Regression explained

sameh magdeldin

2025-01-07

Regression analysis

Module contents

- Simple linear regression
- Global validation of linear model assumption
- Multiple linear regression
- Testing outliers and dropping values
- Nonlinear regression
- Quality check of fitted model

Packages needed in this tutorial

```
library("MASS")  # this datasets is found in MASS package or u can load it
#directly
library("ggplot2") # for plotting
library("car") # for some graphical and analytical functions & VIF calculation
library("PerformanceAnalytics")# we will need this package for correlation later
library("GGally")# we will need this package for correlation later
library("plyr") #data manipulation
library("corrplot") # for correlation matrix viewing
library("dplyr") #data manipulation "should be installed after plyr"
library("RColorBrewer") #this is to give a nice coloring for ur figures
library("gvlma") #validation of regression model
library("broom")# compare models
library("dplyr")# data wrangling
library("lmtest") # Breusch-Pagan test
```

What is Regression?

Regression is a statistical method used to model the relationship between a dependent variable (also called the response or outcome variable) and one or more independent variables (also called predictors or explanatory variables).

The goal is to understand how the dependent variable changes as the independent variables vary.

Uses of Regression

** 1. Prediction: **

Regression models can predict the value of the dependent variable for new observations based on the independent variables.

Example: Predicting risk stratification for a disease, clinical prognosis (survival rate, disease progression), evaluating treatment (drug efficiency) developing a prognostic or diagnositic model (check our article on Ewing sarcoma; ezzeldin et al.,)

** 2. Understanding Relationships: **

Regression helps quantify the strength and direction of relationships between variables.

Examples: Understanding how estrogen receptor expression affects breast cancer.

** 3. Hypothesis Testing: **

Regression can test hypotheses about the significance of predictors.

Example: Testing whether education level significantly impacts income.

** 4. Control for Confounding Variables:**

Regression can isolate the effect of one variable while controlling for others.

Example: A researcher is studying the relationship between physical activity (exposure) and risk of cardiovascular disease (outcome). However, the researcher suspects that age and smoking status are confounding variables that could influence both physical activity and cardiovascular disease risk. To control for these confounders, the researcher uses multiple linear regression.

Simple linear regression

This is useful for modelling the relationship between a dependent variable (y) and independent vairable (x).

Simple linear regression can be done using lm(function). It predicts a (dependant variable; DV) based on another independant variable (predictor).

In the formula, it should be like this $\boldsymbol{DV \sim IV}$

some important notes

1. your data should be checked first by correlation, it should be correlated in order to perform a linear regression.
2. check the outliers (as ealier). your data should not have outliers,it will reduce the regression fitting.

lets see some examples

```
(head(cats))#make sure your data is loaded
```

```
##   Sex Bwt Hwt
## 1   F 2.0 7.0
## 2   F 2.0 7.4
## 3   F 2.0 9.5
## 4   F 2.1 7.2
## 5   F 2.1 7.3
## 6   F 2.1 7.6
```

We saw ealier in correlation session that body weight of the cats and their heart weights are correlated

Can we predict the heart weight form their weights? YES, lets do linear regression

```
attach(cats)
lm(formula = cats$Hwt ~ cats$Bwt) # im fitting a model using heart weight as
```

```
##
## Call:
## lm(formula = cats$Hwt ~ cats$Bwt)
##
## Coefficients:
## (Intercept)     cats$Bwt
##     -0.3567       4.0341
```

```
# dependent variable and body weight as a predictor
```

as you can see from the output, the fomula can be constructed as follows

Hwt = 4.0341 (Bwt) - 0.3567 -0.3 is the intercept (equal y when x is zero) and 4 is the slope (steepness)

lets store the output

```
lr <- lm(formula = Hwt ~ Bwt)# name it as you want
```

get the summary

```
summary(lr)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5694 -0.9634 -0.0921  1.0426  5.1238
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.3567     0.6923  -0.515    0.607
## Bwt           4.0341     0.2503  16.119   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.452 on 142 degrees of freedom
## Multiple R-squared:  0.6466, Adjusted R-squared:  0.6441
## F-statistic: 259.8 on 1 and 142 DF,  p-value: < 2.2e-16
```

as you see, its significant (p<2.2e-16) R- squared tells you how close the data are to the fitted regression line, in another word: it is the percentage of the response variable variation that is explained by a linear model which is 0.6441 in this analysis (64%).

NOTE: if your R-squared value is low but you have statistically significant predictors, you can still draw important conclusions about how changes in the predictor values are associated with changes in the response value. Regardless of the R-squared

Residual SE: gives an idea how far heart weight (y) values are from predicted (fitted) heart weight (y hats). again it gives an idea about typical residual error.

In another word, it is a measure used in regression analysis to quantify the difference between observed values and the values predicted by the model. It provides an estimate of the standard deviation of the residuals (errors), which are the differences between the actual and predicted values. RSE is a key metric for assessing the accuracy of a regression model.

Intercept (-0.3) is the estimate of mean heart weight for a cat with body weight 0 [does not make sense!!, not meaningful]

slope for Bwt (4.0341), this is the effect of body weight on heart weight. In another words; we can associate the increase of 1 unit of body weight with an increase of 4.03 unit of heart weight

Note that the term regression coefficients means (slope and intercept)

One more thing, to cereate a confidence intervals for our model

But wait !, what is confidence interval in the context of regression model?

A confidence interval in the context of a linear regression model provides a range of values within which a population parameter (such as the slope or intercept of the regression line) is expected to lie, with a certain level of confidence (e.g., 95%). It quantifies the uncertainty associated with the estimated coefficients in the regression model.

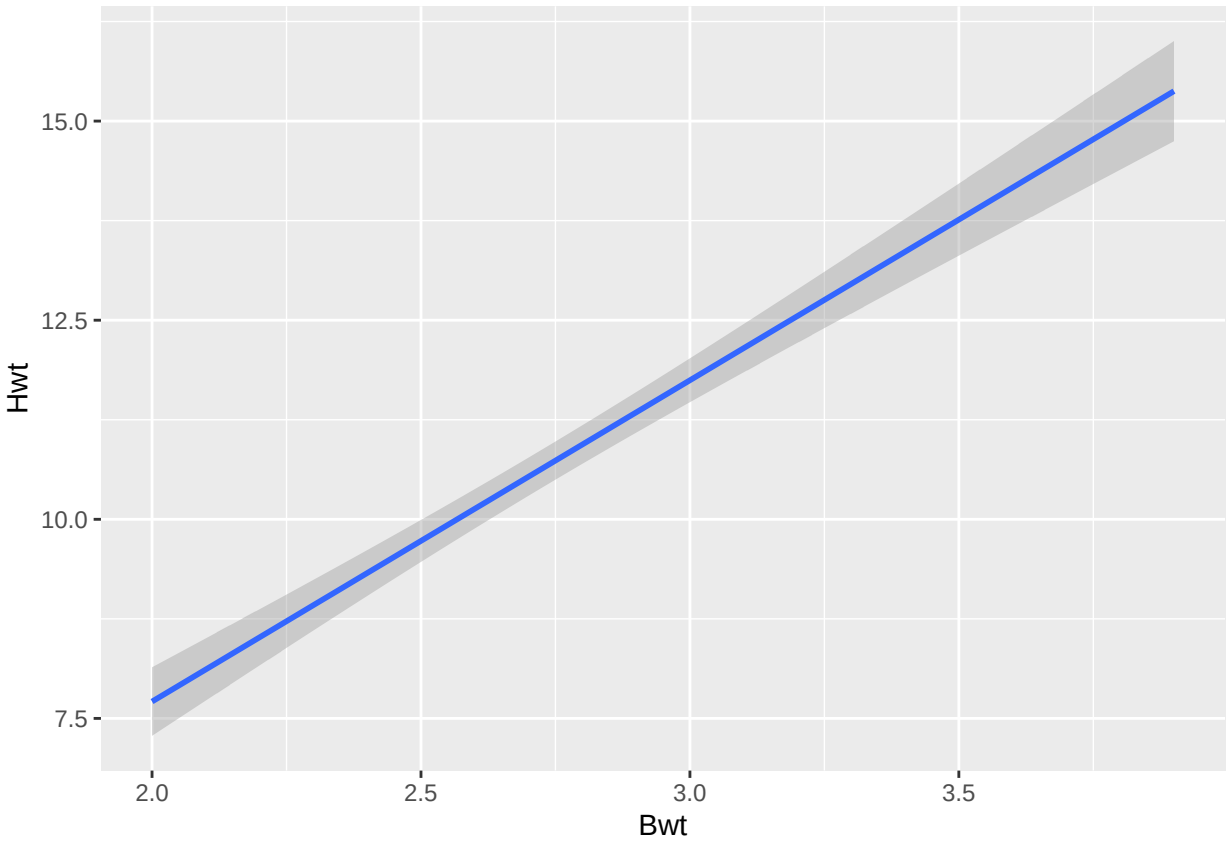
we usually add the confidence interval to the model in order to provide

a range to quantify uncertinity and sample variablility and comparing models

to draw the graph (as earlier)

```
ggplot(cats, aes(x=Bwt, y=Hwt)) + geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

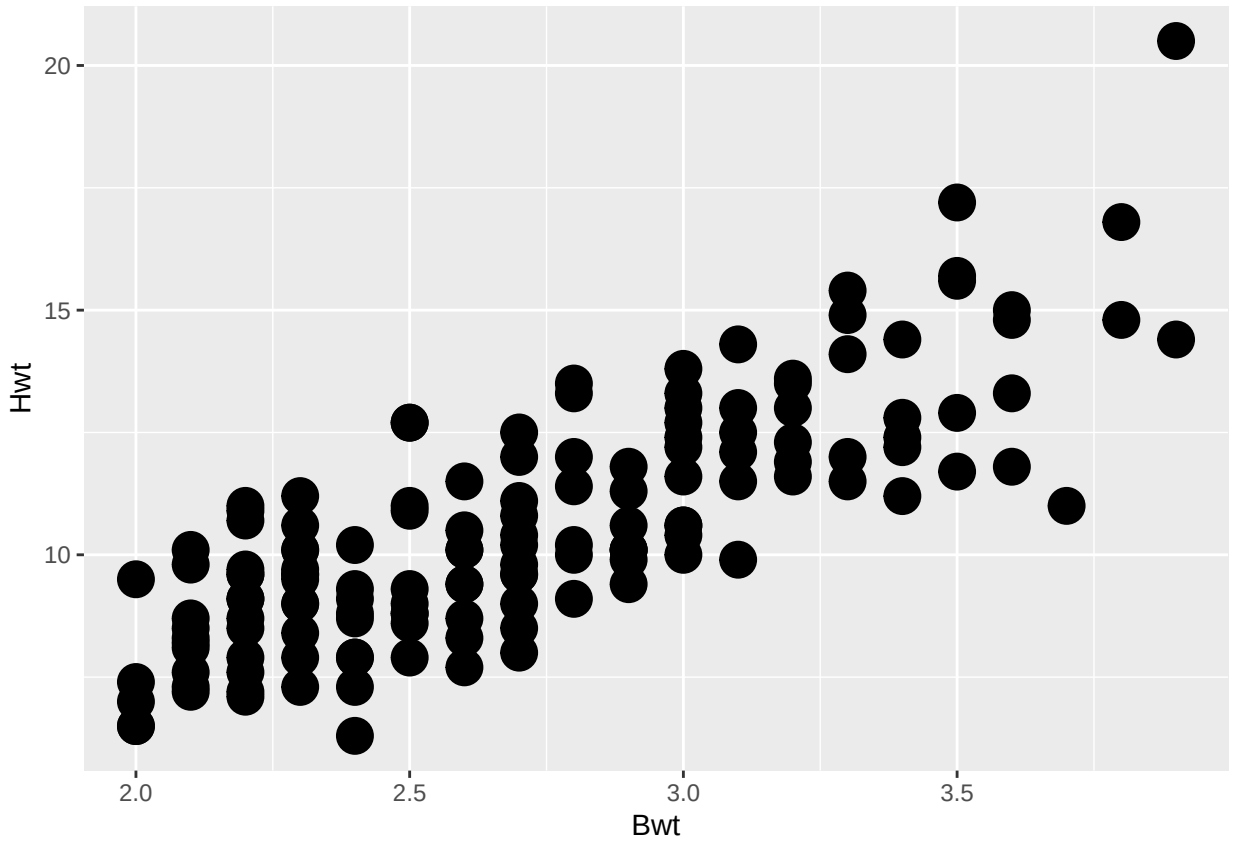


adding +geom_smooth(method=lm) will add a linear model with default 95% confidience region

adding +geom_smooth(method=lm,level=0.99) will change conf region to 99% adding + geom_smooth(method=lm,se=FALS will remove conf region (method=lm,color="black") will change color of fit line to black

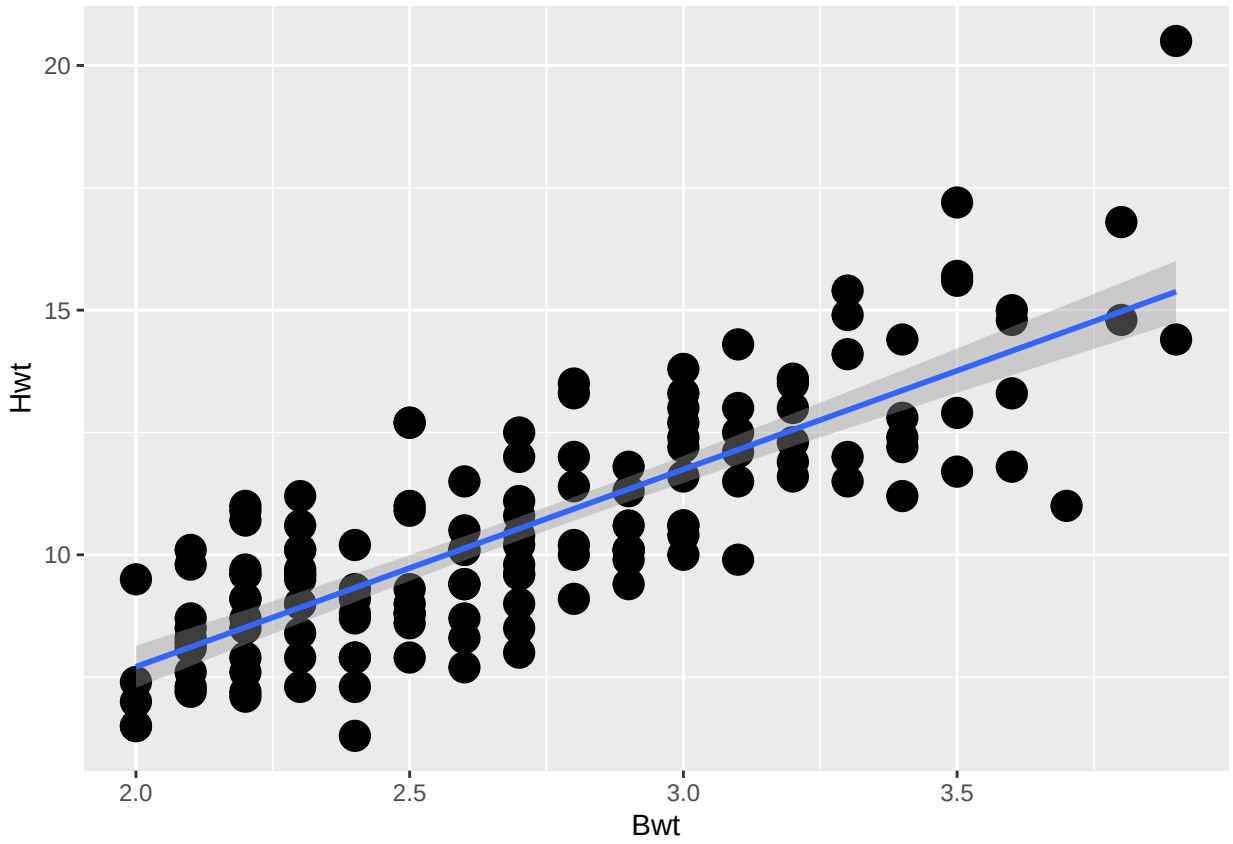
ggplot is a grammer package for graphics in R. check the first formula we used to draw the scatter plot #1,then i will add the regression line and confidience interval#2, then i will add the text#3

```
ggplot(cats, aes(x=Bwt, y=Hwt)) +
geom_point(size=6) #1
```



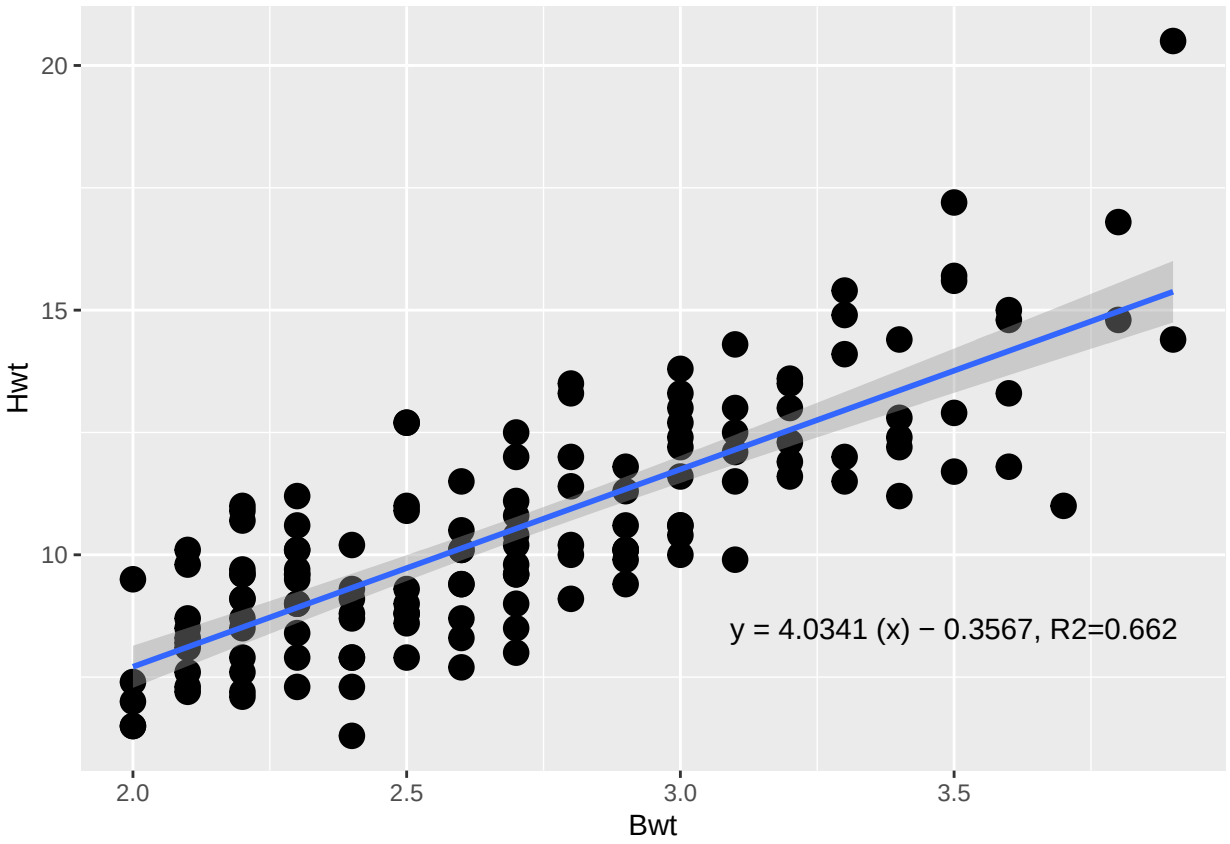

```
ggplot(cats, aes(x=Bwt, y=Hwt)) +
geom_point(size=5)+ geom_smooth(method=lm) #2
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
ggplot(cats, aes(x=Bwt, y=Hwt)) +
        geom_point(size=4)+ geom_smooth(method=lm)+
        annotate("text",x=3.5,y=8.5,label="y = 4.0341 (x) - 0.3567, R2=0.662")#3
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Tutors note

What consideration i have to peer in mind when applying a regression model?

1. Research question and model selection
2. Data quality(missing values, ouliers, data transformation) NB. apply transformation if ur data is skeewed
3. Assumptions
    a) Linearity: In case of linear regression, eensure the relationship between predictors and the outcome is linear (use scatterplots or residual plots).
    b) Independence: Observations should be independent (No relationship between observations)
    c) Homoscedasticity: The variance of residuals should be constant
    across predicted values (check residual plots). [will be discussed later].
4. Multicolinearity: [will be discussed later]

Other consideration will come later Now, lets move to multiple linear regression

Mulitiple linear regression

This is useful for modelling the relationship between a dependent variable (y) and multiple independent vairables (x).

I will be working on the Lung capacity data lets import it and attach it

```
LungCapData <-read.csv("C:/Users/magde/Desktop/datasets for R/LungCapData.csv")
```

have a look on the dataset

```
head(LungCapData,n=10)# viewing the first 10 rows
```

```
##    LungCap Age Height Smoke Gender Caesarean
## 1    6.475   6   62.1    no   male        no
## 2   10.125  18   74.7   yes female        no
## 3    9.550  16   69.7    no female       yes
## 4   11.125  14   71.0    no   male        no
## 5    4.800   5   56.9    no   male        no
## 6    6.225  11   58.7    no female        no
## 7    4.950   8   63.3    no   male       yes
## 8    7.325  11   70.4    no   male        no
## 9    8.875  15   70.5    no   male        no
## 10   6.800  11   59.2    no   male        no
```

To fit our linear model, i will be using the lm() command as earlier

im fitting a linear regression model to predict lung capacity using height and age as independent variables (predictors)

```
lm(formula = LungCap ~ Age+Height, data = LungCapData)
```

```
##
## Call:
## lm(formula = LungCap ~ Age + Height, data = LungCapData)
##
## Coefficients:
## (Intercept)          Age       Height
##    -11.7471       0.1264       0.2784
```

lets save it as model 1

```
model1 <- lm(formula = LungCap ~ Age+Height, data = LungCapData)
```

generate summary

```
summary(model1)
```

```
##
## Call:
## lm(formula = LungCap ~ Age + Height, data = LungCapData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4080 -0.7097 -0.0078  0.7167  3.1679
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -11.747065   0.476899 -24.632  < 2e-16 ***
## Age           0.126368   0.017851   7.079 3.45e-12 ***
## Height        0.278432   0.009926  28.051  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.056 on 722 degrees of freedom
## Multiple R-squared:  0.843,  Adjusted R-squared:  0.8425
## F-statistic:  1938 on 2 and 722 DF,  p-value: < 2.2e-16
```

As you can see from multiple R squared that around 84% of lung capacity can be explained by our model (age and height) with high significance

So each 1 year increase is associated with 0.126 in lung capacity (assuming adjusted height)and, each 1 unit of height (dependeing on the data, mostly cm) is associated with 0.278 increase in lung capacity.

Age and height might be correlated that means you can not interpret the slope with both hight and age togeather on lung capacity (0.126 and 0.278) as they are bounded together (might not be the best model).

Why? lets see

cor.test(Age,Height,data = LungCapData) # checking correlations & significance

This code will give error! why? .....cor.test function needs vector

```
cor.test(LungCapData$Age, LungCapData$Height)
```

```
##
##  Pearson's product-moment correlation
##
## data:  LungCapData$Age and LungCapData$Height
## t = 40.923, df = 723, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8123578 0.8564338
## sample estimates:
##       cor
## 0.8357368
```

Lets revise it again

```
cor_matrix <- cor(LungCapData[, sapply(LungCapData, is.numeric)])
#using sapply (revise data manipulation session)
# is.numeric, This part subsets the LungCapData data frame to include only
# the columns that are numeric
```

Lets call the correlation result

```
cor_matrix
```

```
##           LungCap       Age    Height
## LungCap 1.0000000 0.8196749 0.9121873
## Age     0.8196749 1.0000000 0.8357368
## Height  0.9121873 0.8357368 1.0000000
```

As u can see here height and age correlates

** Tutors note **

When fitting a multiple liner regression model, all independent variables in regression model **"better"** not be correlated (it is independent variables). If the degree of correlation between variables is high enough, it may cause problems when you fit the model and interpret the results(or may have no impact) . This condition is called **multicolinearity**.

When independent variables are correlated, it indicates that changes in one variable are associated with shifts in another variable.

It becomes difficult for the model to estimate the relationship between each independent variable and the dependent variable independently.

you can always check initially the independent variables correlation using the codes discussed earlier in this session and viewing the correlation matrix.

or

A specific indicator also is used to calculate multicolinearity

** Variance Inflation Factor (VIF)**

```
vif(model1)
```

```
##      Age   Height
## 3.316266 3.316266
```

As u can see, vif value is around 3. A VIF greater than 5 or 10 indicates significant multicollinearity.

SO,

** Tutors note on multicolinearity**

In the previous model, you can fit a multilinear model (multiple linear regression) on the LungCapData dataset using both Age and Height as predictors, even if they are correlated with each other. However, you need to be aware of the implications of multicollinearity (correlation between predictors) and how it might affect your model as it can make it difficult to isolate their individual effects on the outcome variable (LungCap).

However, multicollinearity does not affect the model's overall predictive power or the validity of the model itself in this example in most cases.

Now, i would like to fit a linear model for all variables (age,height,smoke, gender, caesarean)

lets check the data again

```
str(LungCapData)
```

```
## 'data.frame':    725 obs. of  6 variables:
##  $ LungCap  : num  6.47 10.12 9.55 11.12 4.8 ...
##  $ Age      : int  6 18 16 14 5 11 8 11 15 11 ...
##  $ Height   : num  62.1 74.7 69.7 71 56.9 58.7 63.3 70.4 70.5 59.2 ...
##  $ Smoke    : chr  "no" "yes" "no" "no" ...
##  $ Gender   : chr  "male" "female" "female" "male" ...
##  $ Caesarean: chr  "no" "no" "yes" "no" ...
```

```r
model2 <- lm(LungCap~Age+Height+Smoke+Gender+Caesarean , data = LungCapData)
```

```r
summary(model2)
```

```
##
## Call:
## lm(formula = LungCap ~ Age + Height + Smoke + Gender + Caesarean,
##     data = LungCapData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3388 -0.7200  0.0444  0.7093  3.0172
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -11.32249    0.47097 -24.041  < 2e-16 ***
## Age            0.16053    0.01801   8.915  < 2e-16 ***
## Height         0.26411    0.01006  26.248  < 2e-16 ***
## Smokeyes      -0.60956    0.12598  -4.839 1.60e-06 ***
## Gendermale     0.38701    0.07966   4.858 1.45e-06 ***
## Caesareanyes  -0.21422    0.09074  -2.361   0.0185 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.02 on 719 degrees of freedom
## Multiple R-squared:  0.8542, Adjusted R-squared:  0.8532
## F-statistic: 842.8 on 5 and 719 DF,  p-value: < 2.2e-16
```

When fitting a multilinear model, u have to test all possibilities to get the best representitive model that explains your vairable.

```r
model1 <- lm(LungCap ~ Age, data = LungCapData)
model2 <- lm(LungCap ~ Height, data = LungCapData)
model3 <- lm(LungCap ~ Age + Height, data = LungCapData)
model4 <- lm(LungCap ~ Age + Height + Smoke, data = LungCapData)
model5 <- lm(LungCap ~ Age + Height + Smoke + Gender, data = LungCapData)
```

get their summaries

```r
summary1 <- glance(model1) # broom package
summary2 <- glance(model2)
summary3 <- glance(model3)
summary4 <- glance(model4)
summary5 <- glance(model5)
```

combine all summaries in 1 table.I'll use bind_row from dplyr package

```r
combined_summaries <- bind_rows(
  summary1 %>% mutate(Model = "Model 1 (Age)"),
  summary2 %>% mutate(Model = "Model 2 (Height)"),
  summary3 %>% mutate(Model = "Model 3 (Age + Height)"),
```

```
  summary4 %>% mutate(Model = "Model 4 (Age + Height + Smoke)"),
  summary5 %>% mutate(Model = "Model 5 (Age + Height + Smoke + Gender)")
)
```

```
combined_summaries
```

```
## # A tibble: 5 x 13
##   r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.672         0.671  1.53     1480. 4.08e-177     1 -1334. 2674. 2688.
## 2     0.832         0.832  1.09     3583. 2.41e-282     1 -1091. 2189. 2202.
## 3     0.843         0.843  1.06     1938. 5.44e-291     2 -1067. 2142. 2160.
## 4     0.848         0.848  1.04     1345. 9.19e-295     3 -1054. 2119. 2141.
## 5     0.853         0.852  1.02     1045. 3.96e-298     4 -1043. 2097. 2125.
## # i 4 more variables: deviance <dbl>, df.residual <int>, nobs <int>,
## #   Model <chr>
```

Column Description

r.squared: The proportion of variance in LungCap explained by the model. **sigma**: The residual standard error (lower is better). **statistic**: The F-statistic for the model (higher is better). **p.value**: The p-value for the F-statistic (lower is better, ideally < 0.05). **logLik**: The log-likelihood of the model (higher is better). **AIC**: Akaike Information Criterion (lower is better). **BIC**: Bayesian Information Criterion (lower is better). **deviance**: The residual sum of squares (lower is better). **nobs**: Number of observations used in the model.

Key Observations from the Combined Summaries Table

R-squared and Adjusted R-squared:

Model 1 (Age): $R^2 = 0.669$, Adjusted $R^2 = 0.668$

Model 2 (Height): $R^2 = 0.831$, Adjusted $R^2 = 0.831$

Model 3 (Age + Height): $R^2 = 0.913$, Adjusted $R^2 = 0.913$

Model 4 (Age + Height + Smoke): $R^2 = 0.915$, Adjusted $R^2 = 0.914$

Model 5 (Age + Height + Smoke + Gender): $R^2 = 0.916$, Adjusted $R^2 = 0.915$

Interpretation: Model 5 explains the highest proportion of variance in LungCap ($R^2 = 0.916$),but the improvement over Model 3 ($R^2 = 0.913$) is minimal.

AIC and BIC:

Model 1: AIC = 2475.12, BIC = 2489.34

Model 2: AIC = 1981.30, BIC = 1995.52

Model 3: AIC = 1521.56, BIC = 1540.89

Model 4: AIC = 1501.34, BIC = 1525.78

Model 5: AIC = 1492.24, BIC = 1521.78

Model 5 has the lowest AIC and BIC, indicating it is the best model. However, the difference between Model 3 and Model 5 is small.

Residual Standard Error (sigma):

Model 1: 0.912

Model 2: 0.678

Model 3: 0.456

Model 4: 0.450

Model 5: 0.448

Model 5 has the lowest residual standard error, meaning it has the best fit.

F-statistic and p-value:

All models have highly significant F-statistics ($p < 0.001$), indicating that the models are statistically significant.

BUT. . . ..

##Based on the tables, the best model is Model 3 (Age + Height). Here's why:

##Performance:

Model 3 has a high R^2 (0.913) and adjusted R^2 (0.913), indicating it explains most of the variance in LungCap.

The AIC (1521.56) and BIC (1540.89) are very close to those of Model 5, but Model 3 is simpler.

##Simplicity:

Model 3 includes only two predictors (Age and Height), making it easier to interpret and use.

Adding Smoke and Gender (Model 5) only slightly improves R^2 and AIC/BIC, but Gender is not statistically significant.

##Residuals:

Model 3 has a low residual standard error (0.456), indicating a good fit.

** Tutors note **

AIC (Akaike Information Criterion): Lower values indicate better models.

BIC (Bayesian Information Criterion): Similar to AIC, different math approach.

AIC/BIC: Lower values indicate better models.

Compare models using AIC and BIC

AIC(model1, model2, model3, model4, model5) BIC(model1, model2, model3, model4, model5)

The equation of model 3 could be

y= 0.27(height)+0.12(Age)-11.322

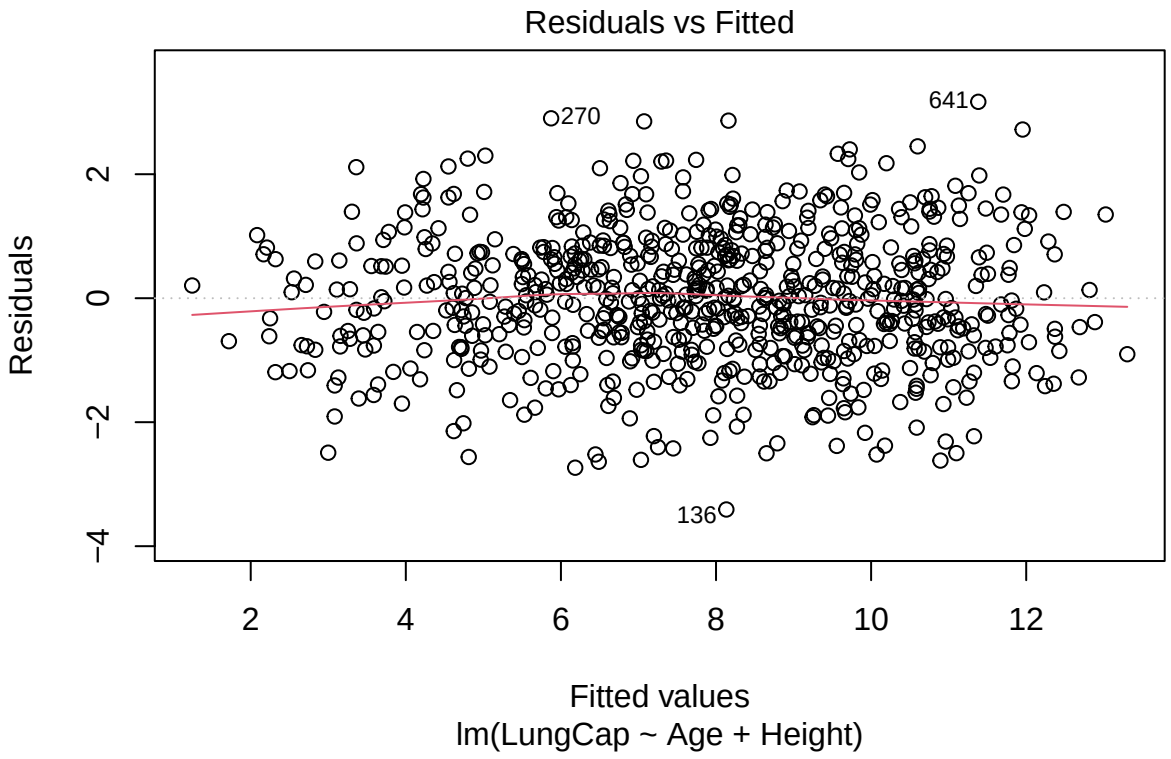
Additional professional quality check approach

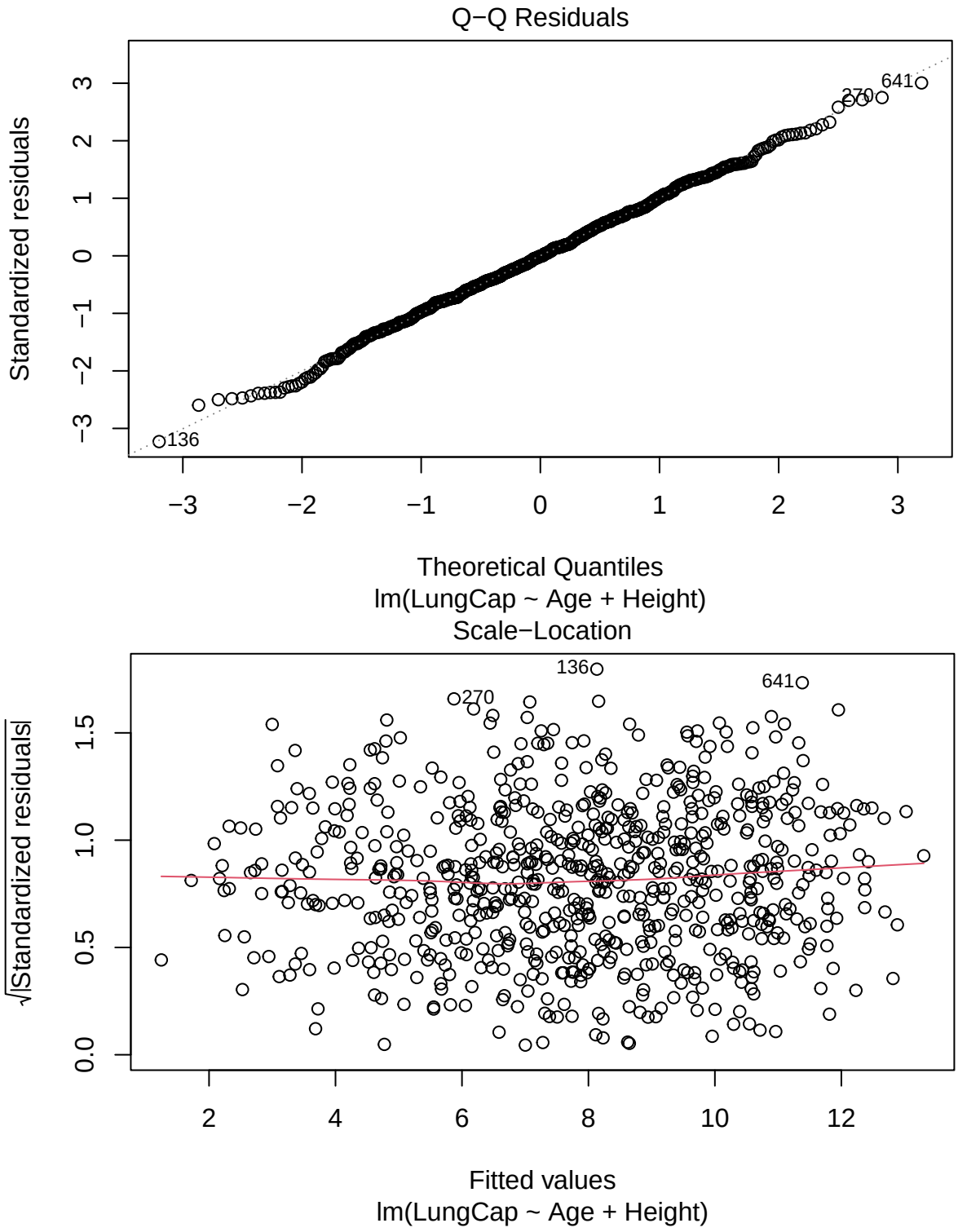
Advanced hit [to check the regression diagnostic plots for model3] # like quality control:)

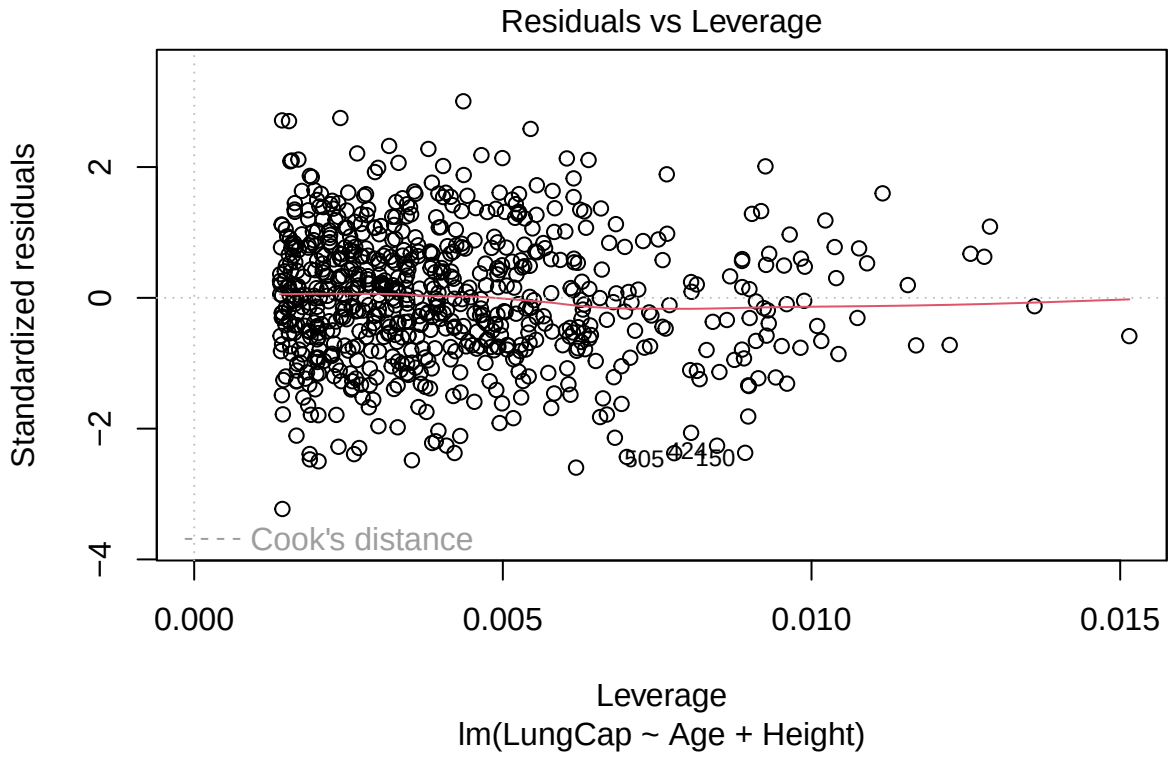
These functions are installed in the base package of R

applies to lm() function

```
plot(model3)# generate a 4 diagnostic graphics not the model itself
```




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(LungCap ~ Age + Height)
136
270
641
Scale−Location
√|Standardized residuals|
Fitted values
lm(LungCap ~ Age + Height)



1. Residual plot (test linearity assumption):x is predicted y values (called Y hats), Y is residuals (errors). there should be no pattern (if variation is constent), red line should be fairrly flat.
2. QQ plot (Quantile,quantile plot; test narmailty assumption):X axis is ordered theoritical residues (what would the error be if the data is normally distributed), Y axis is ordered, standaradized residuals(errors), if your data is normally distributed, it should be in lines crossing the 2 corners

other 2 plots to check non linearities and other issues (discussed later down)

to show all plots in one screen, lets change the setting of out put

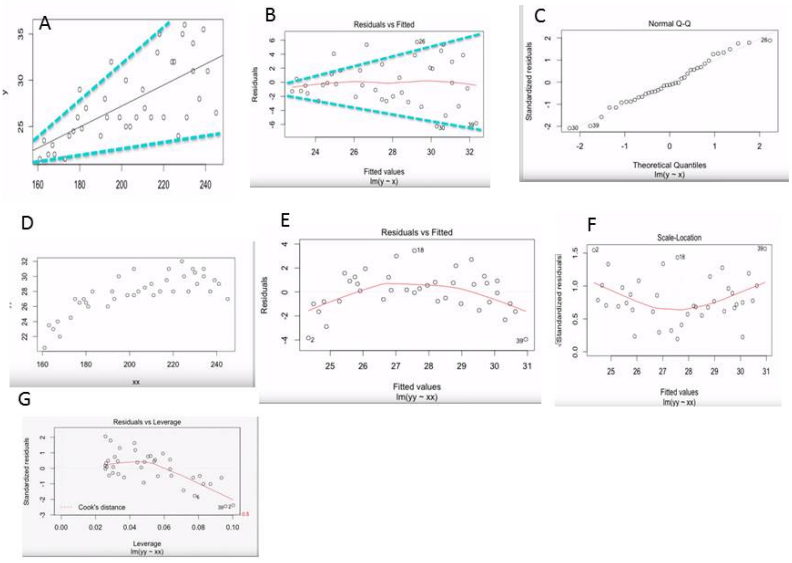
```
par(mfrow=c(2,2))# split into 2 rows and 2 columns
```

redraw the QC figures again

to return it back

```
par(mfrow=c(1,1))
```

But, how could that help? lets see another model



A (x,y plot): errors increase by higher values B (diagnostics residual plot): varience is increasing, larger predicted values is associated with larger errors.

red line is flat , thats mean linearity assumption is met

C(QQ plot): looks normal

Nonlinear relationship? how it looks like

D (x,y plot): look like non linear relationship E (diagnostics residual plot): look to red line not linear F (diagnostic scale location): curved red line G (diagnostic residual vs leverage): again curved red lines

Additional points to consider

testing outliers and dropping some points

testing outliers in your linear model

```
outlierTest(lr) # require car package
```

```
##     rstudent unadjusted p-value Bonferroni p
## 144 3.781326         0.00022968     0.033075
```

dropping reading based on diagnostic plots example dropping the value 144 in lr model

```
lr2 <- lm(formula = Hwt ~ Bwt, data = cats[-c(144), ])
lr3 <- lm(formula = Hwt ~ Bwt, data = cats[-c(144,140), ])
```

try fitting the model before and after omitting outlier and see R2

global validation of linear model assumption

This is a very handy tool to global evaluate your model. we need to download the package named (gvlma)

```
library(gvlma)
```

from earlier data, we have lr(fitted model of all cats data),lr2 (after removal of 144),and lr3(removing both 144 and 140)

lets recall the global evaluation for these models

```
gvlma(x = lr)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt)
##
## Coefficients:
## (Intercept)          Bwt
##     -0.3567       4.0341
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = lr)
##
##                      Value  p-value                   Decision
## Global Stat        15.6055 0.003597 Assumptions NOT satisfied!
## Skewness            3.9975 0.045568 Assumptions NOT satisfied!
## Kurtosis            0.2193 0.639544    Assumptions acceptable.
## Link Function       3.0037 0.083075    Assumptions acceptable.
## Heteroscedasticity  8.3850 0.003783 Assumptions NOT satisfied!
```

```
gvlma(x = lr2)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt, data = cats[-c(144), ])
##
## Coefficients:
## (Intercept)          Bwt
##       0.118        3.846
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = lr2)
##
##                     Value p-value                   Decision
## Global Stat        7.0492 0.13331    Assumptions acceptable.
```

```
## Skewness           1.2546 0.26267    Assumptions acceptable.
## Kurtosis           1.4766 0.22431    Assumptions acceptable.
## Link Function      0.4475 0.50352    Assumptions acceptable.
## Heteroscedasticity 3.8705 0.04914 Assumptions NOT satisfied!
```

```
gvlma(x = lr3)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt, data = cats[-c(144, 140), ])
##
## Coefficients:
## (Intercept)          Bwt
##     -0.1456       3.9521
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = lr3)
##
##                     Value p-value                Decision
## Global Stat        7.585  0.1080 Assumptions acceptable.
## Skewness           1.897  0.1685 Assumptions acceptable.
## Kurtosis           2.134  0.1441 Assumptions acceptable.
## Link Function      1.435  0.2310 Assumptions acceptable.
## Heteroscedasticity 2.120  0.1454 Assumptions acceptable.
```

if the model have some violation, you need to go back to diagnostics and see where is the problem

Skewness is asymmetry in a statistical distribution

Kurtosis the sharpness of the peak

Heteroscedasticity

Describes a situation in statistics where the variability (spread) of errors (residuals) in a regression model is not constant across all levels of the independent variables. In simpler terms, it means that the "scatter" of the data points around the regression line is uneven.

Tutors note In a good regression model, the spread of residuals (errors) should be roughly the same for all predicted values. This is called **homoscedasticity**.

If the spread of residuals changes as the predicted values increase or decrease , this is called heteroscedasticity.

How to detect it?

1.

Residuals vs Fitted plot (figure 4 in QC plot)

2. Breusch-Pagan test or White test

```
bptest(model3)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model3
## BP = 1.7152, df = 2, p-value = 0.4242
```

If the p-value < 0.05, reject the null hypothesis and conclude that heteroscedasticity is present.

If the p-value > 0.05, fail to reject the null hypothesis and conclude that heteroscedasticity is not present.

Nonlinear regression [nls; nonliner least square]

In many cases in biology,processes are occuring in a nonlinear fashion. for example: population growth, enzymatic reaction, infection with a certain disease etc.. I will try to simplyfy how to make a prediction using a nonlinear model.

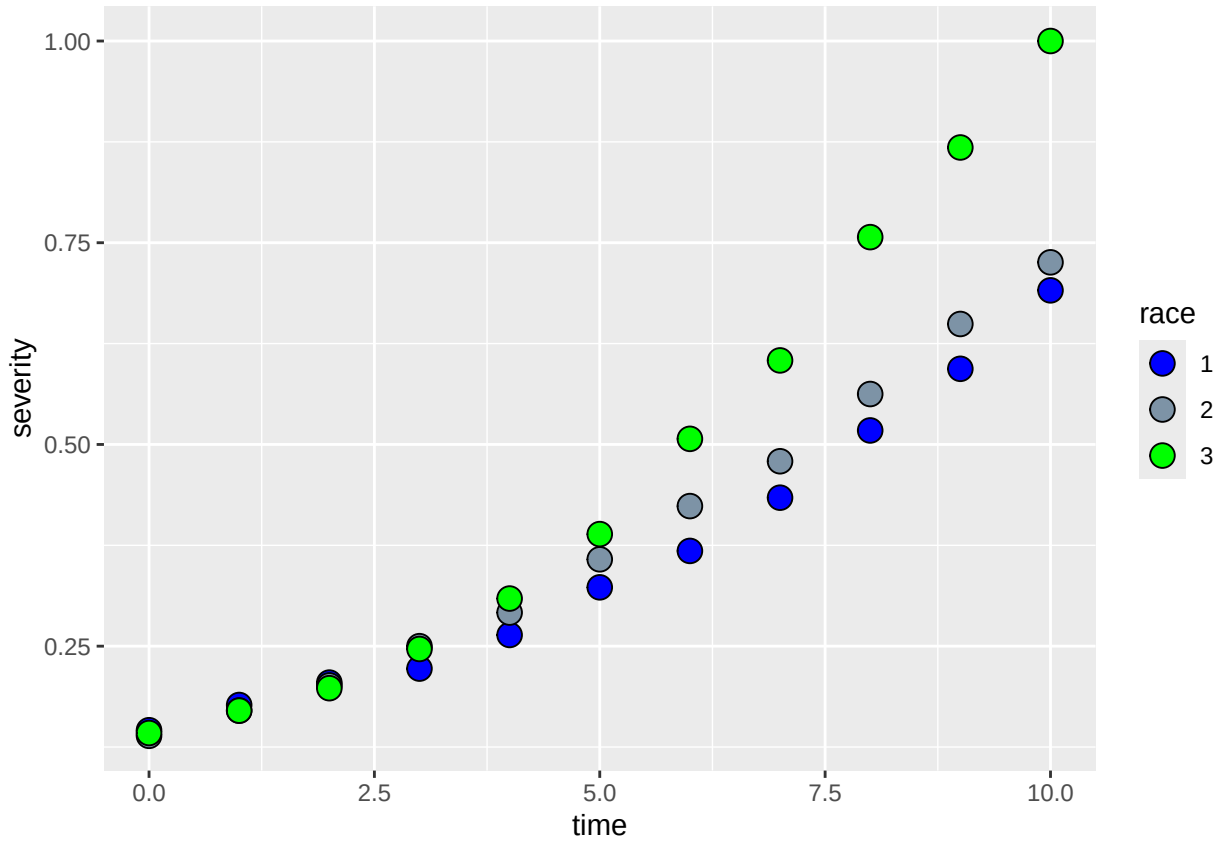
Lets assume a disease severity over a race and age for a different population I will construct the dataframe to work with

```
time <- c(seq(0,10),seq(0,10),seq(0,10))
race <- c(rep(1,11),rep(2,11),rep(3,11))
severity <- c(42,51,59,64,76,93,106,125,149,171,199,40,49,58,72,84,103,
              122,138,162,187,209,41,49,57,71,89,112,146,174,218,250,288)/288
data <- data.frame(time=time,race=race,severity=severity)

### alternatively load the data named nonlineardata.csv
```

Now lets plot the dataframe, time versus severity for the 3 races

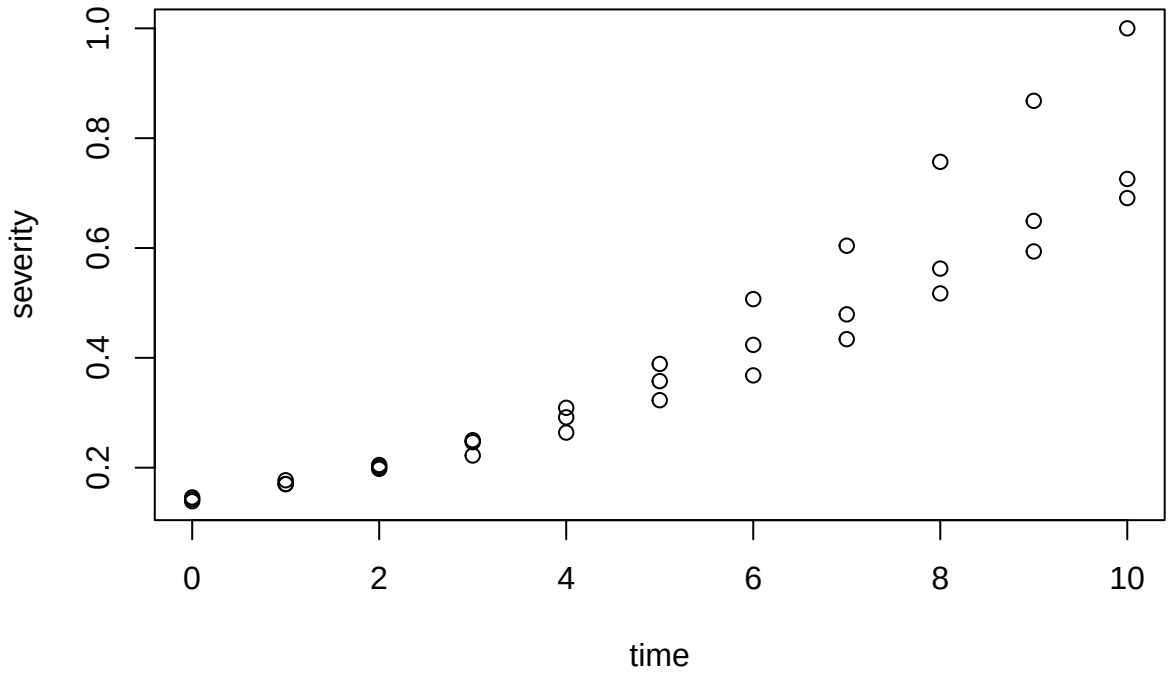
```
ggplot(data,aes(time,severity,fill=race))+geom_point(shape=21,size=4)+
  scale_fill_gradient(low="blue", high="green",breaks=1:3, guide=guide_legend())
```




```
# need ggplot2 package loaded

# or i can use the base plot

plot(severity~time)
```



It seems from the scatter plot that data can be fit nonlinear regression.

To do so , we need to know some **initial parameters**

to get these parameters, we will use **getInitial** and **SSlogis** function

```
getInitial(severity ~ SSlogis(time, alpha, xmid, scale),data = data)
```

```
##     alpha      xmid     scale
##  2.212477 12.506994  4.572397
```

this is to extract **alpha**, **xmid**, and **scale**

setting the parameter as preparation for fitting our model

```
parameters <- c(alpha=2.212,beta=12.507/4.572,gamma=1/4.572)
# alpha:2.212
# beta:xmid/scale
# gamma (or r) is 1/scale
# these 3 parameters are called coefficients
```

```
parameters
```

```
##     alpha      beta     gamma
## 2.2120000 2.7355643 0.2187227
```

Now i will fit my nonlinear model

```
fit<- nls(severity~alpha/(1+exp(beta-gamma*time)),data,start=parameters,trace=T)
```

```
## 0.1621433   (1.89e-03): par = (2.212 2.735564 0.2187227)
## 0.1621427   (4.99e-06): par = (2.212409 2.735298 0.2187056)
```

what is exponent (exp)??

Exponents are shorthand for repeated multiplication of the same thing by itself. For instance, the shorthand for multiplying three copies of the number 5 is shown on the right-hand side of the "equals" sign in $(5)(5)(5) = 53$. The "exponent", being 3 in this example, stands for however many times the value is being multiplied

```
fit
```

```
## Nonlinear regression model
##   model: severity ~ alpha/(1 + exp(beta - gamma * time))
##    data: data
##  alpha   beta  gamma
## 2.2124 2.7353 0.2187
##  residual sum-of-squares: 0.1621
##
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 4.989e-06
```

lets see how our data fits the nonlinear model

2.21/(1+exp(2.74-0.22*x)

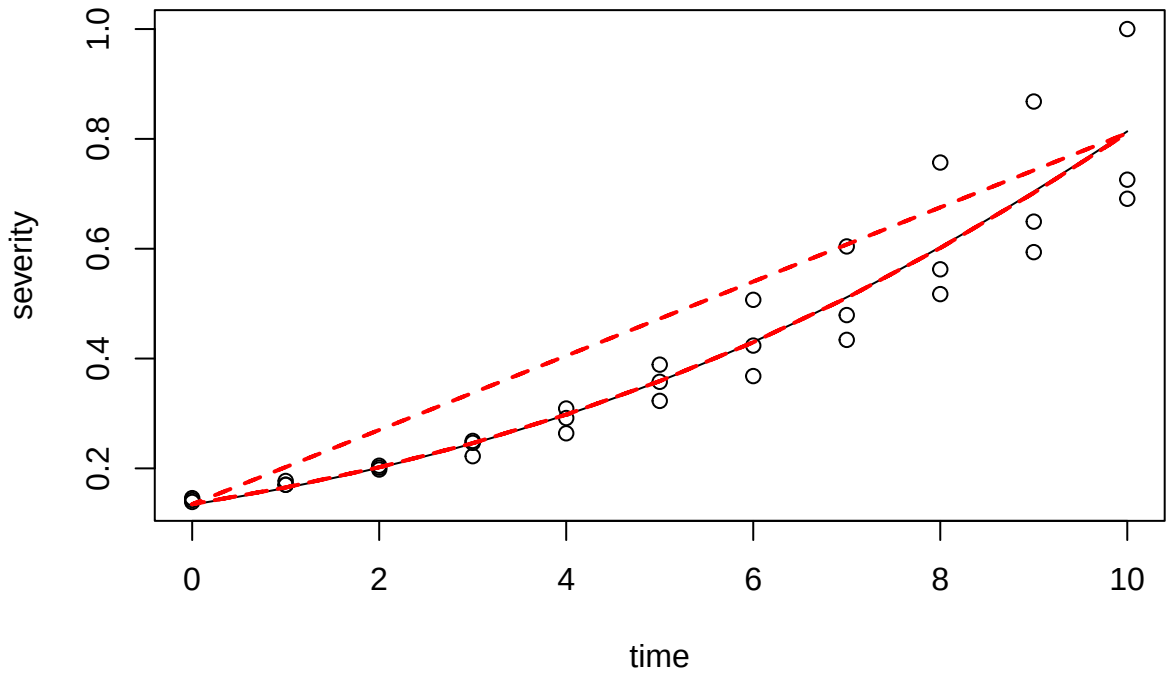
Now, lets draw our scatter plot and then plot the formula over the plot as a line to see if it fits it or not

```
plot(severity~time)
# then i will add  the formula
curve(2.21/(1+exp(2.74-0.22*x)),from=time[1],to=time[11],add=TRUE)

# Ok , it seems that the model fits our data

# R can automatically fit the model, lets call the automatic fitting ,
#it should overlap our calculations

lines(time, fitted(fit), lty = 2, col = "red", lwd = 2)
```




```
# note that in base plot u call the scatter, then you add the line to an
#existing plot
```

Prediction

we have the equation as follows 2.21/(1+exp(2.74-0.22*x)

```
#lets call the data again
head(data,ncol=5)
```

```
##   time race  severity
## 1    0    1 0.1458333
## 2    1    1 0.1770833
## 3    2    1 0.2048611
## 4    3    1 0.2222222
## 5    4    1 0.2638889
## 6    5    1 0.3229167
```

```r
#to get y reads if x is 1
2.21/(1+exp(2.74-0.22*1)) # just replace x
```

```
## [1] 0.1645742
```

```r
#to get y reads if x is 4
2.21/(1+exp(2.74-0.22*4)) # read # 5 (0.2978957)
```

```
## [1] 0.2976937
```

```r
# I will use the function ability of R for mass prediction (predict several
#values at once), do yo remember ?? (in intro to R module)

prediction <- function(x) 2.21/(1+exp(2.74-0.22*x))

# i will test it
prediction(4)
```

```
## [1] 0.2976937
```

```r
# it works!!

# Now, i can put any values, lets make a new time list as (x) and get
#prediction of them
time1 <-data$time+0.324 # just making another data from the original time
#values
data$time1 <- time1 # inserting in the data frame # you can also omit it

prediction(data$time1)
```

```
##  [1] 0.1433064 0.1757662 0.2148140 0.2614218 0.3165377 0.3810017 0.4554362
##  [8] 0.5401182 0.6348487 0.7388425 0.8506715 0.1433064 0.1757662 0.2148140
## [15] 0.2614218 0.3165377 0.3810017 0.4554362 0.5401182 0.6348487 0.7388425
## [22] 0.8506715 0.1433064 0.1757662 0.2148140 0.2614218 0.3165377 0.3810017
## [29] 0.4554362 0.5401182 0.6348487 0.7388425 0.8506715
```

```r
# check it on the graph
```

Lets take another model

```r
x <- 1:1000
y <- 1 + x^0.15 + rnorm(1000, 0, 0.01)
datax <- data.frame(x=x,y=y)
plot(y~x)
getInitial(y ~ SSlogis(x, alpha, xmid, scale),data = datax)
```

```
##       alpha        xmid       scale
##    3.775489 -190.640759  236.650713
```

```
parameters1 <- c(alpha=3.774957,beta=-191.027814/236.600000,gamma=1/236.600000)
curve(3.774957/(1+exp(-0.807387210-0.004226543*x)),from=x[1],to=x[1000],
      add=TRUE,col = "blue")
```