STA 101: Group Project

Plant Pals (Group 4)

2024-06-3

```r
# reading libaries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(ModelMetrics)
```

```
##
## Attaching package: 'ModelMetrics'
##
## The following object is masked from 'package:base':
##
##     kappa
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
# reading data
# note: data was obtained through a given docx, which I made into a google doc, then copy pasted to goo
# note: the data we were given is about 10% of the data they used, so our graphs will look slightly dif
metasequoia <- read_csv("data/metasequoia_data.csv")
```

```
## Rows: 500 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): tree_number, diameter, height
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

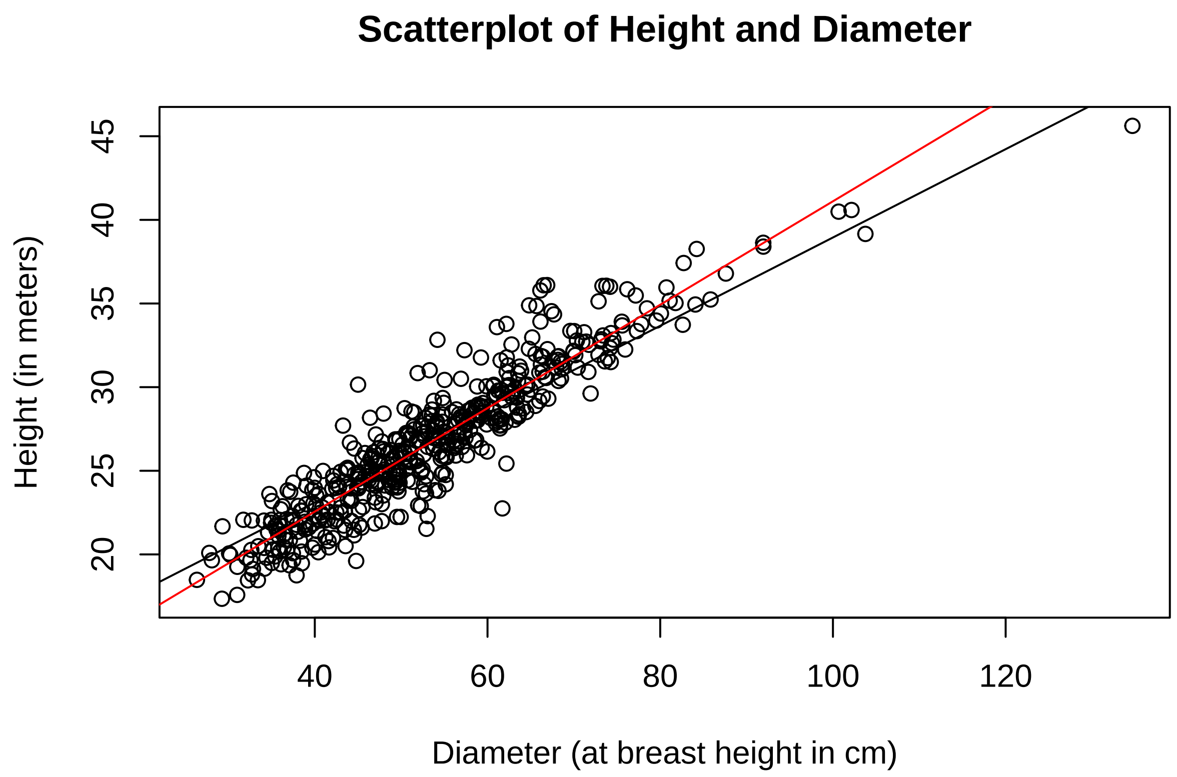
```
# data exploration
metasequoia %>%
  pivot_longer(col = c("height", "diameter"),
               names_to = "datatype",
               values_to = "values") %>%
  group_by(datatype) %>%
  summarise(mean = mean(values),
            max = max(values),
            min = min(values),
            sd = sd(values)) %>%
  t()
```

```
##          [,1]       [,2]
## datatype "diameter" "height"
## mean     "53.36036" "26.69130"
## max      "134.68"   " 45.62"
## min      "26.35"    "17.35"
## sd       "13.33801" " 4.44614"
```

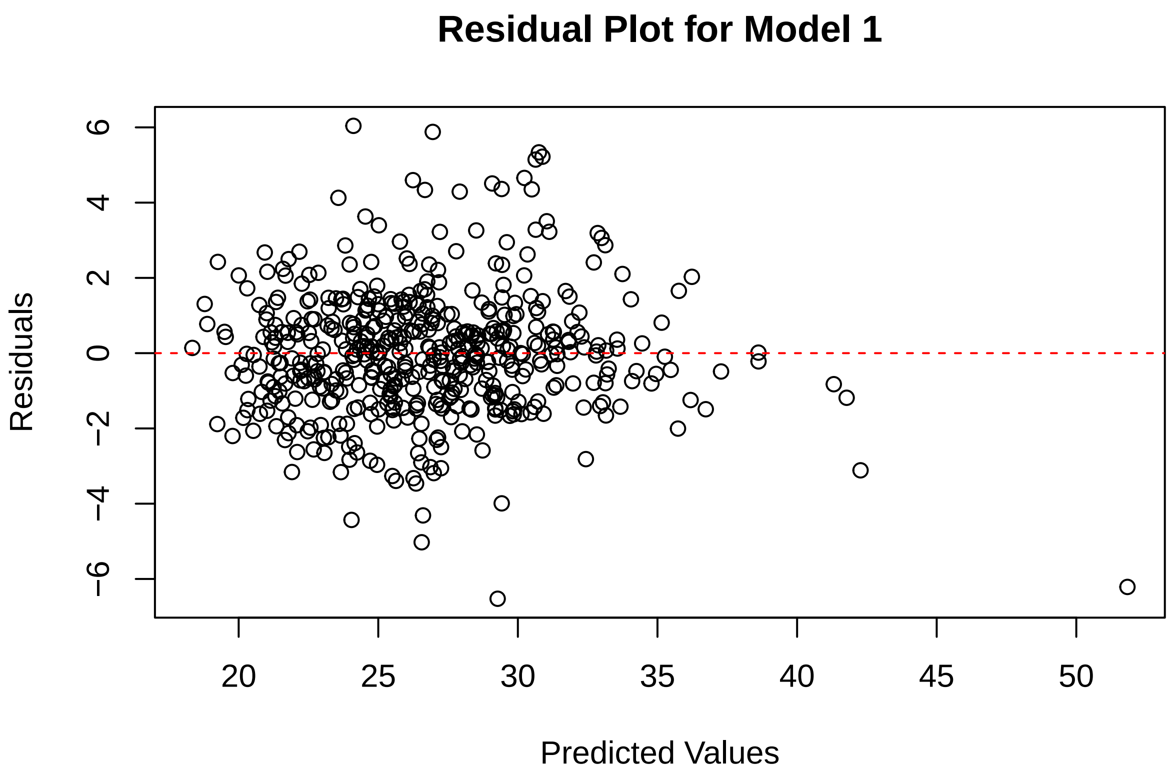
```
# models
metasequoia_model1 <- lm(height ~ diameter, data = metasequoia)
metasequoia_model2 <- lm(height ~ I(log(diameter)), data = metasequoia)
metasequoia_model3 <- lm(height ~ diameter + I(diameter^2), data = metasequoia)
metasequoia_model4 <- lm(height ~ I(diameter^2) + I(diameter^3), data = metasequoia)
metasequoia_model5 <- lm(height ~ I(diameter^-1) + I(diameter^2), data = metasequoia)
# about non-linear models: not sure how to do it and this code is broken
# metasequoia_model8 <- nls(height ~ 1.3 + a1 * (1 - exp(-a1 * diameter))^a2, data = metasequoia, start
```

Model 1: $Y = 10.1942 + 0.3092x$ Model 2: $Y = -39.31 + 16.72log(x)$ Model 3: $Y = 7.4610696 + 0.4066729 - 0.0008166x^2$ Model 4: $Y = 15.75 + 0.005308x^2 + 0.00002802x^3$ Model 5: $Y = 30.73 - 411.7x^{-1} + 0.001373x^2$

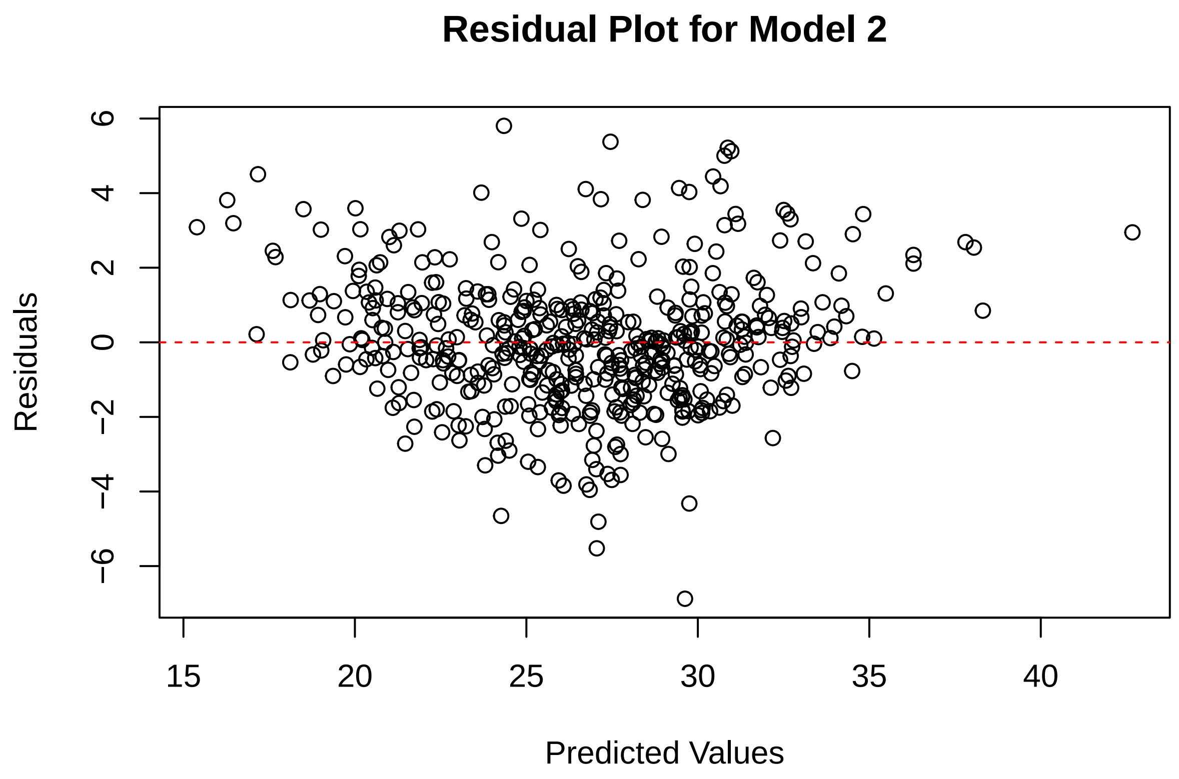
```
# Fig 2. Scatter diagram of the tree height and dbh of a single Metasequoia tree.
plot(height ~ diameter, data = metasequoia, main = "Scatterplot of Height and Diameter", xlab = "Diamete
abline(a = 12.546, b = 0.264) # the paper's data's trendline
abline(metasequoia_model1, col = "red") # trendline for model 1
```



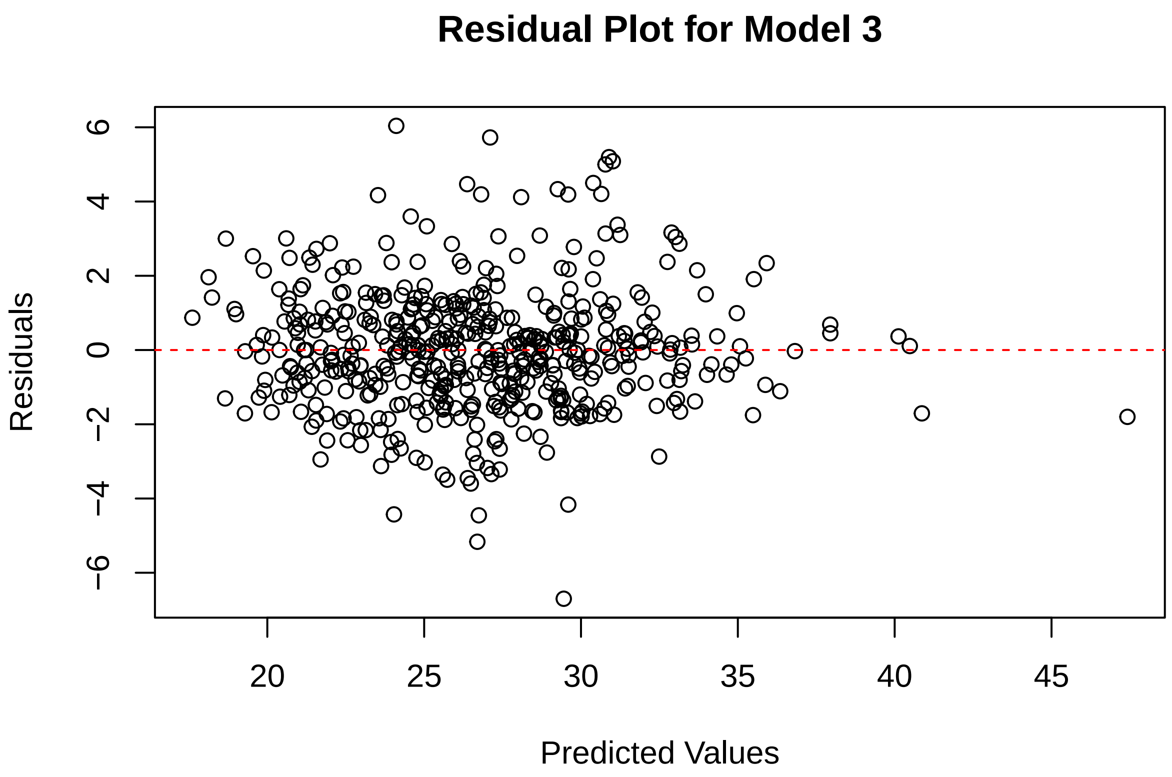
```
#par(mfrow = c(2, 3))
# making residuals plot for model 1
plot(resid(metasequoia_model1) ~ predict(metasequoia_model1), main = "Residual Plot for Model 1", xlab =
abline(h = 0,col = "red",lty = 2)
```



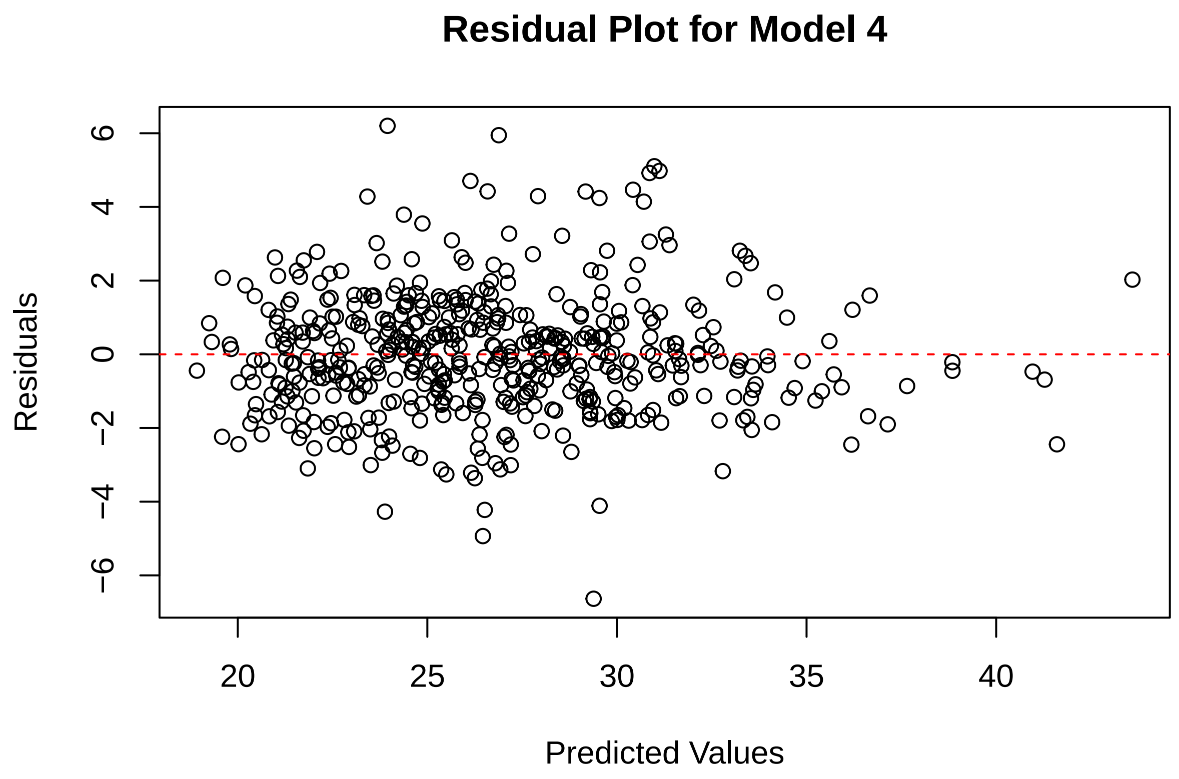
```r
# making residuals plot for model 2
plot(resid(metasequoia_model2) ~ predict(metasequoia_model2), main = "Residual Plot for Model 2", xlab =
abline(h = 0,col = "red",lty = 2)
```



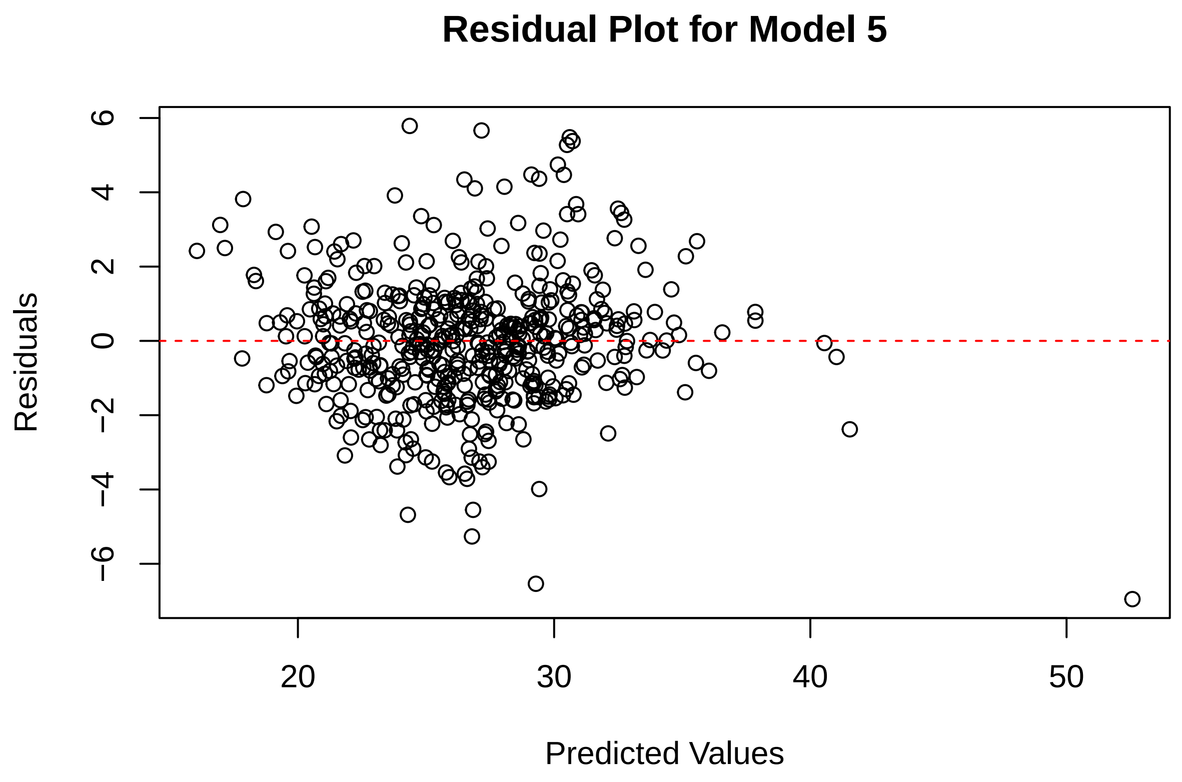
```r
# making residuals plot for model 3
plot(resid(metasequoia_model3) ~ predict(metasequoia_model3), main = "Residual Plot for Model 3", xlab =
abline(h = 0,col = "red",lty = 2)
```



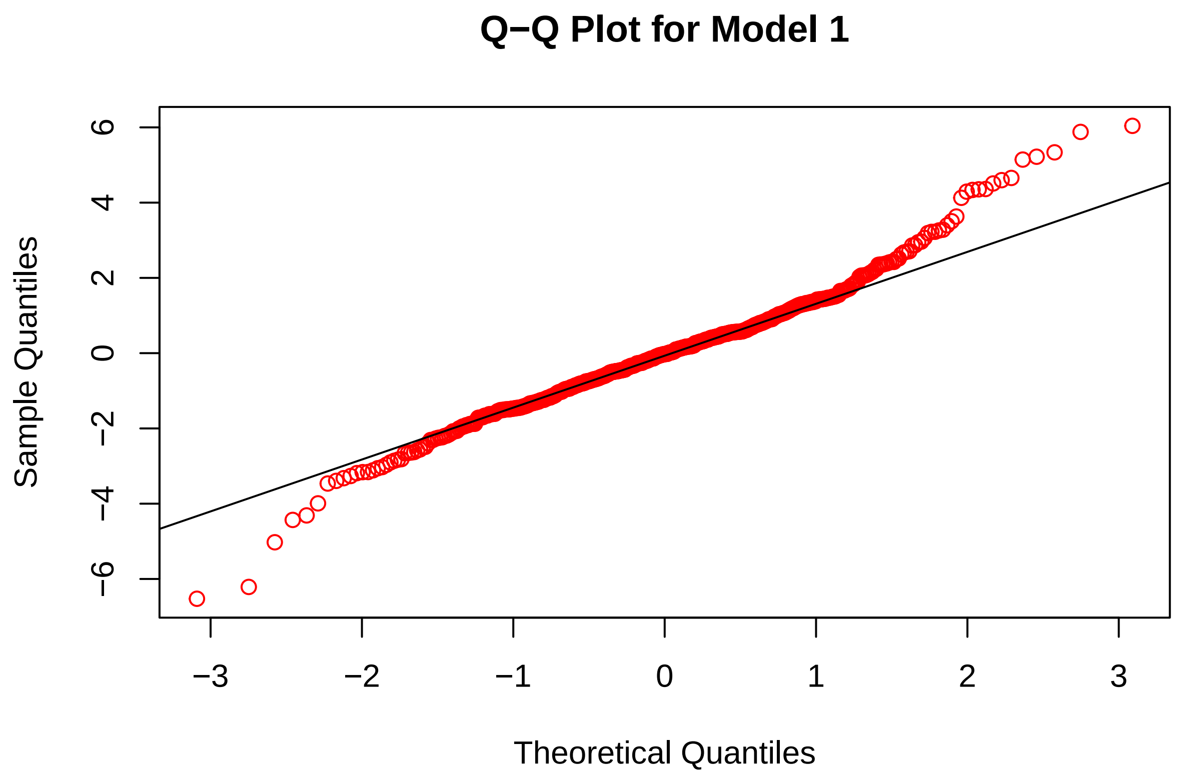
```
# making residuals plot for model 4
plot(resid(metasequoia_model4) ~ predict(metasequoia_model4), main = "Residual Plot for Model 4", xlab =
abline(h = 0,col = "red",lty = 2)
```



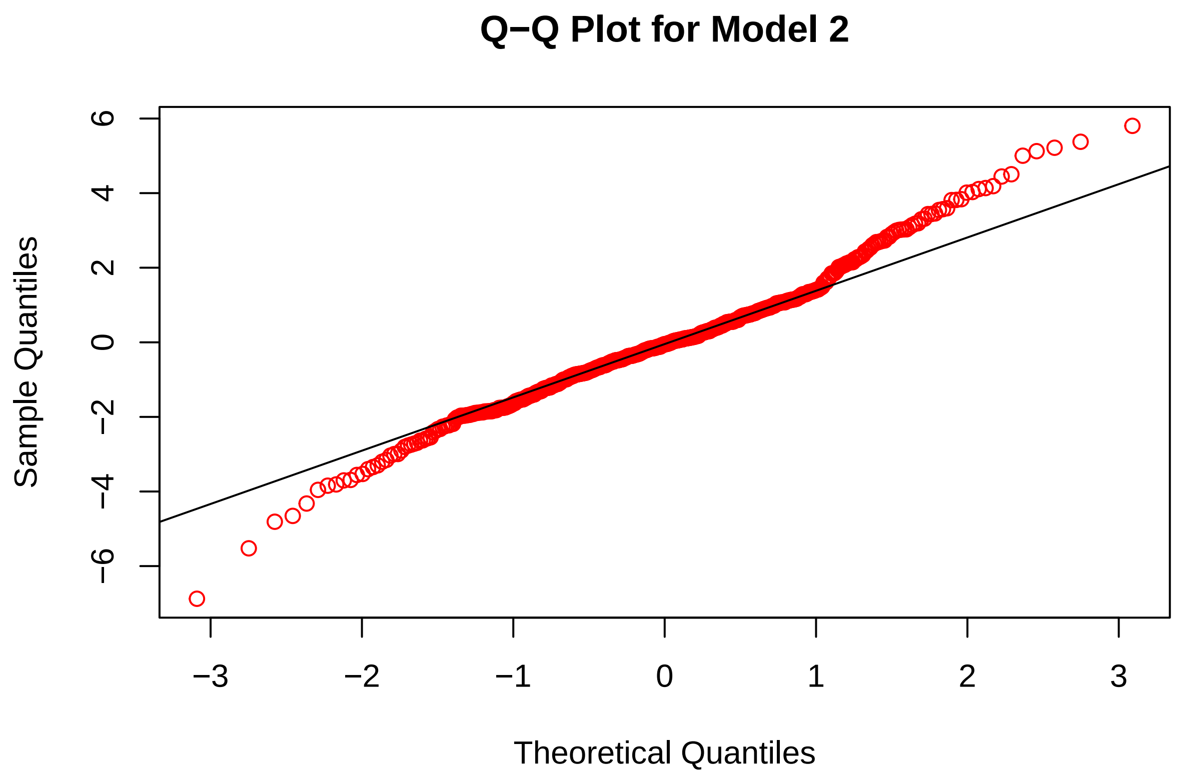
```
# making residuals plot for model 5
plot(resid(metasequoia_model5) ~ predict(metasequoia_model5), main = "Residual Plot for Model 5", xlab =
abline(h = 0,col = "red",lty = 2)
```



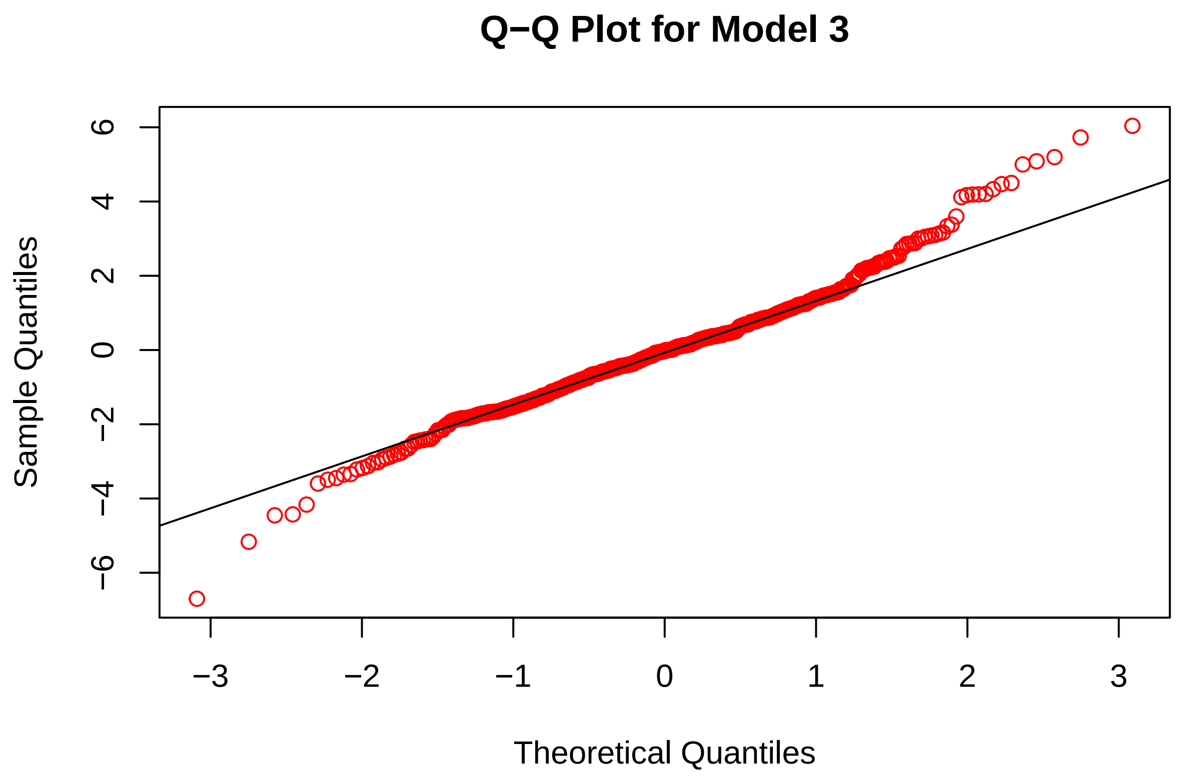
```r
#par(mfrow = c(2, 3))
# making qq plot for model 1
qqnorm(resid(metasequoia_model1), main = "Q-Q Plot for Model 1", col = "red")
qqline(resid(metasequoia_model1))
```



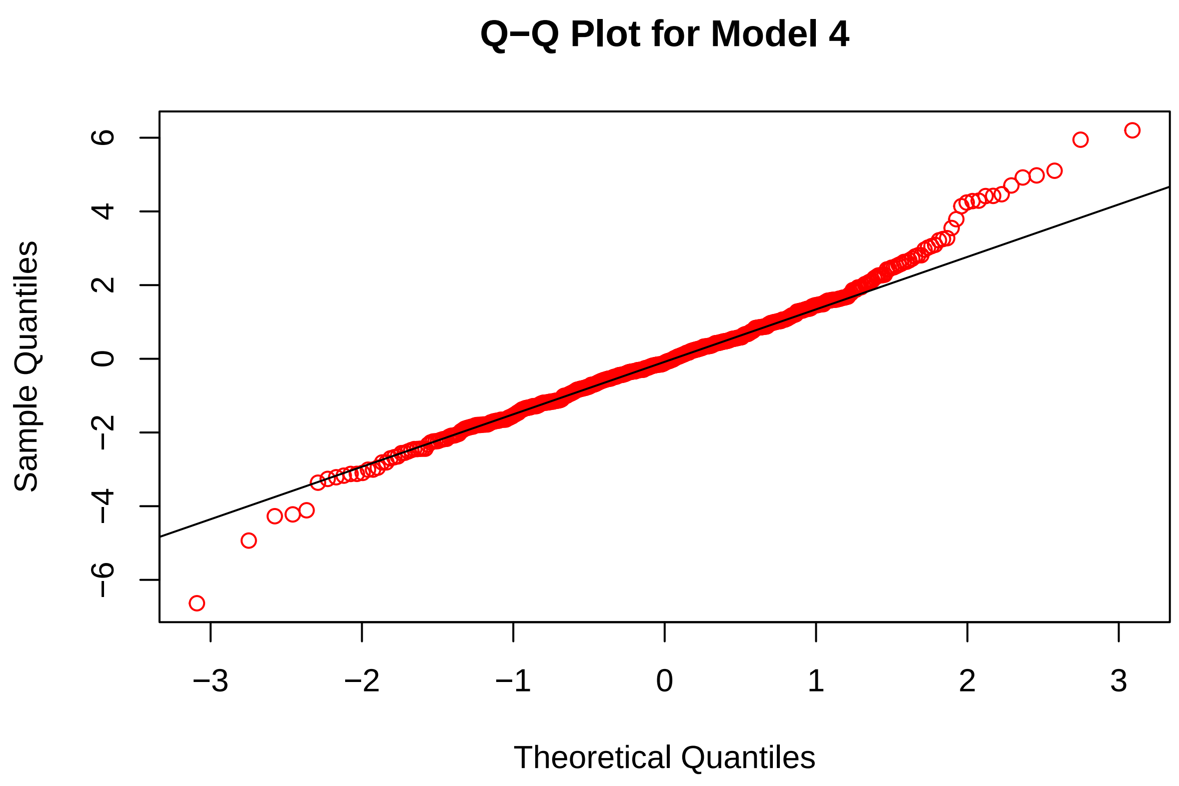
```r
# making qq plot for model 2
qqnorm(resid(metasequoia_model2), main = "Q-Q Plot for Model 2", col = "red")
qqline(resid(metasequoia_model2))
```



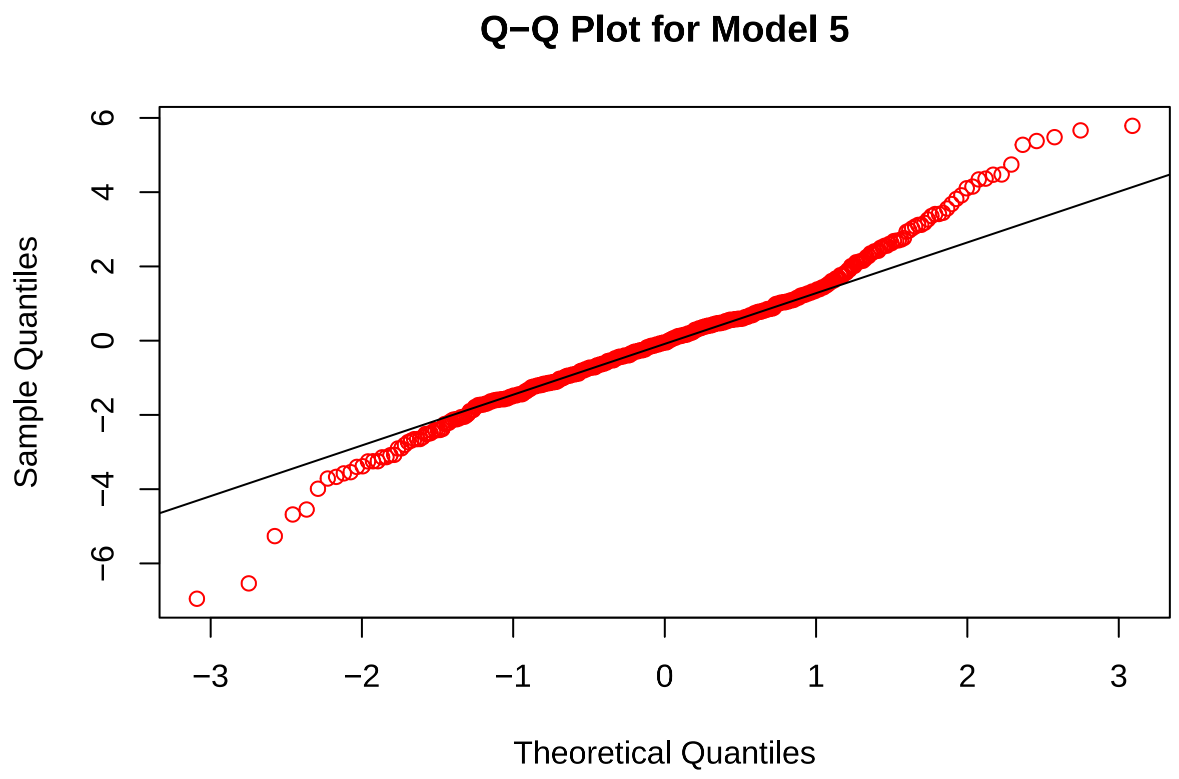
```r
# making qq plot for model 3
qqnorm(resid(metasequoia_model3), main = "Q-Q Plot for Model 3", col = "red")
qqline(resid(metasequoia_model3))
```



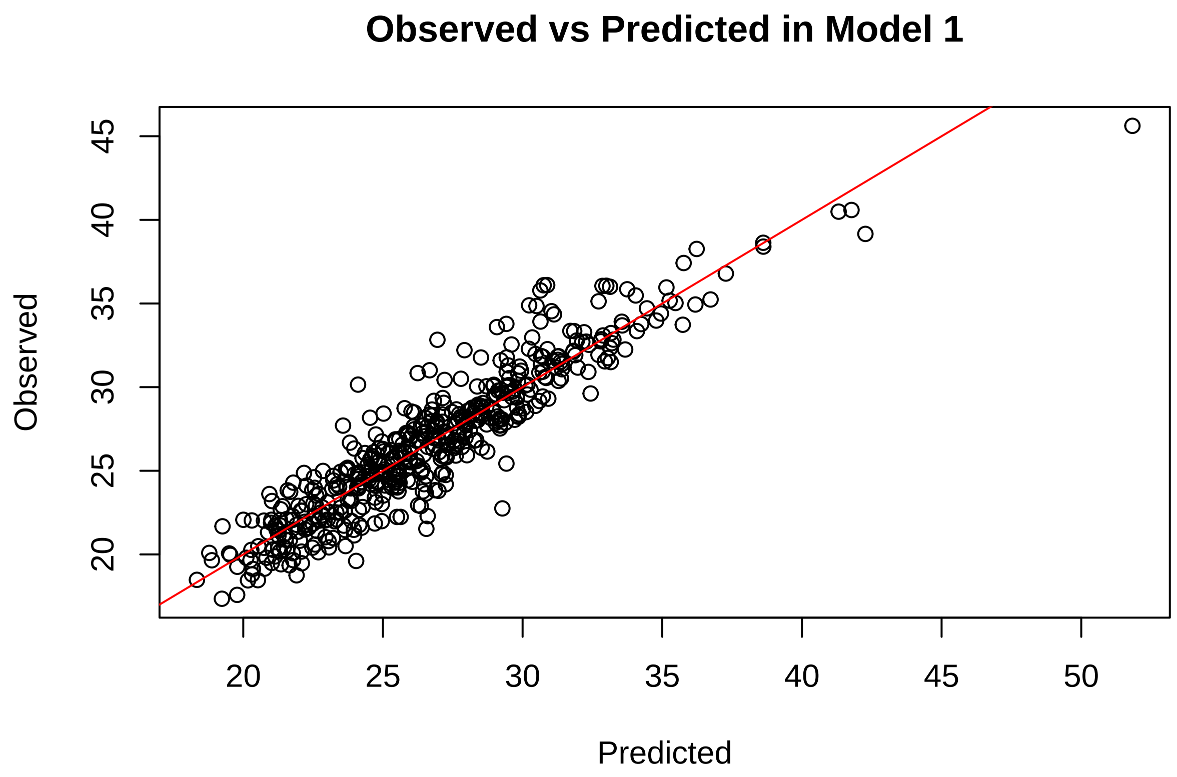
```r
# making qq plot for model 4
qqnorm(resid(metasequoia_model4), main = "Q-Q Plot for Model 4", col = "red")
qqline(resid(metasequoia_model4))
```



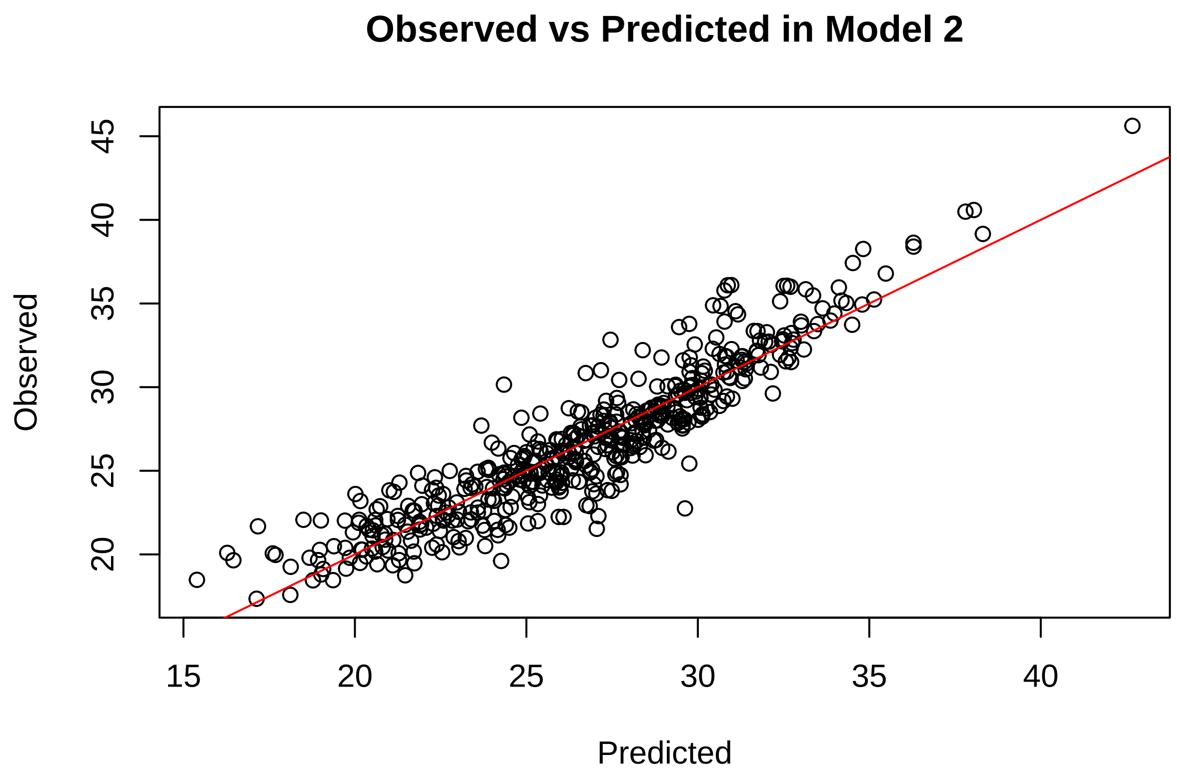
```r
# making qq plot for model 5
qqnorm(resid(metasequoia_model5), main = "Q-Q Plot for Model 5", col = "red")
qqline(resid(metasequoia_model5))
```



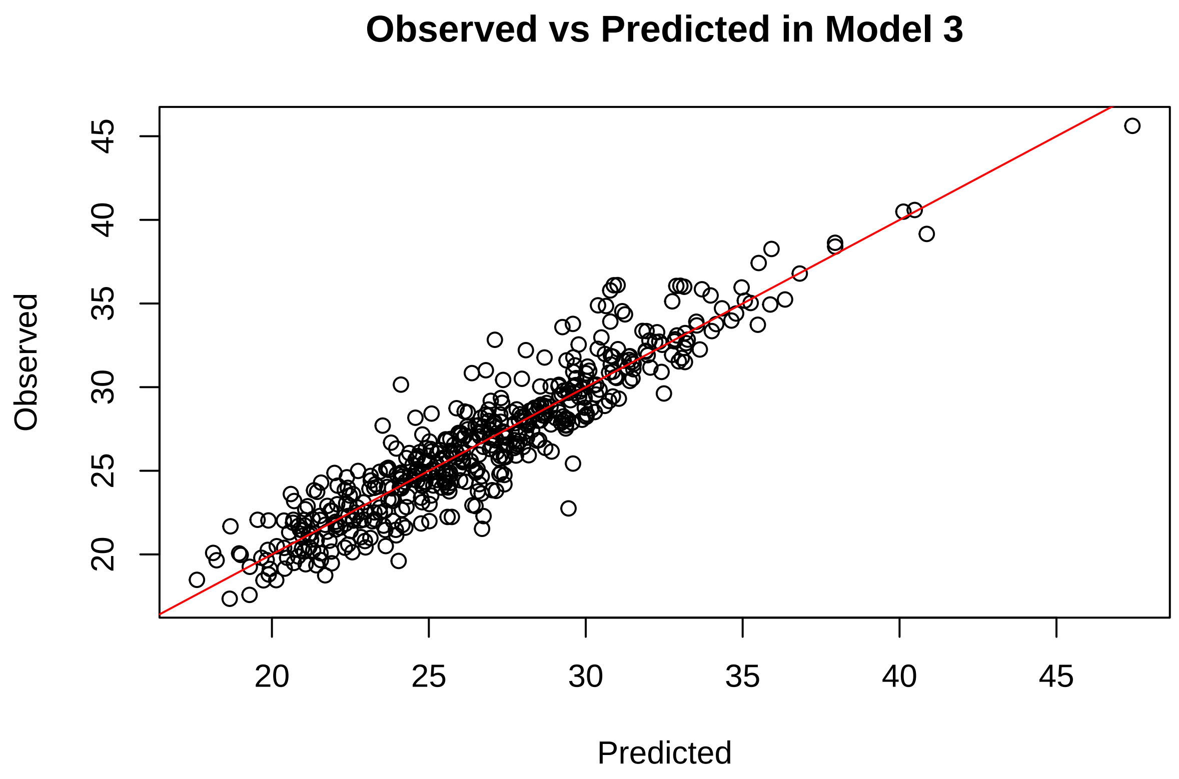
```
#par(mfrow = c(2, 3))
# predicted vs observed for model 1
plot(height ~ predict(metasequoia_model1), data = metasequoia, main = "Observed vs Predicted in Model 1
abline(a = 0, b = 1, col = "red")
```



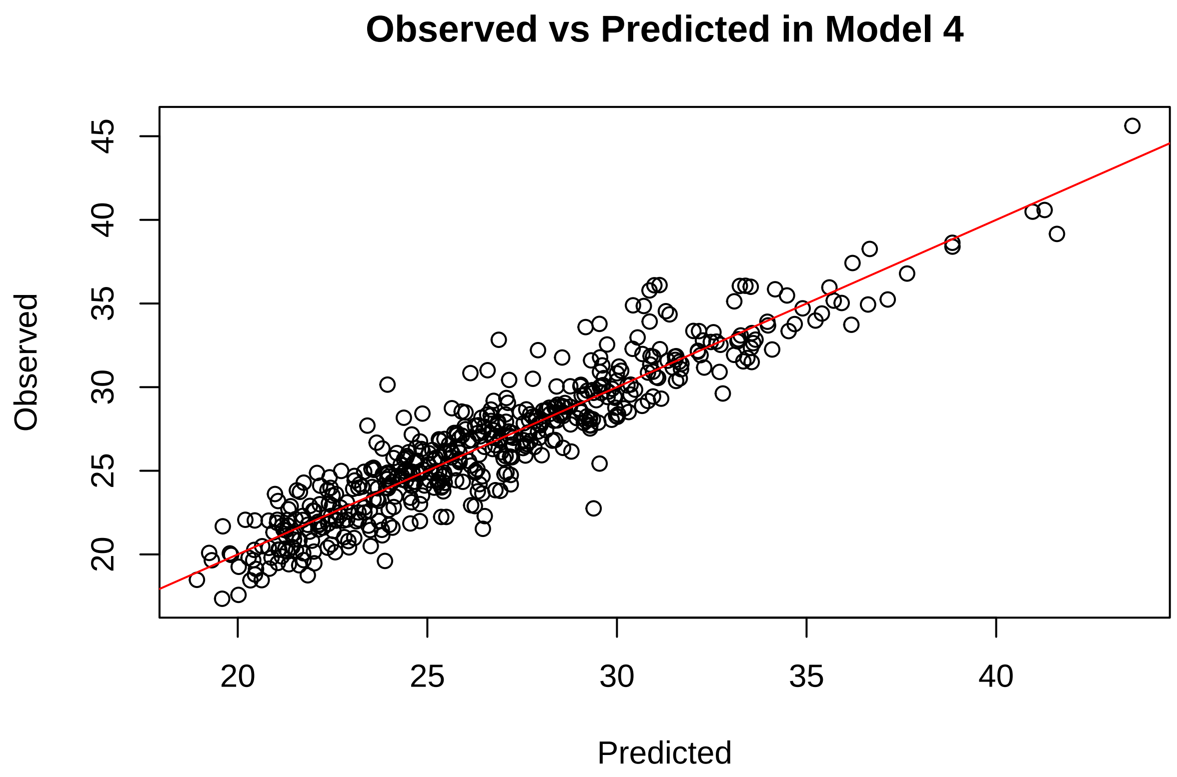
```
# predicted vs observed for model 2
plot(height ~ predict(metasequoia_model2), data = metasequoia, main = "Observed vs Predicted in Model 2
abline(a = 0, b = 1, col = "red")
```



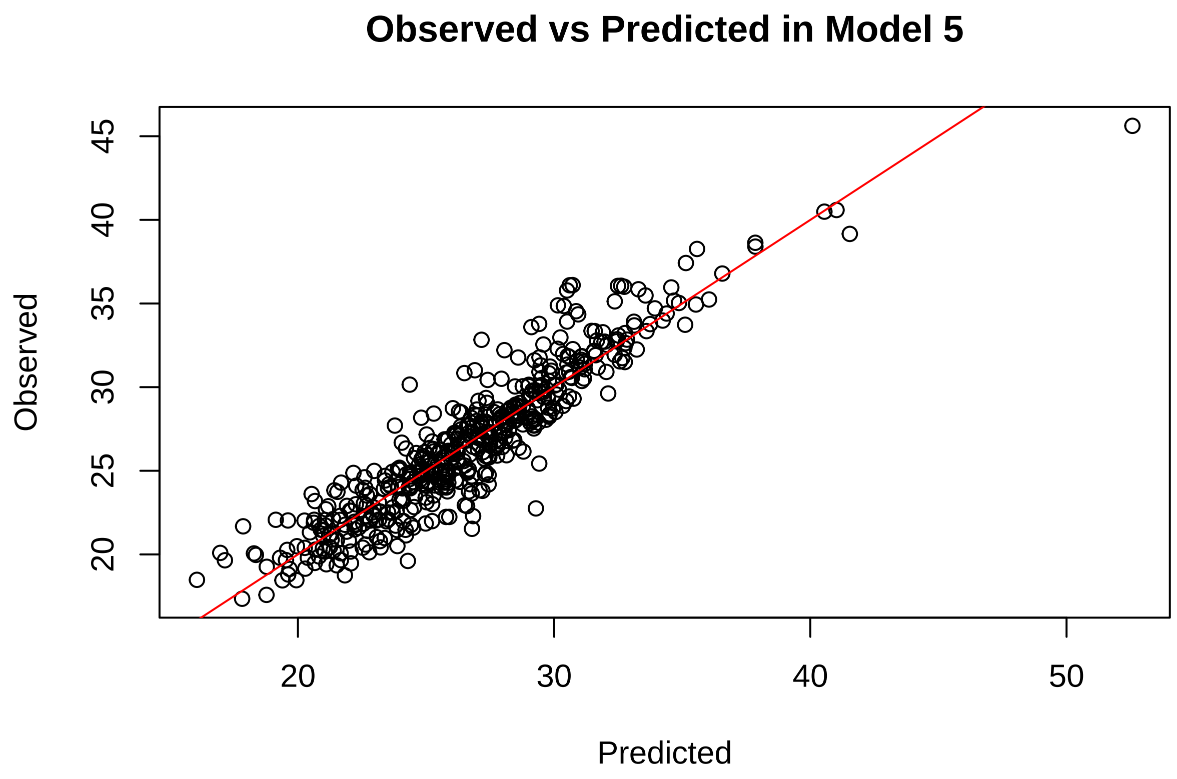
```
# predicted vs observed for model 3
plot(height ~ predict(metasequoia_model3), data = metasequoia, main = "Observed vs Predicted in Model 3
abline(a = 0, b = 1, col = "red")
```



```
# predicted vs observed for model 4
plot(height ~ predict(metasequoia_model4), data = metasequoia, main = "Observed vs Predicted in Model 4
abline(a = 0, b = 1, col = "red")
```



```
# predicted vs observed for model 5
plot(height ~ predict(metasequoia_model5), data = metasequoia, main = "Observed vs Predicted in Model 5
abline(a = 0, b = 1, col = "red")
```



```
# calculating bias
mean((predict(metasequoia_model1) - metasequoia$height) / metasequoia$height) * 100
```

```
## [1] 0.419971
```

```
mean((predict(metasequoia_model2) - metasequoia$height) / metasequoia$height) * 100
```

```
## [1] 0.3725447
```

```
mean((predict(metasequoia_model3) - metasequoia$height) / metasequoia$height) * 100
```

```
## [1] 0.379749
```

```
mean((predict(metasequoia_model4) - metasequoia$height) / metasequoia$height) * 100
```

```
## [1] 0.3978608
```

```
mean((predict(metasequoia_model5) - metasequoia$height) / metasequoia$height) * 100
```

```
## [1] 0.4128386
```

```r
# calculating RMSE
# we want the lowest value which is model 4
rmse(metasequoia_model1) # can also use: rmse(metasequoia$height, predict(metasequoia_model1))
```

```
## [1] 1.66083
```

```r
rmse(metasequoia_model2)
```

```
## [1] 1.739887
```

```r
rmse(metasequoia_model3)
```

```
## [1] 1.635669
```

```r
rmse(metasequoia_model4)
```

```
## [1] 1.631005
```

```r
rmse(metasequoia_model5)
```

```
## [1] 1.70729
```

```r
# calculating AIC
# we want the lowest value which is model 4
AIC(metasequoia_model1)
```

```
## [1] 1932.256
```

```r
AIC(metasequoia_model2)
```

```
## [1] 1978.759
```

```r
AIC(metasequoia_model3)
```

```
## [1] 1918.99
```

```r
AIC(metasequoia_model4)
```

```
## [1] 1916.135
```

```r
AIC(metasequoia_model5)
```

```
## [1] 1961.846
```

```r
# calculating R^2adj
# we want the highest value which is model 4
summary(metasequoia_model1)$adj.r.squared
```

```
## [1] 0.8599043
```

```r
summary(metasequoia_model2)$adj.r.squared
```

```
## [1] 0.8462496
```

```r
summary(metasequoia_model3)$adj.r.squared
```

```
## [1] 0.8638436
```

```r
summary(metasequoia_model4)$adj.r.squared
```

```
## [1] 0.8646189
```

```r
summary(metasequoia_model5)$adj.r.squared
```

```
## [1] 0.8516588
```

```r
# calculating CIs
confint(metasequoia_model1, level = 1-0.05)
```

```
##                 2.5 %     97.5 %
## (Intercept) 9.5906813 10.7977800
## diameter    0.2981895  0.3201372
```

```r
confint(metasequoia_model2, level = 1-0.05)
```

```
##                      2.5 %    97.5 %
## (Intercept)      -41.78667 -36.82948
## I(log(diameter))  16.09408  17.34758
```

```r
confint(metasequoia_model3, level = 1-0.05)
```

```
##                      2.5 %       97.5 %
## (Intercept)    5.969239960  8.952899316
## diameter       0.356681537  0.456664165
## I(diameter^2) -0.001225269 -0.000407843
```

```r
confint(metasequoia_model4, level = 1-0.05) # this one
```

```
##                       2.5 %        97.5 %
## (Intercept)    1.521801e+01  1.628303e+01
## I(diameter^2)  4.946444e-03  5.670146e-03
## I(diameter^3) -3.143211e-05 -2.460142e-05
```

```
confint(metasequoia_model5, level = 1-0.05)
```

```
##                         2.5 %        97.5 %
## (Intercept)     2.909117e+01  3.236874e+01
## I(diameter^-1) -4.698412e+02 -3.535830e+02
## I(diameter^2)   1.197215e-03  1.548137e-03
```

```
#Data processing
sequoia = read.csv("data/metasequoia_data.csv")
sequoia$log.diameter <- log10(sequoia$diameter)
sequoia$squared.diameter <- (sequoia$diameter)^2
sequoia$cubic.diameter <- (sequoia$diameter)^3
sequoia$diameter.to.the.power.of.negativeone <- (sequoia$diameter)^-1

#Model Selection
full.model = lm(height ~ diameter + squared.diameter + cubic.diameter + log.diameter + diameter.to.the.p
empty.model = lm(height ~ 1,data = sequoia)

n = nrow(sequoia)

forward.model.AIC = stepAIC(empty.model, scope = list(lower = empty.model, upper= full.model), k = 2,di
forward.model.BIC = stepAIC(empty.model,  scope = list(lower = empty.model, upper= full.model), k = log
backward.model.AIC = stepAIC(full.model, scope = list(lower = empty.model, upper= full.model), k = 2,di
backward.model.BIC = stepAIC(full.model,  scope = list(lower = empty.model, upper= full.model), k = log
FB.model.AIC = stepAIC(empty.model, scope = list(lower = empty.model, upper= full.model), k = 2,directi
FB.model.BIC = stepAIC(empty.model,  scope = list(lower = empty.model, upper= full.model), k = log(n),t
BF.model.AIC = stepAIC(full.model, scope = list(lower = empty.model, upper= full.model), k = 2,directio
BF.model.BIC = stepAIC(full.model,  scope = list(lower = empty.model, upper= full.model), k = log(n),tr
model4 = lm(height ~  squared.diameter + cubic.diameter, data = sequoia)
```

```
#Calculating AIC
AIC(forward.model.AIC)
```

```
## [1] 1913.133
```

```
AIC(forward.model.BIC)
```

```
## [1] 1916.126
```

```
AIC(backward.model.AIC)
```

```
## [1] 1912.894
```

```
AIC(backward.model.BIC)
```

```
## [1] 1912.894
```

```
AIC(FB.model.AIC)
```

```
## [1] 1913.133
```

```r
AIC(FB.model.BIC)
```

```
## [1] 1916.126
```

```r
AIC(BF.model.AIC)
```

```
## [1] 1912.894
```

```r
AIC(BF.model.BIC)
```

```
## [1] 1912.894
```

```r
AIC(model4)
```

```
## [1] 1916.135
```

```r
#Calculating BIC
BIC(forward.model.AIC)
```

```
## [1] 1934.206
```

```r
BIC(forward.model.BIC)
```

```
## [1] 1932.985
```

```r
BIC(backward.model.AIC)
```

```
## [1] 1933.967
```

```r
BIC(backward.model.BIC)
```

```
## [1] 1933.967
```

```r
BIC(FB.model.AIC)
```

```
## [1] 1934.206
```

```r
BIC(FB.model.BIC)
```

```
## [1] 1932.985
```

```r
BIC(BF.model.AIC)
```

```
## [1] 1933.967
```

```r
BIC(BF.model.BIC)
```

```
## [1] 1933.967
```

```r
BIC(model4)
```

```
## [1] 1932.994
```

```r
#New Best Models
best.AIC.model = backward.model.AIC
best.BIC.model = forward.model.BIC
model4 = lm(height ~  squared.diameter + cubic.diameter, data = sequoia)
summary(best.AIC.model)
```

```
##
## Call:
## lm(formula = height ~ diameter + squared.diameter + log.diameter,
##     data = sequoia)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.7223 -1.0474 -0.0561  0.9061  6.1886
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.595e+01  1.355e+01   3.392 0.000750 ***
## diameter          8.957e-01  1.737e-01   5.157 3.64e-07 ***
## squared.diameter -2.655e-03  6.783e-04  -3.914 0.000103 ***
## log.diameter     -3.443e+01  1.210e+01  -2.846 0.004617 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.629 on 496 degrees of freedom
## Multiple R-squared:  0.8666, Adjusted R-squared:  0.8658
## F-statistic:  1074 on 3 and 496 DF,  p-value: < 2.2e-16
```

```r
summary(best.BIC.model)
```

```
##
## Call:
## lm(formula = height ~ diameter + cubic.diameter, data = sequoia)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.7082 -1.0073 -0.0244  0.8773  6.0630
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     8.406e+00  5.152e-01  16.318  < 2e-16 ***
## diameter        3.567e-01  1.237e-02  28.823  < 2e-16 ***
## cubic.diameter -4.091e-06  9.549e-07  -4.284 2.21e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.636 on 497 degrees of freedom
## Multiple R-squared:  0.8652, Adjusted R-squared:  0.8646
## F-statistic:  1594 on 2 and 497 DF,  p-value: < 2.2e-16
```

```
summary(model4)
```

```
##
## Call:
## lm(formula = height ~ squared.diameter + cubic.diameter, data = sequoia)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.6346 -1.0426 -0.1073  0.8784  6.2001
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.575e+01  2.710e-01   58.11   <2e-16 ***
## squared.diameter  5.308e-03  1.842e-04   28.82   <2e-16 ***
## cubic.diameter   -2.802e-05  1.738e-06  -16.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.636 on 497 degrees of freedom
## Multiple R-squared:  0.8652, Adjusted R-squared:  0.8646
## F-statistic:  1594 on 2 and 497 DF,  p-value: < 2.2e-16
```

```
sequoia$ei = best.AIC.model$residuals
sequoia$yhat = best.AIC.model$fitted.values

ei = best.AIC.model$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.9798, p-value = 2.051e-06
```

```
Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  sequoia$ei and sequoia$Group
## Fligner-Killeen:med chi-squared = 0.075917, df = 1, p-value = 0.7829
```

```r
#B
sequoia$ei = best.BIC.model$residuals
sequoia$yhat = best.BIC.model$fitted.values

ei = best.BIC.model$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.97961, p-value = 1.835e-06
```

```r
Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  sequoia$ei and sequoia$Group
## Fligner-Killeen:med chi-squared = 0.64289, df = 1, p-value = 0.4227
```

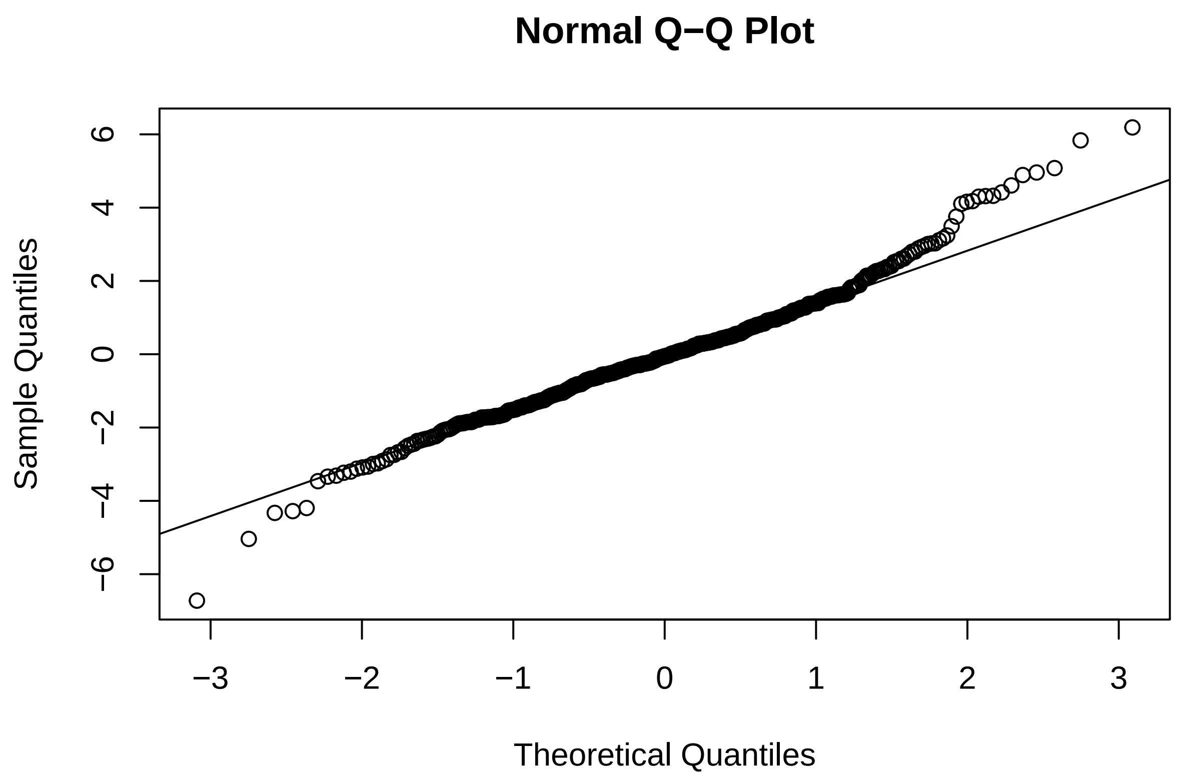
```r
#C
sequoia$ei = model4$residuals
sequoia$yhat = model4$fitted.values

ei = model4$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.98038, p-value = 2.871e-06
```

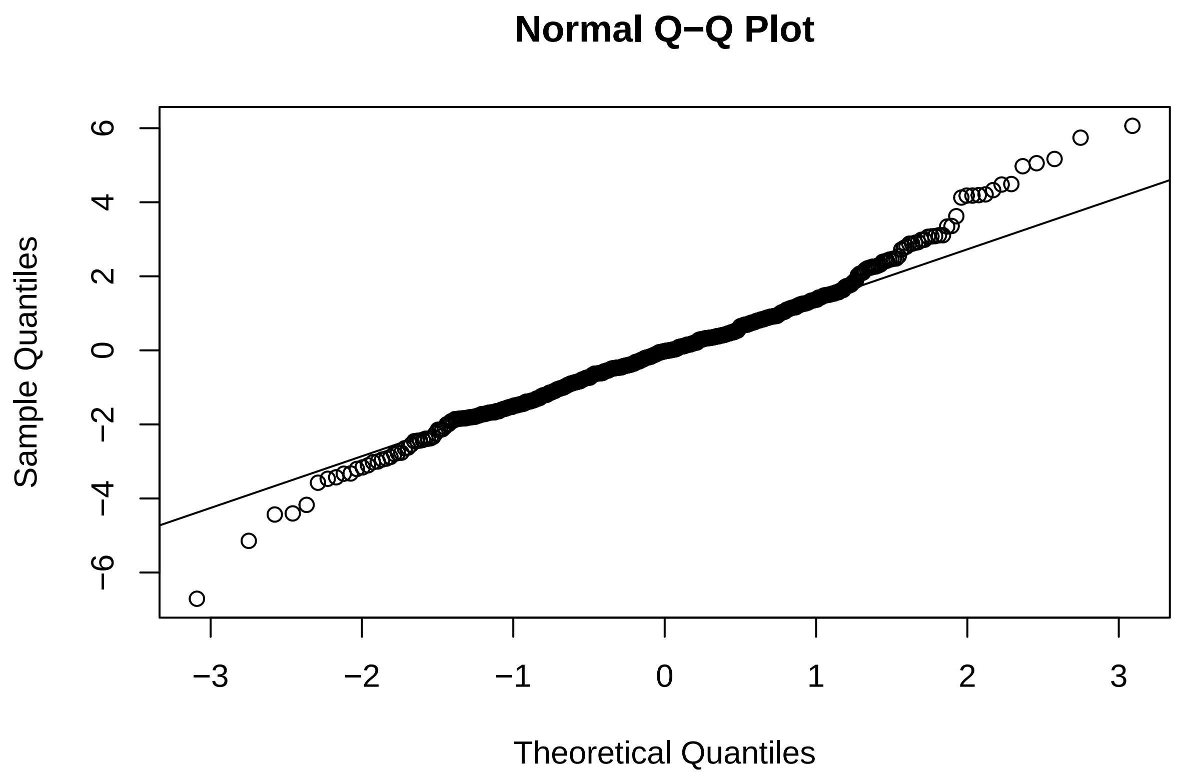
```r
Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  sequoia$ei and sequoia$Group
## Fligner-Killeen:med chi-squared = 0.15298, df = 1, p-value = 0.6957
```

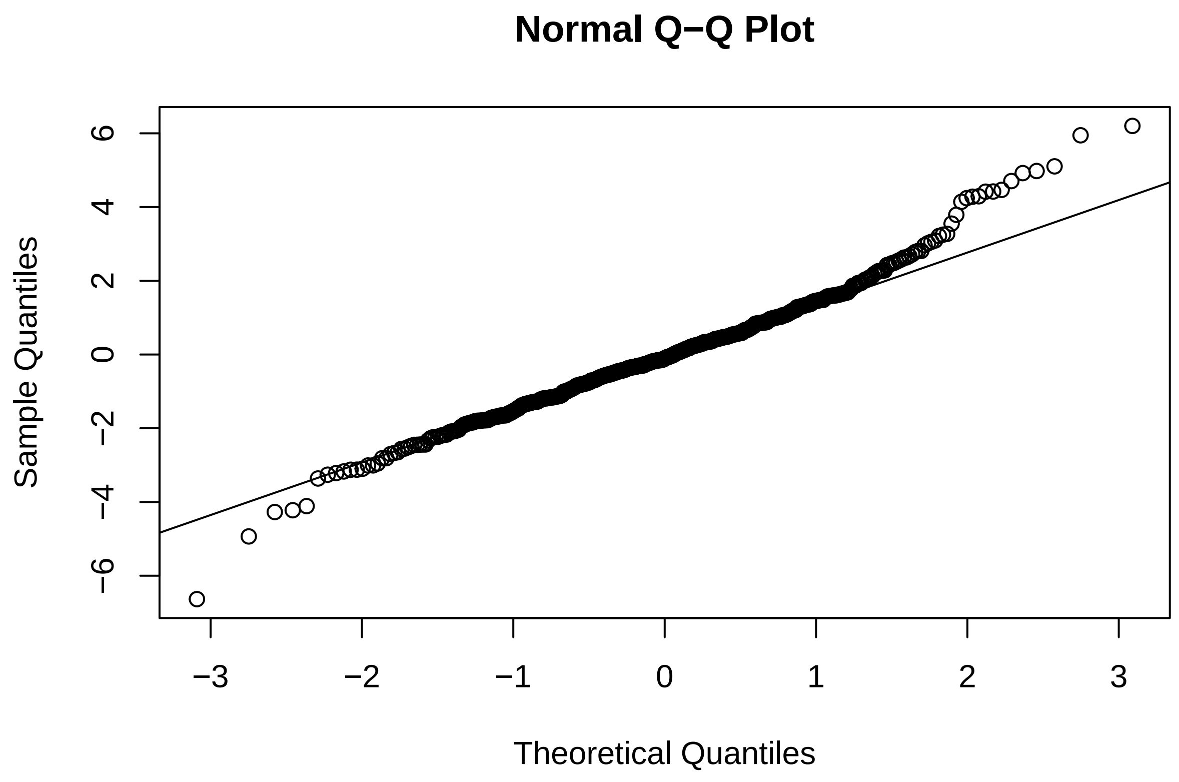
```r
qqnorm(best.AIC.model$residuals)
qqline(best.AIC.model$residuals)
```



```r
qqnorm(best.BIC.model$residuals)
qqline(best.BIC.model$residuals)
```



```
qqnorm(model4$residuals)
qqline(model4$residuals)
```



```
#Removing Outliers
sequoia$residuals = residuals(best.AIC.model)
sequoia$std_residuals = rstandard(best.AIC.model)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data1 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]

sequoia$residuals = residuals(best.BIC.model)
sequoia$std_residuals = rstandard(best.BIC.model)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data2 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]

sequoia$residuals = residuals(model4)
sequoia$std_residuals = rstandard(model4)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data3 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]
```

```r
#Re-model using the new dataset
best.AIC.model$coefficients
```

```
##      (Intercept)         diameter squared.diameter     log.diameter
##     45.947875497      0.895679116     -0.002655002    -34.429255267
```

```r
best.BIC.model$coefficients
```

```
##    (Intercept)       diameter cubic.diameter
##   8.406212e+00   3.566710e-01  -4.090712e-06
```

```r
model4$coefficients
```

```
##      (Intercept) squared.diameter   cubic.diameter
##     1.575052e+01     5.308295e-03    -2.801676e-05
```

```r
model.a = lm(height ~ diameter + squared.diameter + log.diameter, data = new.data1)
model.b = lm(height ~ diameter + cubic.diameter, data = new.data2)
model.c = lm(height ~ squared.diameter + cubic.diameter, data = new.data3)
```

```r
#SW Test
#A
new.data1$ei = model.a$residuals
new.data1$yhat = model.a$fitted.values

ei = model.a$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.99488, p-value = 0.1147
```

```r
#B
new.data2$ei = model.b$residuals
new.data2$yhat = model.b$fitted.values

ei = model.b$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.99404, p-value = 0.06006
```

```
#C
new.data3$ei = model.c$residuals
new.data3$yhat = model.c$fitted.values

ei = model.c$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ei
## W = 0.99566, p-value = 0.2097
```

```
#FK Test
#A
Group = rep("Lower",nrow(new.data1))
Group[new.data1$height < median(new.data1$height)] = "Upper"
Group = as.factor(Group)
new.data1$Group = Group
the.FKtest= fligner.test(new.data1$ei, new.data1$Group)
the.FKtest
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  new.data1$ei and new.data1$Group
## Fligner-Killeen:med chi-squared = 1.5267, df = 1, p-value = 0.2166
```

```
#B
Group = rep("Lower",nrow(new.data2))
Group[new.data2$height < median(new.data2$height)] = "Upper"
Group = as.factor(Group)
new.data2$Group = Group
the.FKtest= fligner.test(new.data2$ei, new.data2$Group)
the.FKtest
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  new.data2$ei and new.data2$Group
## Fligner-Killeen:med chi-squared = 4.1384, df = 1, p-value = 0.04192
```

```
#C
Group = rep("Lower",nrow(new.data3))
Group[new.data3$height < median(new.data3$height)] = "Upper"
Group = as.factor(Group)
new.data3$Group = Group
the.FKtest= fligner.test(new.data3$ei, new.data3$Group)
the.FKtest
```

```
##
```

```
##  Fligner-Killeen test of homogeneity of variances
##
## data:  new.data3$ei and new.data3$Group
## Fligner-Killeen:med chi-squared = 1.9159, df = 1, p-value = 0.1663
```

```
#Quality test of new models
AIC(model.a)
```

```
## [1] 1629.31
```

```
AIC(model.b)
```

```
## [1] 1608.176
```

```
AIC(model.c)
```

```
## [1] 1632.427
```

```
BIC(model.a)
```

```
## [1] 1650.148
```

```
BIC(model.b)
```

```
## [1] 1624.821
```

```
BIC(model.c)
```

```
## [1] 1649.098
```

```
rmse(model.a)
```

```
## [1] 1.321083
```

```
rmse(model.b)
```

```
## [1] 1.30867
```

```
rmse(model.c)
```

```
## [1] 1.328188
```

```
summary(model.a)$adj.r.squared
```

```
## [1] 0.9077727
```

```
summary(model.b)$adj.r.squared
```

```
## [1] 0.9093569
```

```
summary(model.c)$adj.r.squared
```

```
## [1] 0.9032784
```

```
model.a$coefficients
```

```
##     (Intercept)        diameter squared.diameter     log.diameter
##    44.579952369     0.863216948     -0.002501801    -32.932998516
```

```
model.b$coefficients
```

```
##   (Intercept)      diameter cubic.diameter
##  8.659310e+00  3.496200e-01  -3.762588e-06
```

```
model.c$coefficients
```

```
##     (Intercept) squared.diameter   cubic.diameter
##    1.535264e+01     5.668494e-03    -3.238495e-05
```

```
alpha = 0.05
the.CIs = confint(model.b,level = 1-alpha)
the.CIs
```

```
##                        2.5 %        97.5 %
## (Intercept)     7.840136e+00  9.478484e+00
## diameter        3.299136e-01  3.693264e-01
## cubic.diameter -5.275971e-06 -2.249205e-06
```

```
test.stuff = summary(model.b)$coefficients
summary(model.b)$coefficients
```

```
##                     Estimate   Std. Error   t value      Pr(>|t|)
## (Intercept)     8.659310e+00 4.168796e-01 20.771728  1.648354e-68
## diameter        3.496200e-01 1.002864e-02 34.862165 1.530932e-132
## cubic.diameter -3.762588e-06 7.701642e-07 -4.885436  1.416719e-06
```

```
model.b
```

```
##
## Call:
## lm(formula = height ~ diameter + cubic.diameter, data = new.data2)
##
## Coefficients:
##    (Intercept)        diameter  cubic.diameter
##      8.659e+00       3.496e-01      -3.763e-06
```

Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
# reading libaries
library(tidyverse)
library(ModelMetrics)
library(MASS)
# reading data
# note: data was obtained through a given docx, which I made into a google doc, then copy pasted to goo
# note: the data we were given is about 10% of the data they used, so our graphs will look slightly dif
metasequoia <- read_csv("data/metasequoia_data.csv")
# data exploration
metasequoia %>%
  pivot_longer(col = c("height", "diameter"),
               names_to = "datatype",
               values_to = "values") %>%
  group_by(datatype) %>%
  summarise(mean = mean(values),
            max = max(values),
            min = min(values),
            sd = sd(values)) %>%
  t()
# models
metasequoia_model1 <- lm(height ~ diameter, data = metasequoia)
metasequoia_model2 <- lm(height ~ I(log(diameter)), data = metasequoia)
metasequoia_model3 <- lm(height ~ diameter + I(diameter^2), data = metasequoia)
metasequoia_model4 <- lm(height ~ I(diameter^2) + I(diameter^3), data = metasequoia)
metasequoia_model5 <- lm(height ~ I(diameter^-1) + I(diameter^2), data = metasequoia)
# about non-linear models: not sure how to do it and this code is broken
# metasequoia_model8 <- nls(height ~ 1.3 + a1 * (1 - exp(-a1 * diameter))^a2, data = metasequoia, start
# Fig 2. Scatter diagram of the tree height and dbh of a single Metasequoia tree.
plot(height ~ diameter, data = metasequoia, main = "Scatterplot of Height and Diameter", xlab = "Diamete
abline(a = 12.546, b = 0.264) # the paper's data's trendline
abline(metasequoia_model1, col = "red") # trendline for model 1
#par(mfrow = c(2, 3))
# making residuals plot for model 1
plot(resid(metasequoia_model1) ~ predict(metasequoia_model1), main = "Residual Plot for Model 1", xlab =
abline(h = 0,col = "red",lty = 2)
# making residuals plot for model 2
plot(resid(metasequoia_model2) ~ predict(metasequoia_model2), main = "Residual Plot for Model 2", xlab =
abline(h = 0,col = "red",lty = 2)
# making residuals plot for model 3
plot(resid(metasequoia_model3) ~ predict(metasequoia_model3), main = "Residual Plot for Model 3", xlab =
abline(h = 0,col = "red",lty = 2)
# making residuals plot for model 4
plot(resid(metasequoia_model4) ~ predict(metasequoia_model4), main = "Residual Plot for Model 4", xlab =
abline(h = 0,col = "red",lty = 2)
# making residuals plot for model 5
plot(resid(metasequoia_model5) ~ predict(metasequoia_model5), main = "Residual Plot for Model 5", xlab =
abline(h = 0,col = "red",lty = 2)
#par(mfrow = c(2, 3))
# making qq plot for model 1
qqnorm(resid(metasequoia_model1), main = "Q-Q Plot for Model 1", col = "red")
```

```r
qqline(resid(metasequoia_model1))
# making qq plot for model 2
qqnorm(resid(metasequoia_model2), main = "Q-Q Plot for Model 2", col = "red")
qqline(resid(metasequoia_model2))
# making qq plot for model 3
qqnorm(resid(metasequoia_model3), main = "Q-Q Plot for Model 3", col = "red")
qqline(resid(metasequoia_model3))
# making qq plot for model 4
qqnorm(resid(metasequoia_model4), main = "Q-Q Plot for Model 4", col = "red")
qqline(resid(metasequoia_model4))
# making qq plot for model 5
qqnorm(resid(metasequoia_model5), main = "Q-Q Plot for Model 5", col = "red")
qqline(resid(metasequoia_model5))
#par(mfrow = c(2, 3))
# predicted vs observed for model 1
plot(height ~ predict(metasequoia_model1), data = metasequoia, main = "Observed vs Predicted in Model 1
abline(a = 0, b = 1, col = "red")
# predicted vs observed for model 2
plot(height ~ predict(metasequoia_model2), data = metasequoia, main = "Observed vs Predicted in Model 2
abline(a = 0, b = 1, col = "red")
# predicted vs observed for model 3
plot(height ~ predict(metasequoia_model3), data = metasequoia, main = "Observed vs Predicted in Model 3
abline(a = 0, b = 1, col = "red")
# predicted vs observed for model 4
plot(height ~ predict(metasequoia_model4), data = metasequoia, main = "Observed vs Predicted in Model 4
abline(a = 0, b = 1, col = "red")
# predicted vs observed for model 5
plot(height ~ predict(metasequoia_model5), data = metasequoia, main = "Observed vs Predicted in Model 5
abline(a = 0, b = 1, col = "red")
# calculating bias
mean((predict(metasequoia_model1) - metasequoia$height) / metasequoia$height) * 100
mean((predict(metasequoia_model2) - metasequoia$height) / metasequoia$height) * 100
mean((predict(metasequoia_model3) - metasequoia$height) / metasequoia$height) * 100
mean((predict(metasequoia_model4) - metasequoia$height) / metasequoia$height) * 100
mean((predict(metasequoia_model5) - metasequoia$height) / metasequoia$height) * 100
# calculating RMSE
# we want the lowest value which is model 4
rmse(metasequoia_model1) # can also use: rmse(metasequoia$height, predict(metasequoia_model1))
rmse(metasequoia_model2)
rmse(metasequoia_model3)
rmse(metasequoia_model4)
rmse(metasequoia_model5)
# calculating AIC
# we want the lowest value which is model 4
AIC(metasequoia_model1)
AIC(metasequoia_model2)
AIC(metasequoia_model3)
AIC(metasequoia_model4)
AIC(metasequoia_model5)
# calculating R^2adj
# we want the highest value which is model 4
summary(metasequoia_model1)$adj.r.squared
summary(metasequoia_model2)$adj.r.squared
summary(metasequoia_model3)$adj.r.squared
summary(metasequoia_model4)$adj.r.squared
summary(metasequoia_model5)$adj.r.squared
# calculating CIs
confint(metasequoia_model1, level = 1-0.05)
confint(metasequoia_model2, level = 1-0.05)
confint(metasequoia_model3, level = 1-0.05)
confint(metasequoia_model4, level = 1-0.05) # this one
confint(metasequoia_model5, level = 1-0.05)
#Data processing
sequoia = read.csv("data/metasequoia_data.csv")
sequoia$log.diameter <- log10(sequoia$diameter)
sequoia$squared.diameter <- (sequoia$diameter)^2
sequoia$cubic.diameter <- (sequoia$diameter)^3
sequoia$diameter.to.the.power.of.negativeone <- (sequoia$diameter)^-1

#Model Selection
full.model = lm(height ~ diameter + squared.diameter + cubic.diameter + log.diameter + diameter.to.the.p
empty.model = lm(height ~ 1,data = sequoia)

n = nrow(sequoia)

forward.model.AIC = stepAIC(empty.model, scope = list(lower = empty.model, upper= full.model), k = 2,di
forward.model.BIC = stepAIC(empty.model,  scope = list(lower = empty.model, upper= full.model), k = log
backward.model.AIC = stepAIC(full.model, scope = list(lower = empty.model, upper= full.model), k = 2,di
backward.model.BIC = stepAIC(full.model,  scope = list(lower = empty.model, upper= full.model), k = log
FB.model.AIC = stepAIC(empty.model, scope = list(lower = empty.model, upper= full.model), k = 2,directi
FB.model.BIC = stepAIC(empty.model,  scope = list(lower = empty.model, upper= full.model), k = log(n),t
BF.model.AIC = stepAIC(full.model, scope = list(lower = empty.model, upper= full.model), k = 2,directio
BF.model.BIC = stepAIC(full.model,  scope = list(lower = empty.model, upper= full.model), k = log(n),tr
model4 = lm(height ~  squared.diameter + cubic.diameter, data = sequoia)
#Calculating AIC
AIC(forward.model.AIC)
AIC(forward.model.BIC)
AIC(backward.model.AIC)
AIC(backward.model.BIC)
AIC(FB.model.AIC)
AIC(FB.model.BIC)
AIC(BF.model.AIC)
AIC(BF.model.BIC)
AIC(model4)
#Calculating BIC
BIC(forward.model.AIC)
BIC(forward.model.BIC)
BIC(backward.model.AIC)
BIC(backward.model.BIC)
BIC(FB.model.AIC)
BIC(FB.model.BIC)
BIC(BF.model.AIC)
BIC(BF.model.BIC)
BIC(model4)
#New Best Models
best.AIC.model = backward.model.AIC
best.BIC.model = forward.model.BIC
model4 = lm(height ~  squared.diameter + cubic.diameter, data = sequoia)
summary(best.AIC.model)
summary(best.BIC.model)
summary(model4)
sequoia$ei = best.AIC.model$residuals
sequoia$yhat = best.AIC.model$fitted.values

ei = best.AIC.model$residuals
the.SWtest = shapiro.test(ei)
the.SWtest

Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest

#B
sequoia$ei = best.BIC.model$residuals
sequoia$yhat = best.BIC.model$fitted.values

ei = best.BIC.model$residuals
the.SWtest = shapiro.test(ei)
the.SWtest

Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest

#C
sequoia$ei = model4$residuals
sequoia$yhat = model4$fitted.values

ei = model4$residuals
the.SWtest = shapiro.test(ei)
the.SWtest

Group = rep("Lower",nrow(sequoia))
Group[sequoia$height < median(sequoia$height)] = "Upper"
Group = as.factor(Group)
sequoia$Group = Group
the.FKtest= fligner.test(sequoia$ei, sequoia$Group)
the.FKtest

qqnorm(best.AIC.model$residuals)
qqline(best.AIC.model$residuals)

qqnorm(best.BIC.model$residuals)
```

```r
qqline(best.BIC.model$residuals)

qqnorm(model4$residuals)
qqline(model4$residuals)
#Removing Outliers
sequoia$residuals = residuals(best.AIC.model)
sequoia$std_residuals = rstandard(best.AIC.model)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data1 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]

sequoia$residuals = residuals(best.BIC.model)
sequoia$std_residuals = rstandard(best.BIC.model)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data2 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]

sequoia$residuals = residuals(model4)
sequoia$std_residuals = rstandard(model4)

threshold = 2
outliers = sequoia[abs(sequoia$std_residuals) > threshold, ]

new.data3 <- sequoia[abs(sequoia$std_residuals) <= threshold, ]
#Re-model using the new dataset
best.AIC.model$coefficients
best.BIC.model$coefficients
model4$coefficients

model.a = lm(height ~ diameter + squared.diameter + log.diameter, data = new.data1)
model.b = lm(height ~ diameter + cubic.diameter, data = new.data2)
model.c = lm(height ~ squared.diameter + cubic.diameter, data = new.data3)
#SW Test
#A
new.data1$ei = model.a$residuals
new.data1$yhat = model.a$fitted.values

ei = model.a$residuals
the.SWtest = shapiro.test(ei)
the.SWtest

#B
new.data2$ei = model.b$residuals
new.data2$yhat = model.b$fitted.values

ei = model.b$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
```

```r
#C
new.data3$ei = model.c$residuals
new.data3$yhat = model.c$fitted.values

ei = model.c$residuals
the.SWtest = shapiro.test(ei)
the.SWtest
#FK Test
#A
Group = rep("Lower",nrow(new.data1))
Group[new.data1$height < median(new.data1$height)] = "Upper"
Group = as.factor(Group)
new.data1$Group = Group
the.FKtest= fligner.test(new.data1$ei, new.data1$Group)
the.FKtest

#B
Group = rep("Lower",nrow(new.data2))
Group[new.data2$height < median(new.data2$height)] = "Upper"
Group = as.factor(Group)
new.data2$Group = Group
the.FKtest= fligner.test(new.data2$ei, new.data2$Group)
the.FKtest

#C
Group = rep("Lower",nrow(new.data3))
Group[new.data3$height < median(new.data3$height)] = "Upper"
Group = as.factor(Group)
new.data3$Group = Group
the.FKtest= fligner.test(new.data3$ei, new.data3$Group)
the.FKtest
#Quality test of new models
AIC(model.a)
AIC(model.b)
AIC(model.c)

BIC(model.a)
BIC(model.b)
BIC(model.c)

rmse(model.a)
rmse(model.b)
rmse(model.c)

summary(model.a)$adj.r.squared
summary(model.b)$adj.r.squared
summary(model.c)$adj.r.squared
model.a$coefficients
model.b$coefficients
model.c$coefficients
alpha = 0.05
the.CIs = confint(model.b,level = 1-alpha)
the.CIs
```

```r
test.stuff = summary(model.b)$coefficients
summary(model.b)$coefficients
model.b
```