Airfoil Self-Noise - UCI Machine Learning Repository

Abhinay Krishna Vellala

7/18/2020

Contents

1. Data Extraction.

Airfoil Self-Noise dataset is a NASA data set, obtained from a series of aerodynamic and acoustic tests of two and three-dimensional airfoil blade sections conducted in an anechoic wind tunnel.

Attribute Information:

This problem has the following inputs:

1. Frequency, in Hertzs.
2. Angle of attack, in degrees.
3. Chord length, in meters.
4. Free-stream velocity, in meters per second.
5. Suction side displacement thickness, in meters.

The only output is: 6. Scaled sound pressure level, in decibels.

All the arrtibutes above determine the Pressure level in the atmosphere.

```
nasa = read.table(
  "https://archive.ics.uci.edu/ml/machine-learning-databases/00291/airfoil_self_noise.dat")
colnames(nasa) = c("Frequency", "AngleAttack",
                   "ChordLength", "Velocity", "Suction", "Pressure")
str(nasa)
```

```
## 'data.frame':    1503 obs. of  6 variables:
##  $ Frequency  : int  800 1000 1250 1600 2000 2500 3150 4000 5000 6300 ...
##  $ AngleAttack: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ChordLength: num  0.305 0.305 0.305 0.305 0.305 ...
##  $ Velocity   : num  71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 ...
##  $ Suction    : num  0.00266 0.00266 0.00266 0.00266 0.00266 ...
##  $ Pressure   : num  126 125 126 128 127 ...
```

All the variables in the data seem to be continuous. Before, we can conclude anything, we check the correlation.

To check the relation between each variable, correlation between each variable is checked. We know that the target variable is Pressure. So, correlation between each variable with Pressure is checked.

2. Exploratory Data Analysis(EDA)

```
# correlation between Frequency and Pressure
cat("The correlation between each variable is\n")
```

```
## The correlation between each variable is
```

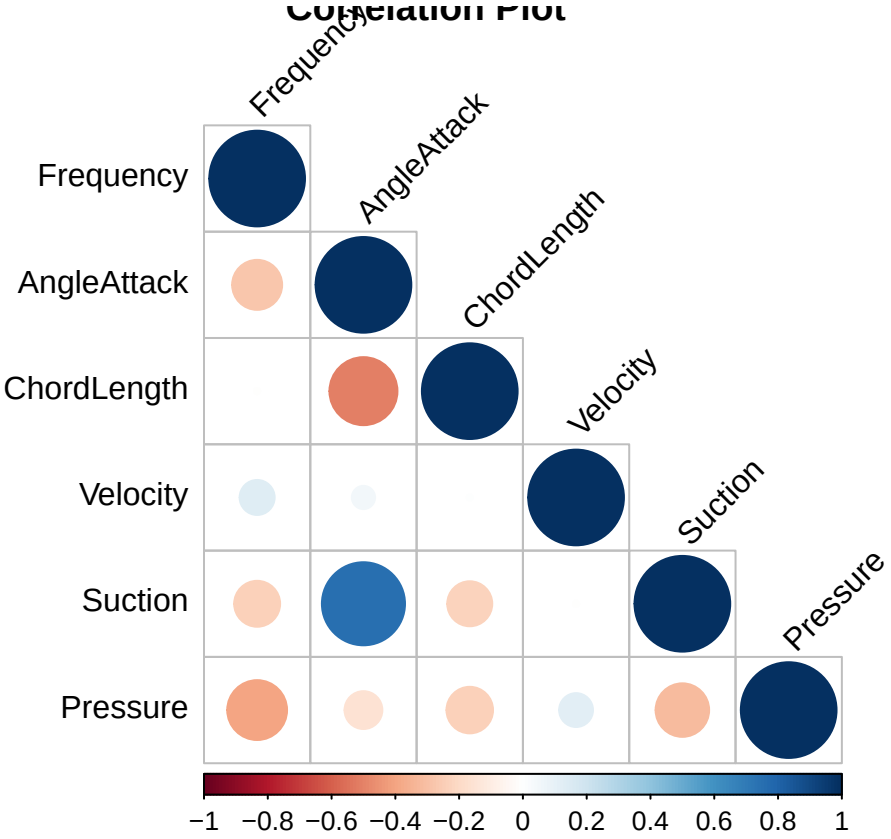
```
print(cor(nasa))
```

```
##                 Frequency AngleAttack  ChordLength     Velocity      Suction
## Frequency    1.000000000 -0.27276454 -0.003660639  0.133663831 -0.230107353
## AngleAttack -0.272764536  1.00000000 -0.504868150  0.058759565  0.753393785
## ChordLength -0.003660639 -0.50486815  1.000000000  0.003786629 -0.220842431
## Velocity     0.133663831  0.05875957  0.003786629  1.000000000 -0.003974013
## Suction     -0.230107353  0.75339378 -0.220842431 -0.003974013  1.000000000
## Pressure    -0.390711412 -0.15610753 -0.236161512  0.125102801 -0.312669506
##                Pressure
## Frequency   -0.3907114
## AngleAttack -0.1561075
## ChordLength -0.2361615
## Velocity     0.1251028
## Suction     -0.3126695
## Pressure     1.0000000
```

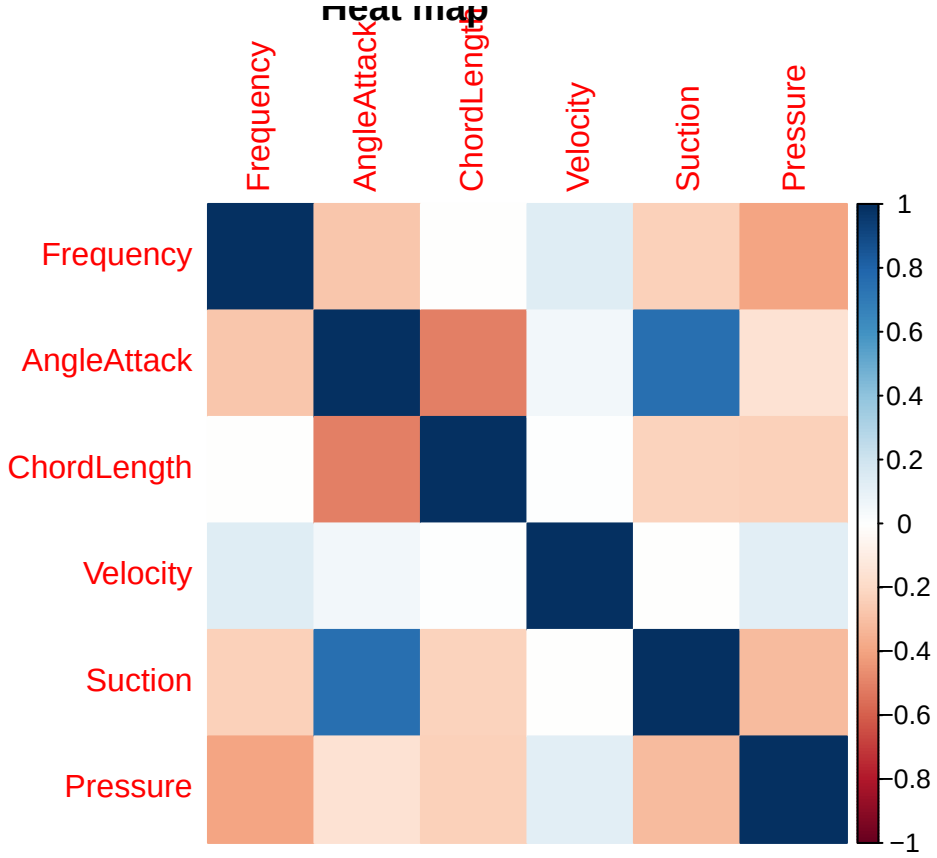
```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(cor(nasa), type = "lower", order = "original", tl.col = "black",
         tl.srt = 45, title = "Correlation Plot")
```



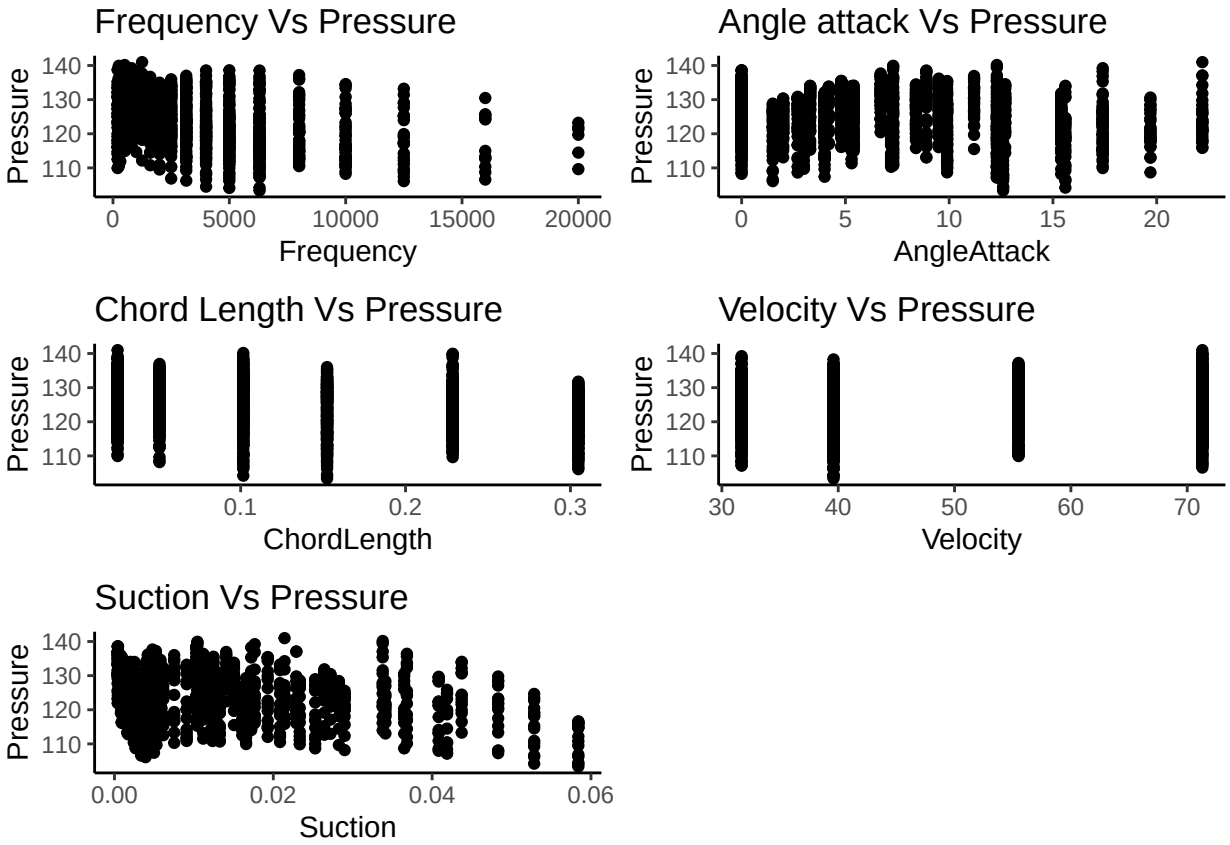

```
corrplot(cor(nasa), method = "color", title = "Heat map")
```



From the correlation plot, it looks like there is no great correclation between each variable and target and also many negative correlations observed. Let's explore more and check the linearity of each varaible with respect to target.

```
library(ggplot2)
library(gridExtra)
p1 = ggplot(nasa, aes(Frequency, Pressure)) + geom_point() + ggtitle("Frequency Vs Pressure") + theme_cl
p2 = ggplot(nasa, aes(AngleAttack, Pressure)) + geom_point() + ggtitle("Angle attack Vs Pressure") + the
p3 = ggplot(nasa, aes(ChordLength, Pressure)) + geom_point() + ggtitle("Chord Length Vs Pressure") + the
p4 = ggplot(nasa, aes(Velocity, Pressure)) + geom_point() + ggtitle("Velocity Vs Pressure") + theme_clas
p5 = ggplot(nasa, aes(Suction, Pressure)) + geom_point() + ggtitle("Suction Vs Pressure") + theme_classi

grid.arrange(p1,p2,p3,p4,p5, nrow = 3)
```



Individually, there is no Linear dependency observed from the plot. However, we'll try using Linear regression and also check if any different model can give better predictions.

3. Model Building

To start building the model, we initially assume to the data to be normally distributed.

$$P(Y = Pressure \mid X = x_1, ..x_5) \sim N(\mu, \Sigma^2)$$

The Likelihood function is the model we are trying to build and it will be

$$L(\mu, \Sigma^2; x_1, ..x_5) = (2\pi\Sigma^2)^{-n/2} exp\left(-\frac{1}{2\Sigma^2}\sum_{j=1}^{n}(x_j - \mu)^2\right)$$

Our goal is to maximize this Likelihood function so that the error is minimum.

Let's divide the data into train and test in the ratio of 70/30.

```
# Divide the data
n = dim(nasa)[1]
set.seed(12345)
id = sample(1:n, floor(0.7*n))
train = nasa[id,]
test = nasa[-id,]
```

3.1 Linear Regression

In Linear regression, the model likelihood is maximized by using the coefficients(β) so that the error is minimum and likelihood is maximum. This is given in the Linear equation.

$$Y = \beta_0 + \beta_1 x_1 + .. + \beta_5 x_5$$

To further simplify this, let us write, $(\beta_0, .., \beta_5)$ as β and $x_1, ..x_5$ as X. The above equation becomes

$$Y = \beta X$$

where X is a multivariate matrix of all the attributes given above and Y is Pressure Levels.

This Linear equation best fits Y which can help us to get the maximum likelihood of the model. To get to this, we need to minimize the error. In linear regression, the error is measured as Sum of Squared Error or Mean Squared Error or Root Mean Squared Error. In this paper, we use Root Mean Squared Error.

The optimal β coefficients will give us the best fitted line which will give least sum of squares and RMSE. Our primary goal in the Linear regression is to determine the optimal Beta coefficients($\hat{\beta}$) which can minimize the error.

This process of regression is performed and root mean squared error is determined.

```
# Linear regression
model = lm(Pressure ~ ., data = train)
summary(model)
```

```
##
## Call:
## lm(formula = Pressure ~ ., data = train)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.2283  -2.9116  -0.3639   3.2592  15.6505
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.323e+02  6.758e-01 195.840   <2e-16 ***
## Frequency   -1.318e-03  5.122e-05 -25.736   <2e-16 ***
## AngleAttack -4.374e-01  4.751e-02  -9.208   <2e-16 ***
## ChordLength -3.465e+01  2.006e+00 -17.273   <2e-16 ***
## Velocity     1.111e-01  9.789e-03  11.347   <2e-16 ***
## Suction     -1.550e+02  1.758e+01  -8.821   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.879 on 1046 degrees of freedom
## Multiple R-squared:  0.5275, Adjusted R-squared:  0.5252
## F-statistic: 233.5 on 5 and 1046 DF,  p-value: < 2.2e-16
```

Every coefficient was given high significance. The residual median is about 0 and standard error not high. The model has given a best fitted line with better standard deviation. Let's predict the fitted values with test data.

```
# prediction
pressurePred = predict(model, newdata = test)
sse = sum((test$Pressure - pressurePred)^2)
mse = sse/nrow(test)
cat("The mean square error of the model is ",mse)
```

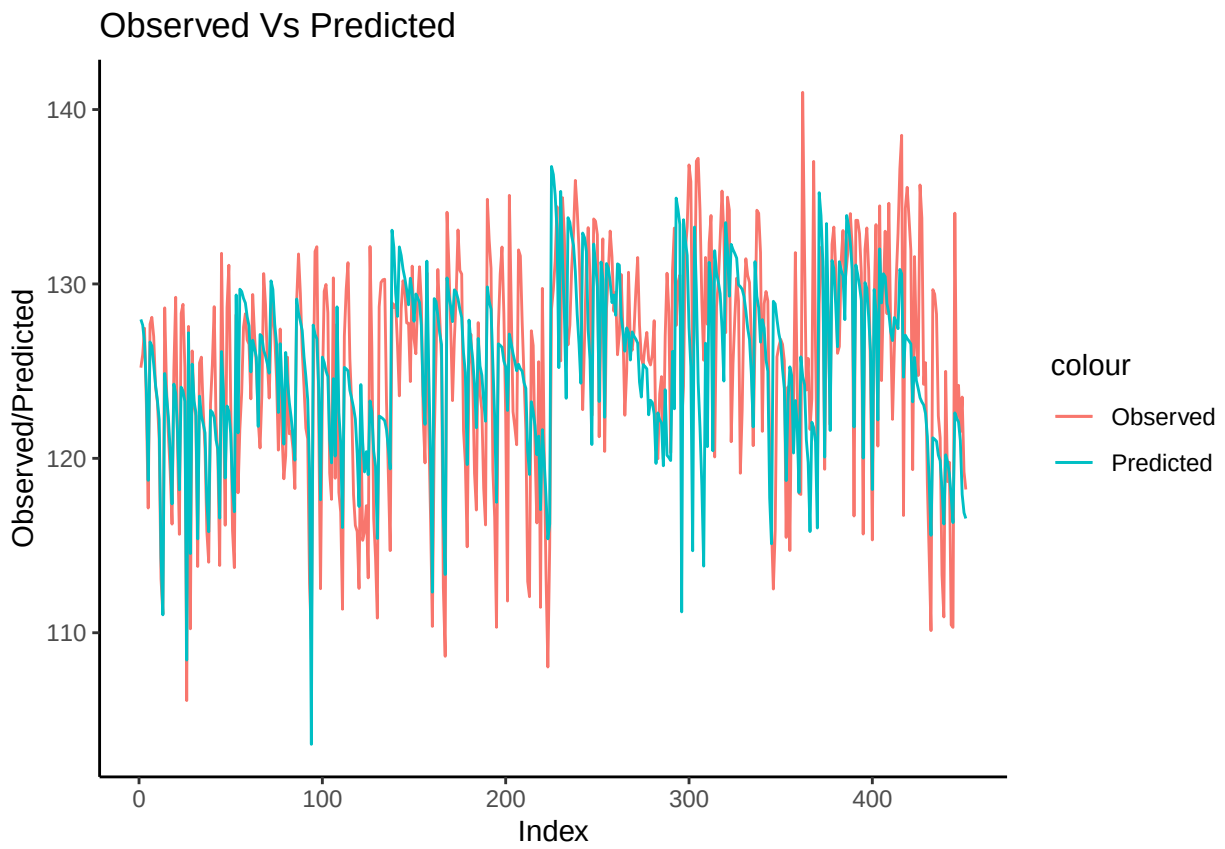
```
## The mean square error of the model is  21.84368
```

```
cat("\nRoot mean squared error of the model is",sqrt(mse))
```

```
##
## Root mean squared error of the model is 4.673722
```

The model has given RMSE as 4.7 which is quite low. Let's plot the observed values vs predicted to check the fianl predicted values.

```
ggplot() + geom_line(aes(1:length(test$Pressure), test$Pressure,color = "Observed")) +
  geom_line(aes(1:length(test$Pressure), pressurePred,color = "Predicted")) +
  xlab("Index") + ylab("Observed/Predicted") +
  ggtitle("Observed Vs Predicted") + theme_classic()
```



The model did a good job in predicting the data as the observed and predicted values overlap actual values frequently.

3.2 Random Forest - Ensemble Method

Even though, Linear regression did a good job in giving us the least error, there is still a chance of getting a better error than Linear regression. The reason is, from the plots, we did not see any linear dependency of Pressure Levels with any attribute. So, Linear regression might be a weak model.

To boost this weak model we use Ensemble method which will combine all the weak models which makes a strong prediction. One of the Ensemble method is Random Forest which uses a technique called Bagging where the data is resampled multiple times and multiple decision trees are grown. The best decision tree is the one with least error.

Here, we have modelled 50 Random Forest models by increasing the threshold of number of decision trees in each model. Increase in the number of decision trees increases the complexity of model which in turn yelid better predictions. More complex models might lead to overfitting which is also observed below.

Let's build a Random Forest and check if that can predict better than Linear regression. While building Random Forest, the number of trees used for buliding Random forest is increased in (10,20,30,. . . ,500)

```
## Decision tree
library(randomForest)
```

```
## randomForest 4.6-14
```

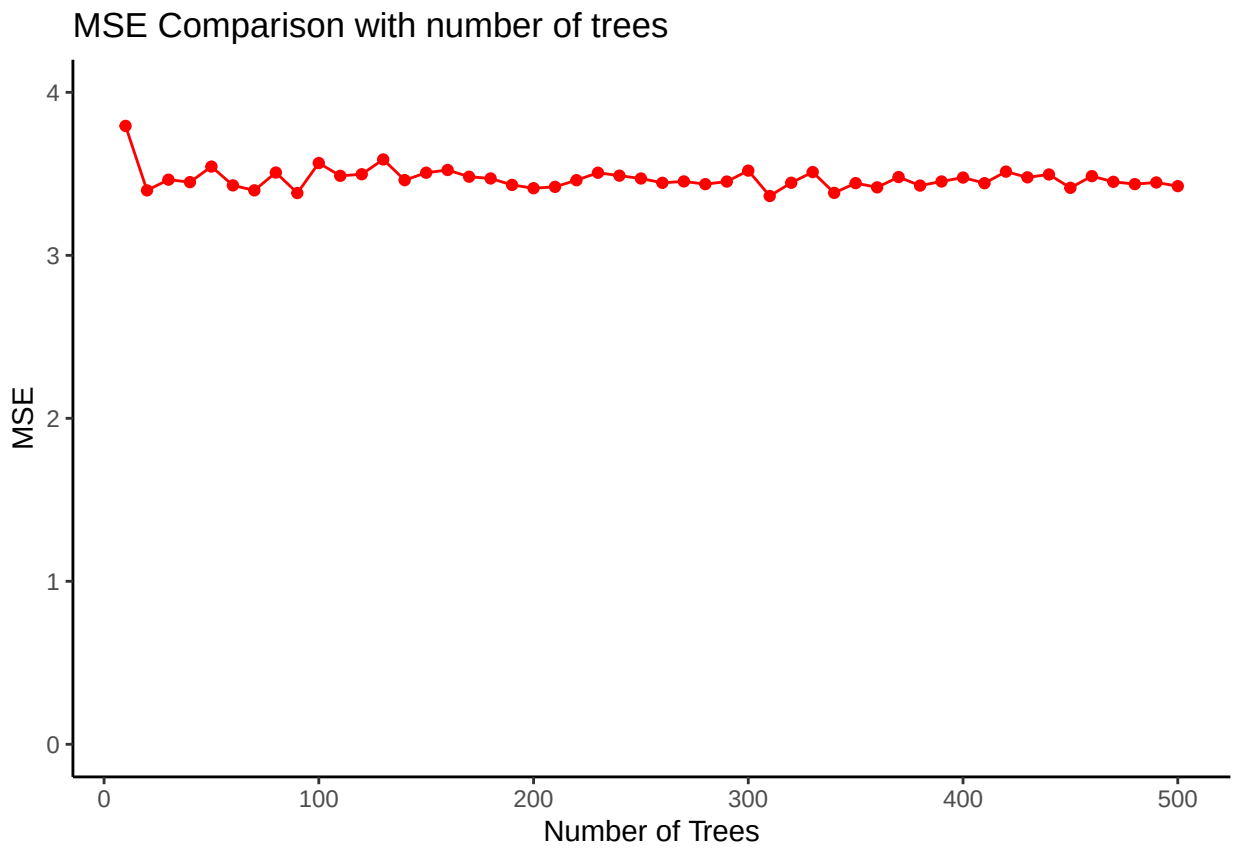
```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
trees = seq(10,500,10)
mseForest = vector()
for (i in 1:length(trees)) {
  modelForest = randomForest(train$Pressure ~ ., data = train, ntree = trees[i])
  newpred = predict(modelForest,newdata = test)
  sseForest = sum((newpred-test$Pressure)^2)
  mseForest[i] = sseForest/length(newpred)
}
ggplot() + geom_point(aes(trees, sqrt(mseForest)), color = "red") +
  geom_line(aes(trees, sqrt(mseForest)), color = "red") +
  xlab("Number of Trees") + ylab("MSE") + ggtitle("MSE Comparison with number of trees") + theme_classi
```



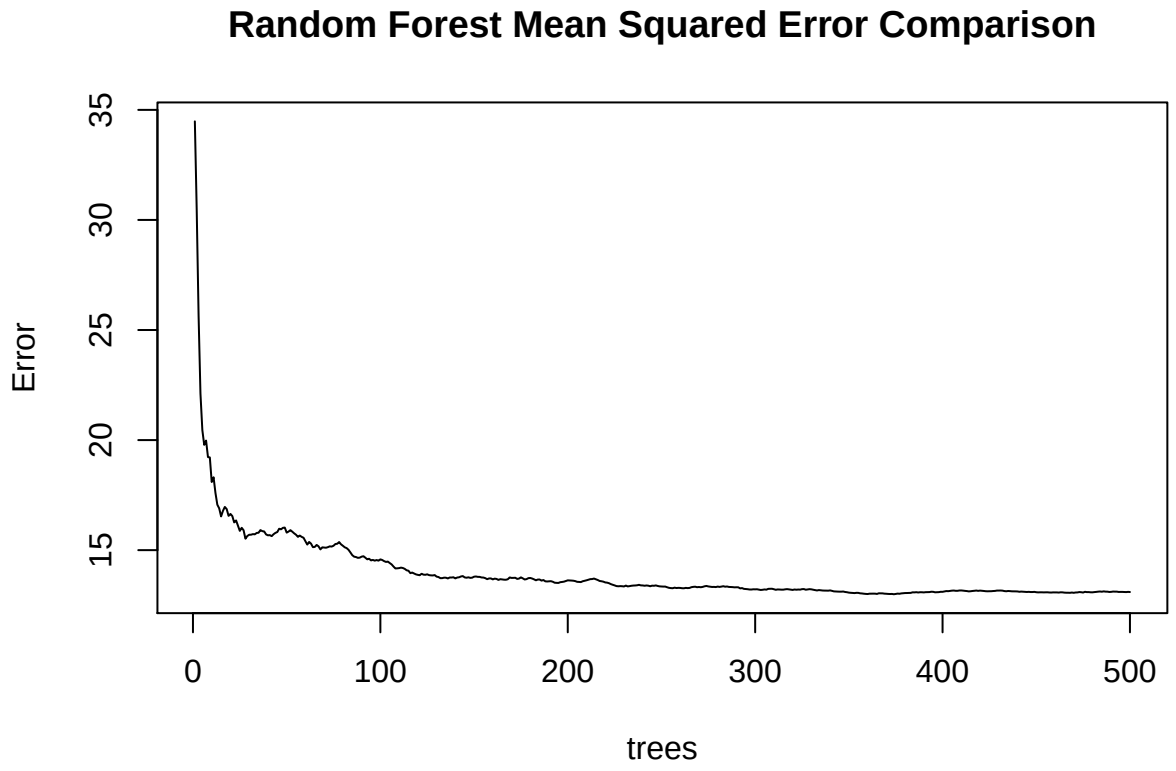
Looks like Random forest did a better job than Linear regression as the RMSE values are below 4 from the first model. The model also stated perfroming bad. This is because of the overfitting problem discussed above.

```
msetable = matrix(nrow = length(trees), ncol = 2)
for (i in 1:length(trees)) {
  msetable[i,1] = trees[i]
  msetable[i,2] = mseForest[i]
}
t = msetable[which(msetable[,2] == min(msetable[,2])),1]

cat("The minimum RMSE is",sqrt(min(msetable[,2])), "and the ideal number of
    trees to predict the data with this RMSE is", t)
```

```
## The minimum RMSE is 3.363957 and the ideal number of
##     trees to predict the data with this RMSE is 310
```

```
plot(modelForest, main = "Random Forest Mean Squared Error Comparison")
```



With increase in the complexity of the Random Forest model, we have achieved the best number of decision trees which can fit the data better and can give good predictions. As we can see after some iterations, the error rate has achieved converegence point.

4. Inference

The data says, Pressure levels is a function of all the given attributes. We have created a fuction of Pressue with respect to β coefficients and created a Linear model. But, as the model is a weak classifier for the data, we have created a Random Forest and got a better predictions for our data.