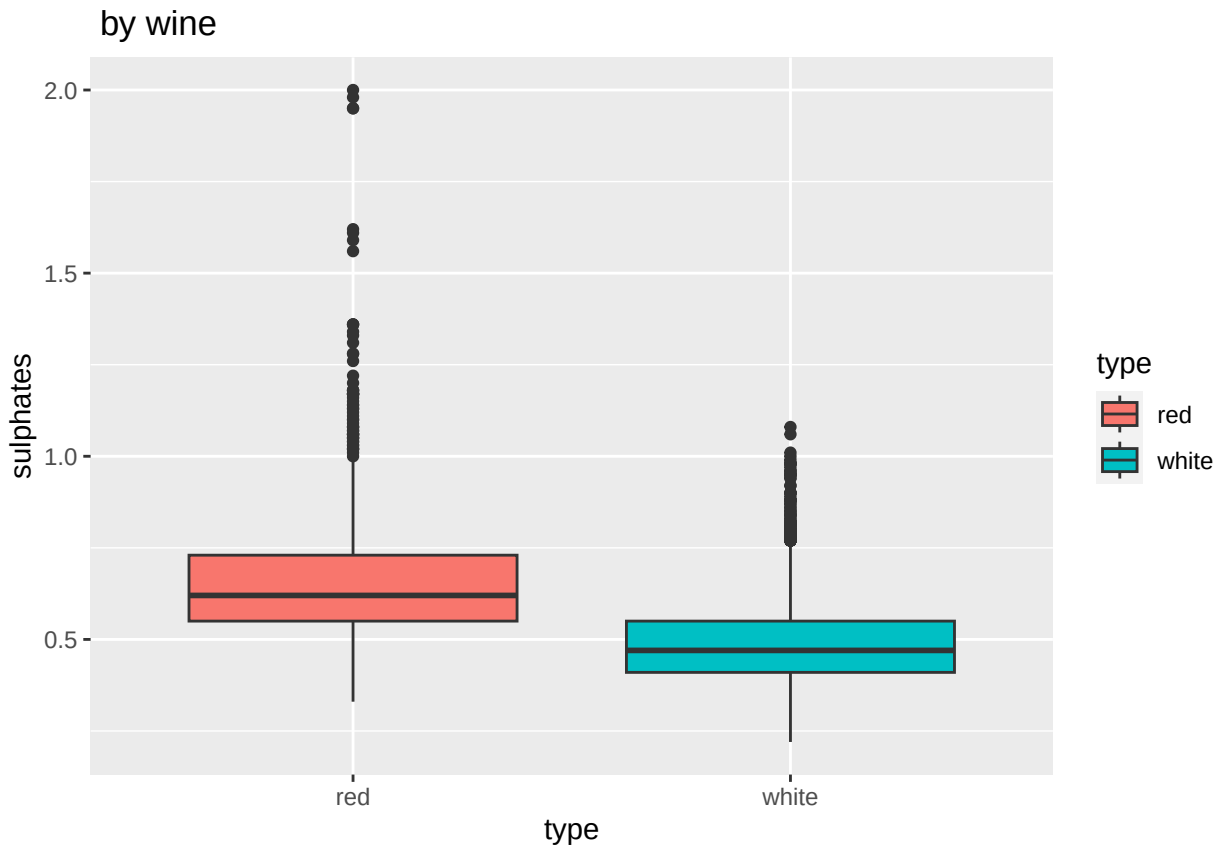

```
ggplot(wine, aes(x= type, y=density, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



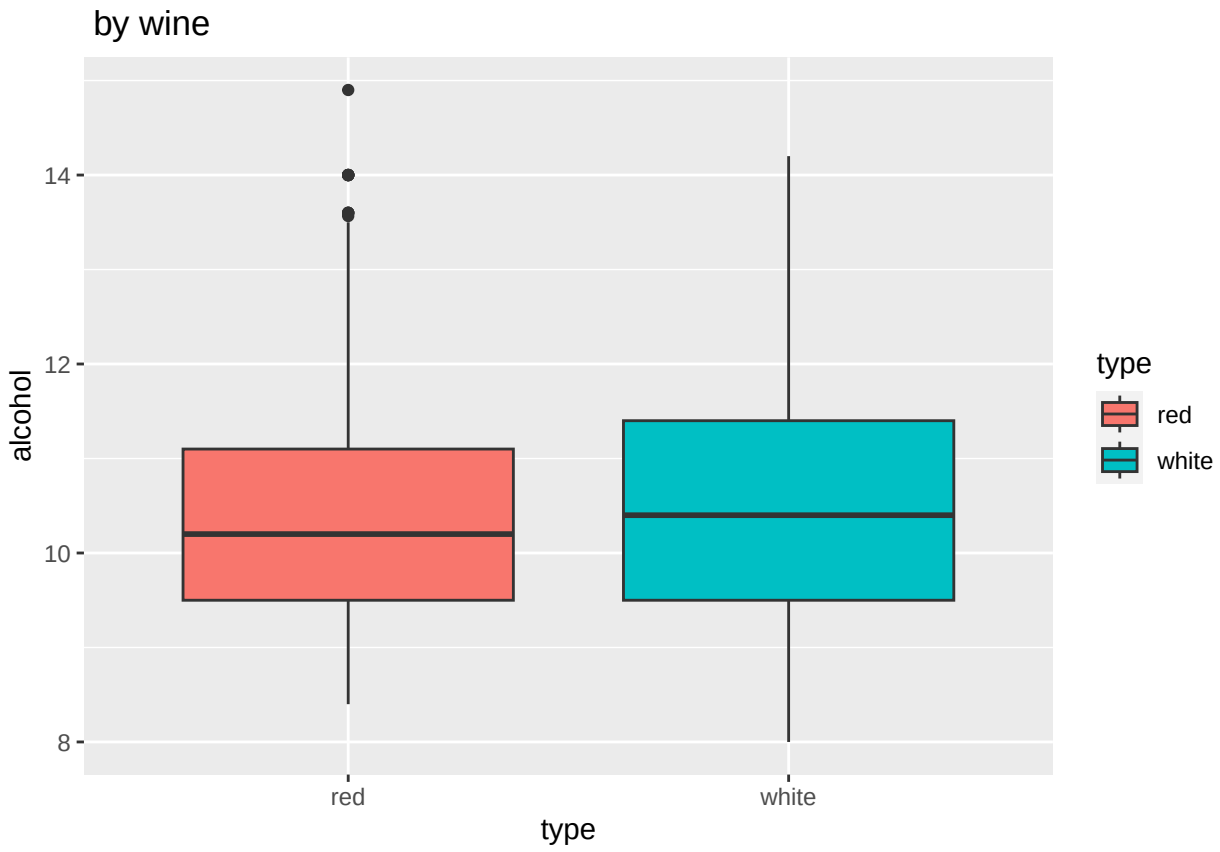

```
ggplot(wine, aes(x= type, y=pH, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```




```
ggplot(wine, aes(x= type, y=sulphates, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```




```
ggplot(wine, aes(x= type, y=alcohol, fill=type)) +
  geom_boxplot() +
  ggtitle(paste(colnames(data)[2], "by wine"))
```



Remove the extreme outliers

```
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v purrr   1.0.0      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
# density, residual sugar, free sulfur dioxide, and total sulfur dioxide

outlier_remove <- function(data, col) {
  Q1 <- quantile(data[,col], 0.25)
  Q3 <- quantile(data[,col], 0.75)
  IQR <- Q3 - Q1
  multiplier <- 3
  lower_bound <- Q1 - multiplier * IQR
  upper_bound <- Q3 + multiplier * IQR

  wine_outlier <- wine[data[,col] >= lower_bound & data[,col] <= upper_bound, ]
```

```
  return(wine_outlier)
}

df <- outlier_remove(wine,"density")
df2 <- outlier_remove(df,"residual.sugar")
df3 <- outlier_remove(df2,"free.sulfur.dioxide")
wine <- df3
nrow(wine)
```

```
## [1] 6489
```

```
wine <- wine[wine$quality != 3 & wine$quality != 9,]
```

```
whitewine <- wine[wine$type=="white",]
redwine <- wine[wine$type=="red",]
```

Standardized the data

```
standardize.predictor <- function( dat ){
  standardized.dat <- data.frame(tmp = rep(NA, times=nrow(dat)))
  for( i in 2:12 ){
    square <- ( dat[,i] - mean(dat[,i]) )**2
    den <- sqrt( (1/length(dat[,i])) * sum(square) )
    standardized.dat[,i] <- dat[,i] / den
  }
  standardized.dat <- select(standardized.dat, -tmp)
  colnames(standardized.dat) <- colnames(dat)[2:12]
  standardized.dat <- standardized.dat %>%
    mutate(type=dat[,1], quality=dat[,13])
  return(standardized.dat)
}

redwine.standard <- standardize.predictor(redwine)
whitewine.standard <- standardize.predictor(whitewine)
wine.standard <- rbind(redwine.standard,whitewine.standard)

# save the standardize data for further model building
write.csv(wine.standard,"wine.standard.csv", row.names=FALSE)
write.csv(redwine.standard, "redwine.standard.csv", row.names=FALSE)
write.csv(whitewine.standard, "whitewine.standard.csv", row.names=FALSE)
```

Read the dataset that is saved

```
redwine <- read.csv("redwine.standard.csv")
whitewine <- read.csv("whitewine.standard.csv")
```

```
table(wine$quality)
```

```
##
##    4    5    6    7    8
##  215 2134 2834 1078  193
```

Final Project (STSCI 4740)

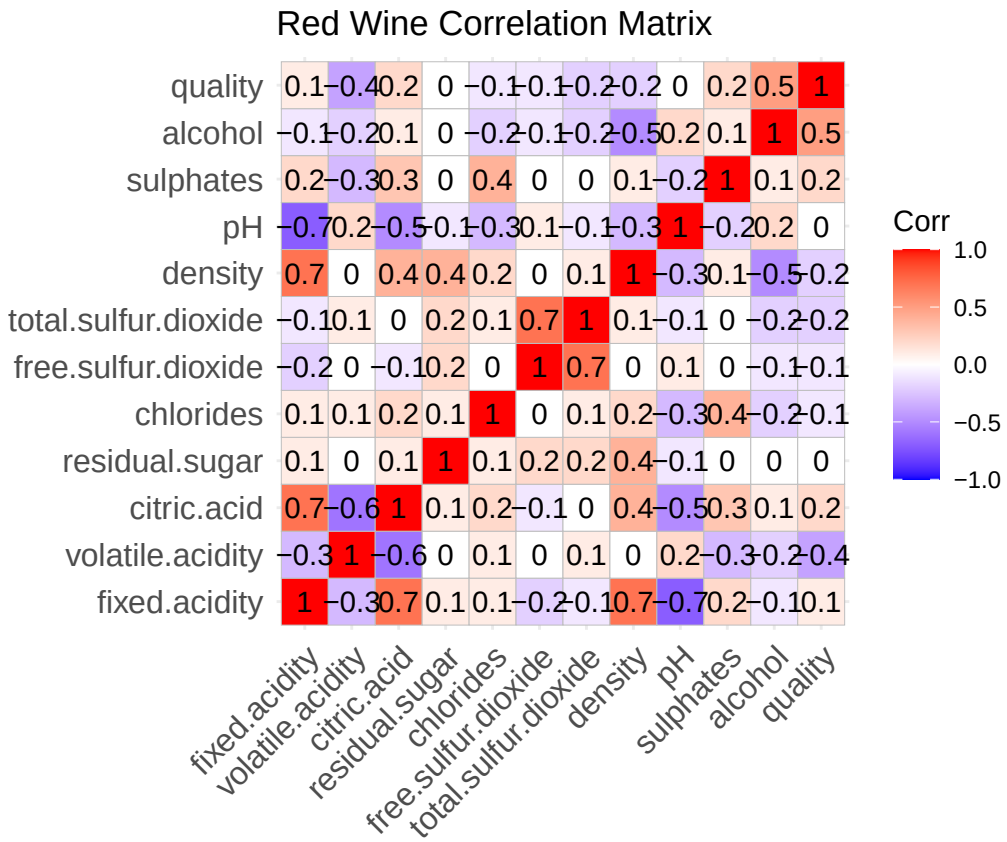
Nicholas Paschall

2023-04-24

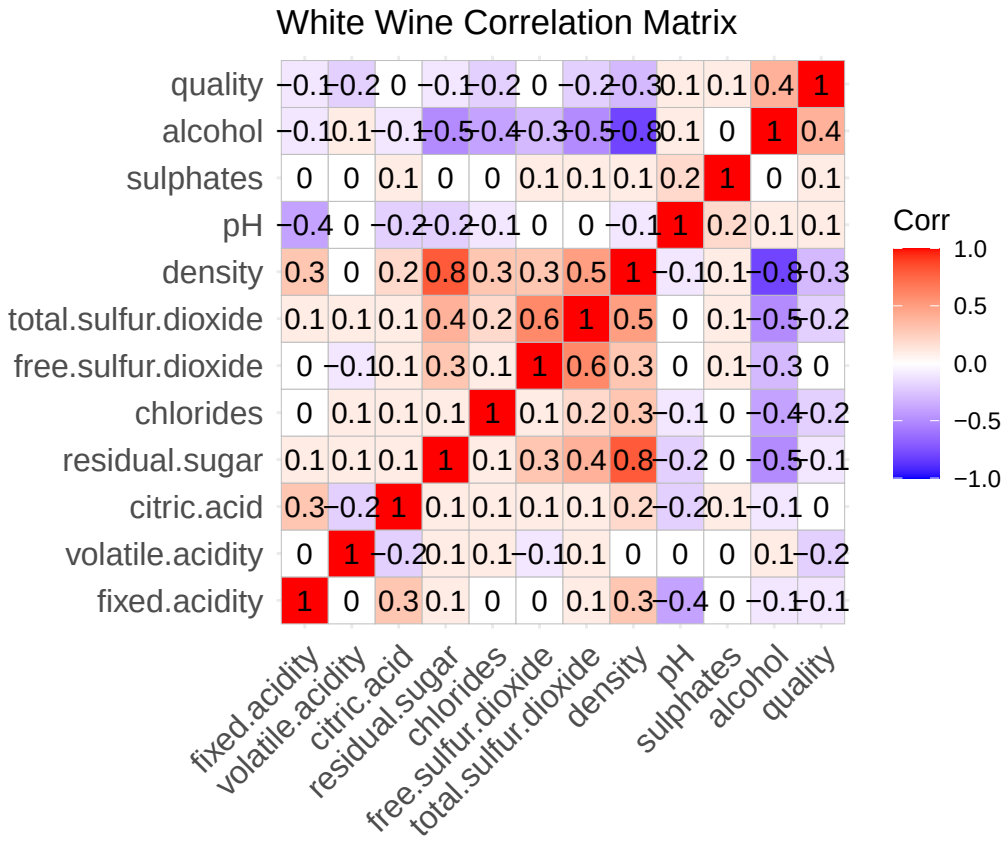
Part 2: Feature selection

CORRELATION MATRIX

```
# CORRELATION MATRICES FOR REDWINE AND WHITEWINE
redwine.mat <- as.matrix(redwine[c(1:11,13)])
redwine.corr <- round(cor(redwine.mat),1)
ggcorrplot::ggcorrplot(redwine.corr, lab=TRUE) +
  ggplot2::ggtitle("Red Wine Correlation Matrix")
```



```
whitewine.mat <- as.matrix(whitewine[c(1:11,13)])
whitewine.corr <- round(cor(whitewine.mat),1)
ggcorrplot::ggcorrplot(whitewine.corr, lab=TRUE) +
  ggplot2::ggtitle("White Wine Correlation Matrix")
```



Final Project (STSCI 4740)

Nicholas Paschall

2023-04-24

Part 3: Machine Learning Methods

Multiple Linear Regression

```
#Red Wine
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
# Set the seed for reproducibility
set.seed(123)

# Define the control object for the cross-validation
control <- trainControl(method = "cv",
                        number = 5, # 5-fold cross-validation
                        savePredictions = "final",
                        classProbs = FALSE,
                        summaryFunction = defaultSummary)

# Train the multiple linear regression model with cross-validation
lm_cv_model <- train(quality ~  density + volatile.acidity
                          + total.sulfur.dioxide + sulphates + alcohol,
                     data = redwine,
                     method = "lm",
                     trControl = control)

# Print the model details and performance metrics
print(lm_cv_model)
```

```
## Linear Regression
##
## 1589 samples
##    5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
```

```
## Summary of sample sizes: 1271, 1272, 1270, 1271, 1272
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.6366842  0.3416918  0.5006134
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
# Extract the cross-validated predictions from the model
cv_predictions <- lm_cv_model$pred

# Calculate the difference between the observed and predicted values
squared_errors <- (cv_predictions$obs - cv_predictions$pred)^2

# Calculate the mean of the squared errors
mse <- mean(squared_errors)

# Print the Mean Squared Error (MSE)
print(paste("Mean Squared Error (MSE):", mse))
```

```
## [1] "Mean Squared Error (MSE): 0.405736702877539"
```

```
#White Wine
library(caret)

# Set the seed for reproducibility
set.seed(123)

# Define the control object for the cross-validation
control <- trainControl(method = "cv",
                        number = 5, # 5-fold cross-validation
                        savePredictions = "final",
                        classProbs = FALSE,
                        summaryFunction = defaultSummary)

# Train the multiple linear regression model with cross-validation
lm_cv_model <- train(quality ~  pH + volatile.acidity
                          + residual.sugar + free.sulfur.dioxide
                          + alcohol,
                     data = whitewine,
                      method = "lm",
                      trControl = control)

# Print the model details and performance metrics
print(lm_cv_model)
```

```
## Linear Regression
##
## 4865 samples
##    5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
```

```
## Summary of sample sizes: 3892, 3892, 3891, 3892, 3893
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.7325629  0.2785544  0.5790683
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
# Extract the cross-validated predictions from the model
cv_predictions <- lm_cv_model$pred

# Calculate the difference between the observed and predicted values
squared_errors <- (cv_predictions$obs - cv_predictions$pred)^2

# Calculate the mean of the squared errors
mse <- mean(squared_errors)

# Print the Mean Squared Error (MSE)
print(paste("Mean Squared Error (MSE):", mse))
```

```
## [1] "Mean Squared Error (MSE): 0.5367999374442"
```

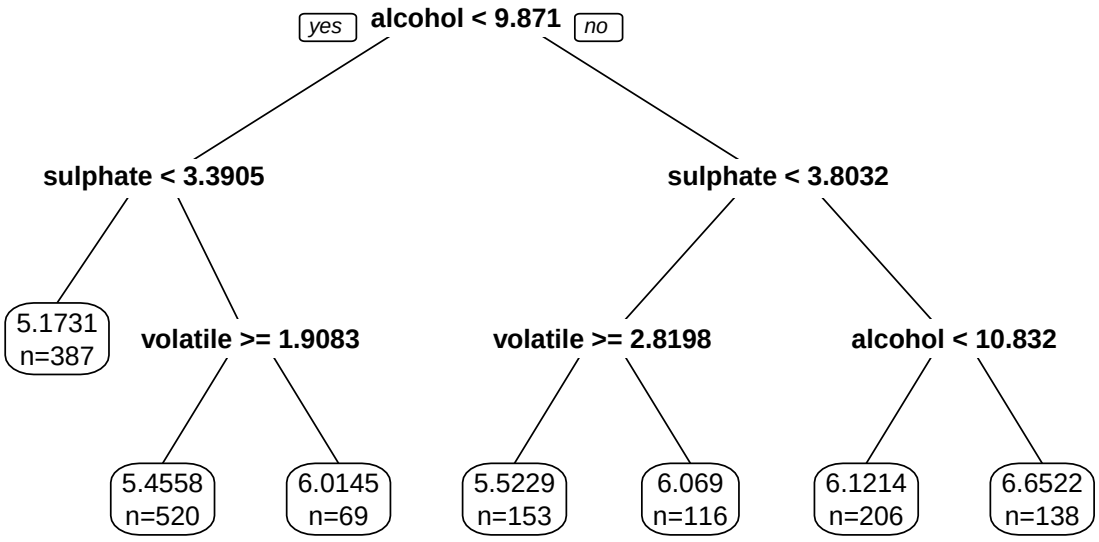
Regression Tree

```
library(caret)
library(rpart.plot)
```

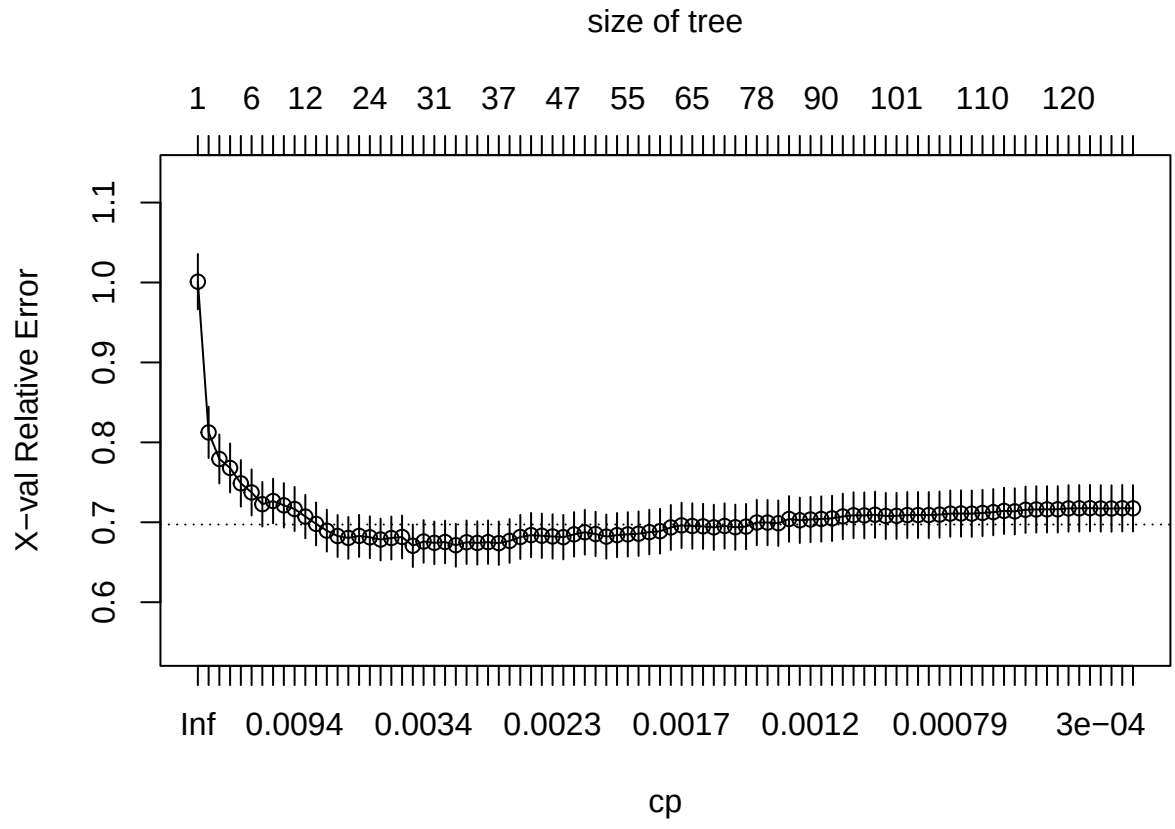
```
## Loading required package: rpart
```

```
library(rpart)
#### Red wine

#build the initial tree
tree <- rpart(quality ~  alcohol+sulphates+volatile.acidity+ total.sulfur.dioxide+ density,
           data = redwine, control=rpart.control(cp=.0001))
```

```
#identify best cp value to use
#produce a pruned tree based on the best cp value
pruned_tree <- prune(tree, cp=0.01244574 )
#plot the pruned tree
prp(pruned_tree,
    faclen=0, #use full names for factor labels
    extra=1, #display number of obs. for each terminal node
    roundint=F, #don't round to integers in output
    digits=5) #display 5 decimal places in output
```

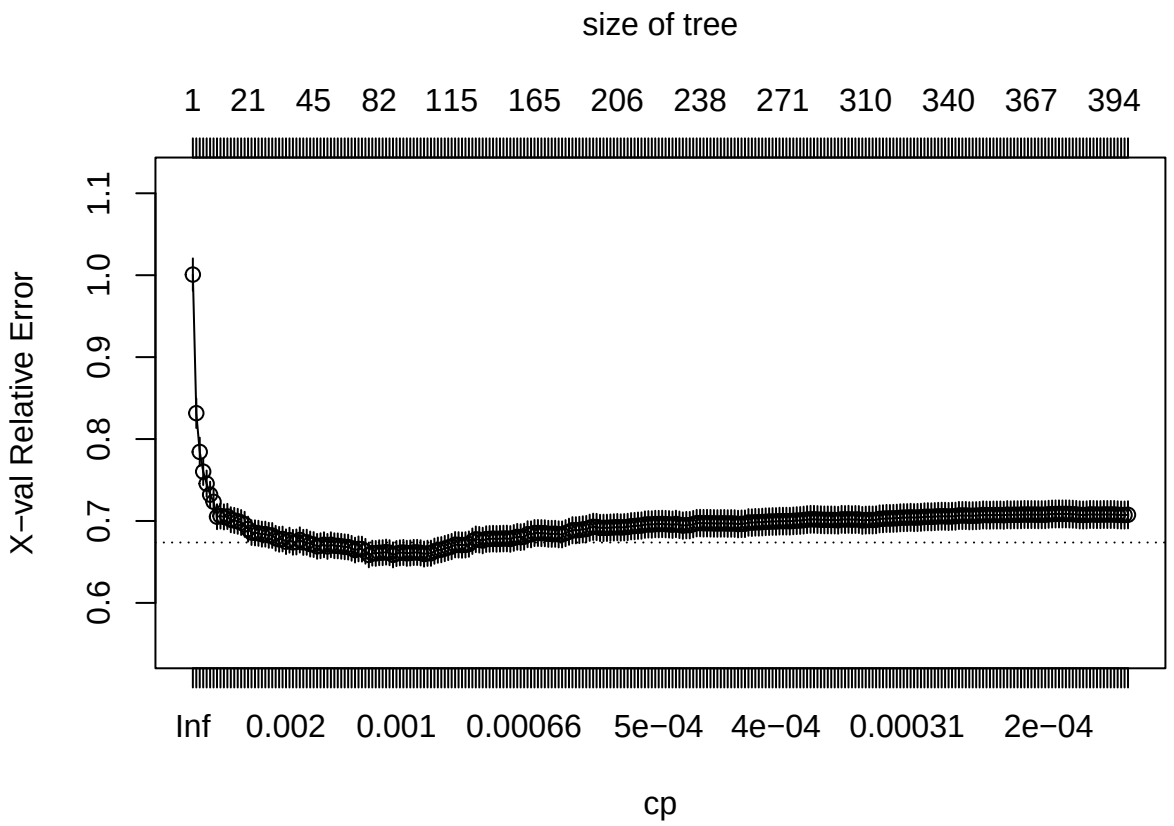



```
#plot the redwine tree
plotcp(tree)
```

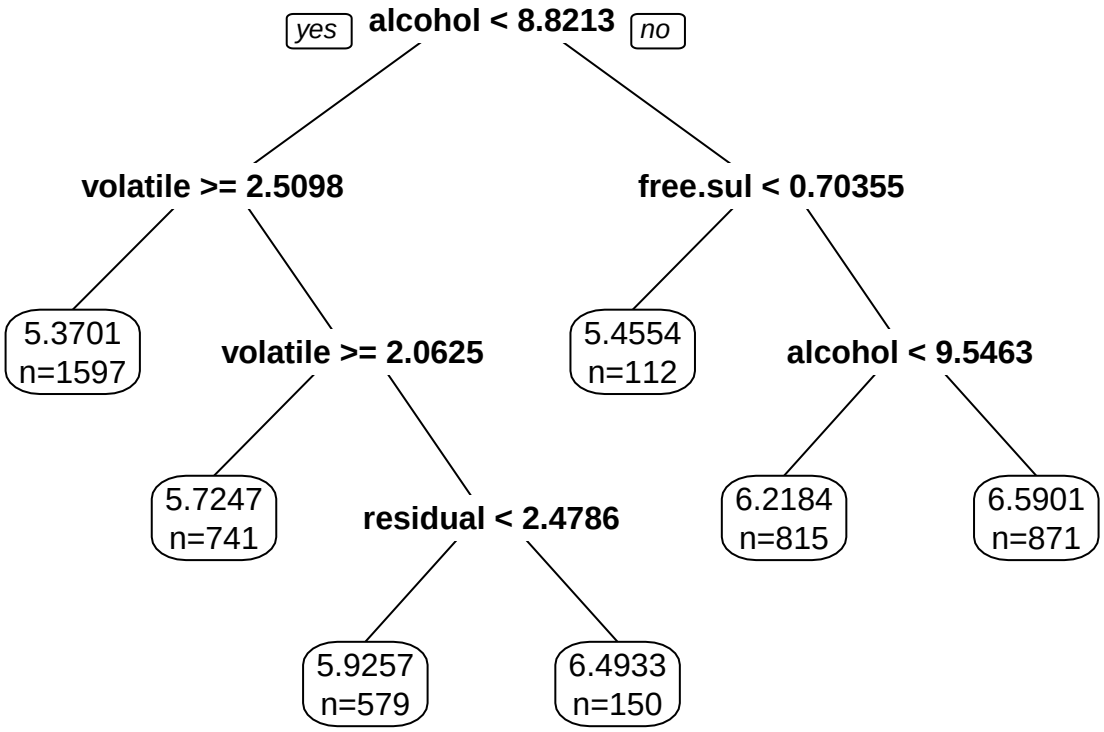



```
#### White wine
#build the initial tree
tree_white <- rpart(quality ~ volatile.acidity + free.sulfur.dioxide +alcohol + pH +residual.sugar, data
               control=rpart.control(cp=.0001))
```

```
plotcp(tree_white)
```




```
#produce a pruned tree based on the best cp value
pruned_tree2 <- prune(tree_white, cp=0.00968719  )

#plot the pruned tree
prp(pruned_tree2,
    faclen=0, #use full names for factor labels
    extra=1, #display number of obs. for each terminal node
    roundint=F, #don't round to integers in output
    digits=5) #display 5 decimal places in output
```



Gradient boosting

```
# Red Wine Model
library(gbm)
```

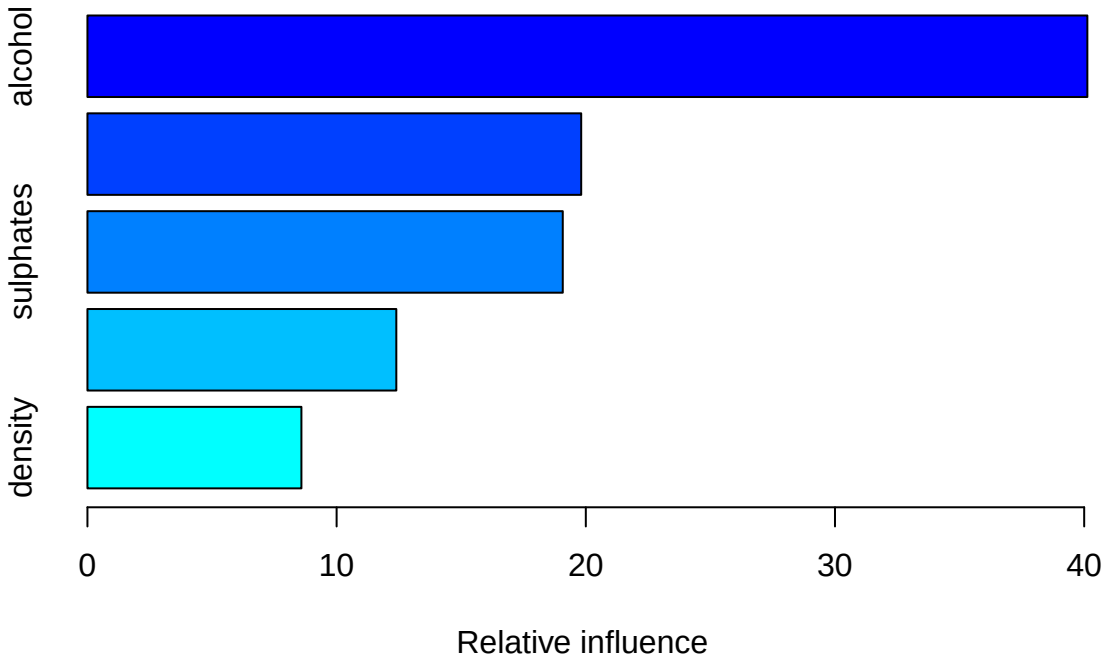
```
## Loaded gbm 2.1.8.1
```

```
library(caret)

# Set the seed for reproducibility
set.seed(123)

# Define the control object for the cross-validation
control <- trainControl(method = "cv",
                        number = 5, # 5-fold cross-validation
                        savePredictions = "final",
                        classProbs = TRUE,
                        summaryFunction = defaultSummary)

# Set up a grid of hyperparameters to search
gbm_grid <- expand.grid(interaction.depth = c(1, 2, 3), # Tune the tree depth
                        n.trees = c(50, 100, 150),      # Tune the number of trees
                        shrinkage = c(0.01, 0.1, 0.1),  # Tune the learning rate
                        n.minobsinnode = 10)

# Train the model with cross-validation
gbm_cv_model <- train(quality ~  density + volatile.acidity
                        + total.sulfur.dioxide + sulphates + alcohol,
```

```
                      data = redwine,
                      method = "gbm",
                      trControl = control,
                      tuneGrid = gbm_grid,
                      distribution = "gaussian",
                      verbose = FALSE)
```

```
## Warning in train.default(x, y, weights = w, ...): cannnot compute class
## probabilities for regression
```

```
# Print the model details and best hyperparameters
#print(gbm_cv_model)
#print(gbm_cv_model$results)
summary (gbm_cv_model)
```




```
##                                     var   rel.inf
## alcohol                         alcohol 40.127569
## volatile.acidity       volatile.acidity 19.814810
## sulphates                     sulphates 19.075515
## total.sulfur.dioxide total.sulfur.dioxide 12.395155
## density                         density  8.586951
```

```
# Extract the cross-validated predictions from the model
cv_predictions <- gbm_cv_model$pred

# Calculate the difference between the observed and predicted values
squared_errors <- (cv_predictions$obs - cv_predictions$pred)^2

# Calculate the mean of the squared errors
mse <- mean(squared_errors)
```

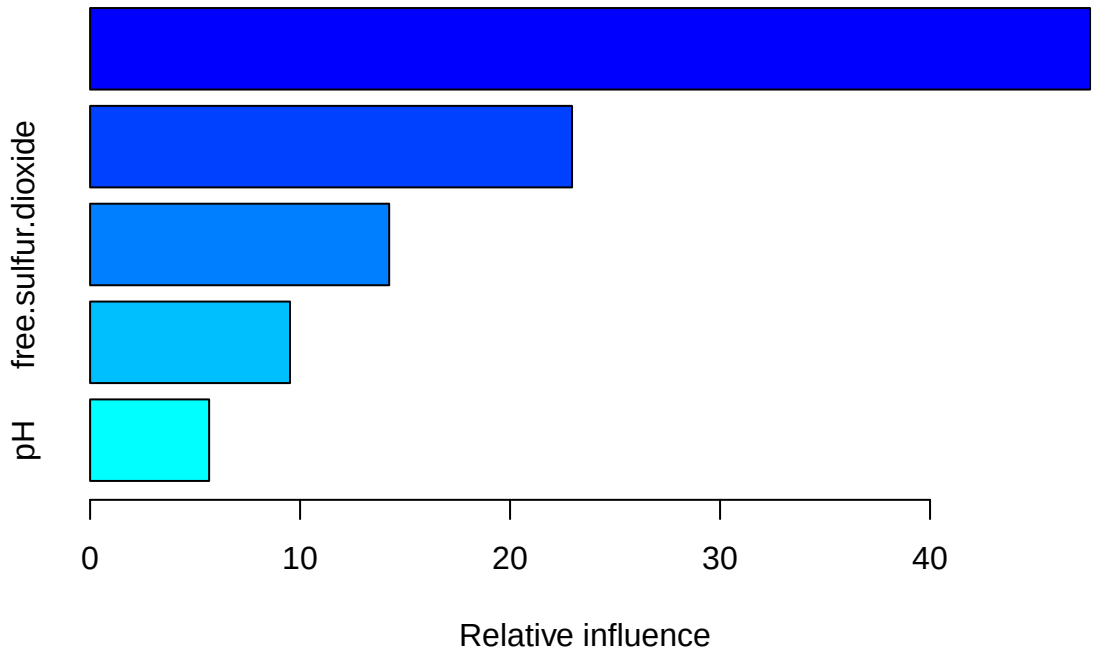
```r
# Print the Mean Squared Error
print(paste("Mean Squared Error (MSE):", mse))
```

```
## [1] "Mean Squared Error (MSE): 0.367422738373722"
```

```r
# White Wine Model
library(gbm)
library(caret)

# Set the seed for reproducibility
set.seed(123)

# Define the control object for the cross-validation
control <- trainControl(method = "cv",
                        number = 5, # 5-fold cross-validation
                        savePredictions = "final",
                        classProbs = TRUE,
                        summaryFunction = defaultSummary)

# Set up a grid of hyperparameters to search
gbm_grid <- expand.grid(interaction.depth = c(1, 2, 3), # Tune the tree depth
                        n.trees = c(50, 100, 150),      # Tune the number of trees
                        shrinkage = c(0.01, 0.1, 0.1),  # Tune the learning rate
                        n.minobsinnode = 10)

# Train the model with cross-validation
gbm_cv_model <- train(quality ~  pH + volatile.acidity
                          + residual.sugar + free.sulfur.dioxide
                          + alcohol,
                      data = whitewine,
                      method = "gbm",
                      trControl = control,
                      tuneGrid = gbm_grid,
                      distribution = "gaussian",
                      verbose = FALSE)
```

```
## Warning in train.default(x, y, weights = w, ...): cannnot compute class
## probabilities for regression
```

```r
# Print the model details and best hyperparameters
summary (gbm_cv_model)
```




```
##                                   var   rel.inf
## alcohol                       alcohol 47.619919
## volatile.acidity     volatile.acidity 22.950685
## free.sulfur.dioxide free.sulfur.dioxide 14.242893
## residual.sugar         residual.sugar  9.519144
## pH                                 pH  5.667359
```

```r
# Extract the cross-validated predictions from the model
cv_predictions <- gbm_cv_model$pred

# Calculate the difference between the observed and predicted values
squared_errors <- (cv_predictions$obs - cv_predictions$pred)^2

# Calculate the mean of the squared errors
mse <- mean(squared_errors)

# Print the Mean Squared Error
print(paste("Mean Squared Error (MSE):", mse))
```

```
## [1] "Mean Squared Error (MSE): 0.464876298705148"
```

Final Project (STSCI 4740)

Nicholas Paschall

2023-04-24

RFE

Red wine RFE

```
X <- redwine[,1:11]
y <- redwine[,13]
# Set up the control object for RFE
rfe_ctrl <- rfeControl(functions = rfFuncs,
                       method = "cv",
                       number = 5,  # Number of folds for cross-validation
                       verbose = FALSE)

# Perform RFE using the randomForest model
set.seed(123)
rfe_results <- rfe(X, y,
                   sizes = 1:ncol(X),  # Number of features to select at each step
                   rfeControl = rfe_ctrl)
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

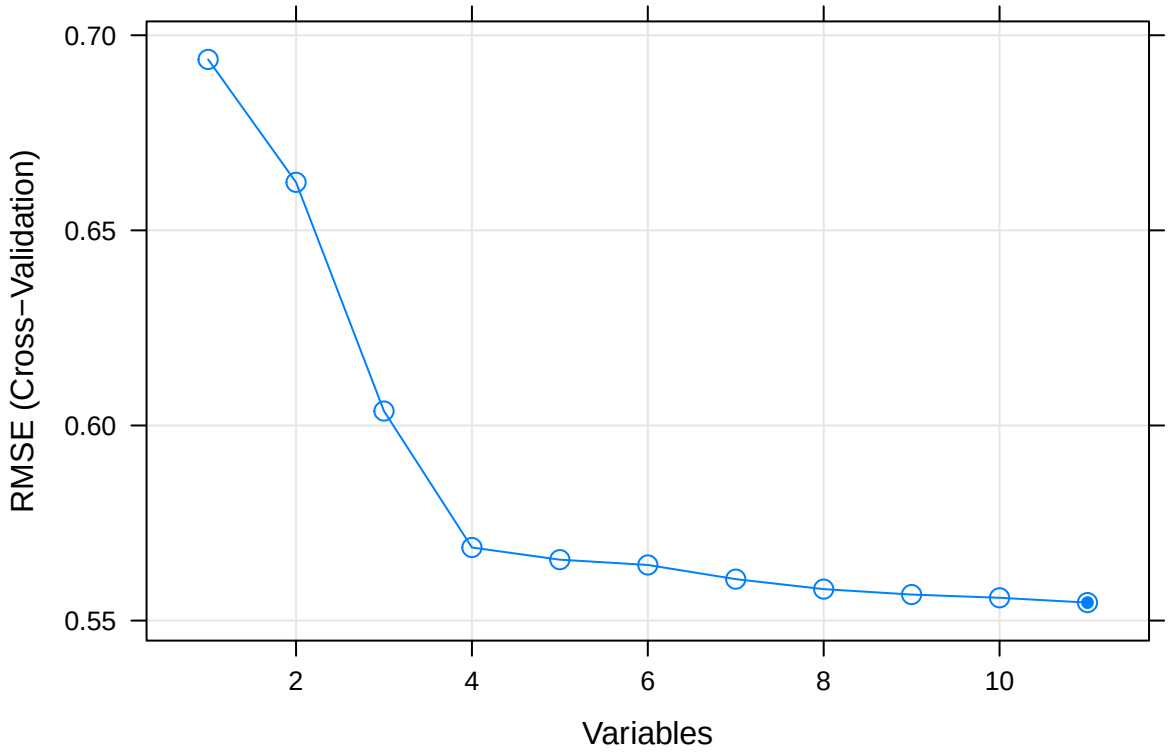
```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```r
# Print the results
print(rfe_results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (5 fold)
##
## Resampling performance over subset size:
##
##  Variables   RMSE Rsquared    MAE  RMSESD RsquaredSD   MAESD Selected
##          1 0.6938   0.2229 0.5551 0.03013    0.06556 0.01571
##          2 0.6623   0.2964 0.5148 0.03169    0.05702 0.02249
##          3 0.6037   0.4068 0.4610 0.03306    0.05633 0.02204
##          4 0.5687   0.4762 0.4293 0.03037    0.06715 0.02375
```

```
##          5 0.5656   0.4835 0.4259 0.03073    0.06363 0.02553
##          6 0.5643   0.4829 0.4210 0.03009    0.05849 0.02672
##          7 0.5606   0.4909 0.4202 0.03028    0.06040 0.02687
##          8 0.5581   0.4977 0.4195 0.02920    0.05943 0.02645
##          9 0.5567   0.4983 0.4165 0.02957    0.06126 0.02752
##         10 0.5558   0.5013 0.4155 0.02546    0.05579 0.02477
##         11 0.5546   0.5050 0.4156 0.02573    0.05688 0.02513        *
##
## The top 5 variables (out of 11):
##    alcohol, sulphates, volatile.acidity, total.sulfur.dioxide, density
```

```r
# Plot the results
plot(rfe_results, type = c("g", "o"), cex = 1.2)
```



White wine RFE

```r
X <- whitewine[,1:11]
y <- whitewine[,13]
# Set up the control object for RFE
rfe_ctrl <- rfeControl(functions = rfFuncs,
                       method = "cv",
                       number = 5,  # Number of folds for cross-validation
                       verbose = FALSE)

# Perform RFE using the randomForest model
set.seed(123)
rfe_results <- rfe(X, y,
```

```r
                   sizes = 1:ncol(X),  # Number of features to select at each step
                   rfeControl = rfe_ctrl)
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

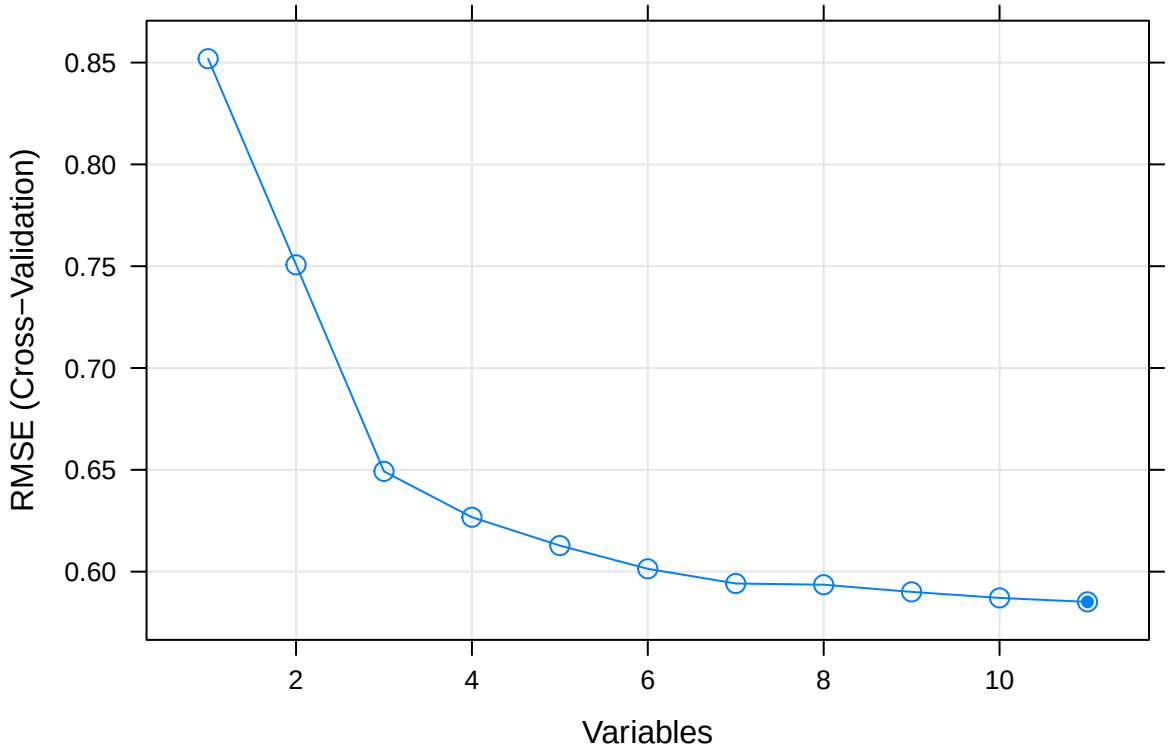
```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```
## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, importance = TRUE, ...): The response has
## five or fewer unique values. Are you sure you want to do regression?
```

```
# Print the results
print(rfe_results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (5 fold)
##
## Resampling performance over subset size:
##
##  Variables   RMSE Rsquared    MAE  RMSESD RsquaredSD    MAESD Selected
##          1 0.8519  0.03317 0.6559 0.01664   0.005999 0.010676
##          2 0.7507  0.24534 0.5808 0.06032   0.117455 0.042391
##          3 0.6492  0.43418 0.5012 0.01083   0.018037 0.008743
##          4 0.6267  0.47650 0.4790 0.01278   0.011872 0.007077
##          5 0.6128  0.50412 0.4648 0.01764   0.016736 0.008239
##          6 0.6014  0.51815 0.4458 0.01581   0.016753 0.008683
##          7 0.5942  0.53209 0.4415 0.01545   0.017193 0.007103
##          8 0.5936  0.53442 0.4405 0.01459   0.014698 0.008873
##          9 0.5901  0.53836 0.4354 0.01373   0.015744 0.009888
##         10 0.5871  0.54443 0.4333 0.01449   0.014844 0.010984
##         11 0.5852  0.54834 0.4325 0.01625   0.013146 0.011585        *
##
## The top 5 variables (out of 11):
##    volatile.acidity, free.sulfur.dioxide, alcohol, pH, residual.sugar
```

```
# Plot the results
plot(rfe_results, type = c("g", "o"), cex = 1.2)
```


RMSE (Cross−Validation)
0.85
0.80
0.75
0.70
0.65
0.60
2
4
6
8
10
Variables