Class4

Dr. Pacifique

2025-10-08

Regression

Regression is a set of methodologies used to analyze the relationship between dependent(outcome variable or response variable) with independent variables also called inputs or explanatory variables. Regression can be used to provide an equation for predicting the response from explanatory variables.

Varieties of regression analysis and their use

Simple linear: Predicting a quantitative response variable from quantitative explanatory variable

Polynomial: Predicting a quantitative response variable from a quantitative explanatory variable where the relationship is modeled as an nth order polynomial

Multiple Linear: Predicting a quantitative from two or more explanatory variables.

Multilevel: Predictig a response variable from data that have hierarchical structure

Multivariate: Predicting more than one response variable from one or more explanatory variables.

Logistic: Predicting a categorical response variable from one or more explanatory variables.

Poisson: Predicting a response variable representing counts from on or more variables

Cox proportional hazards: Predicting time to event(death, failure, relapse) from one or more explanatory variables.

Time-series: Modelling time series data with correlated errors.

Nonlinear: Predicting a quantitative response variable from one or more explanatory variables where the form of the models is nonlinear

Nonparametric: Predicting a quantitative response from one or more explanatory variables, where the form of the model is derived from the data and not specified a prior.

Robust: Predicting a quantitative response variable from one or more explanatory variables using an approach that is resistant to the effect of influential observations.

Main task are : R function to fit OLS regression models, evaluate the fit, test assumptions, and select among competing models

Little explanation on Linear model

-It can be used to predict a continuous variable -It can be expressed as y=mx+b where b is the y intercept and m is the slope -It cam also be written as y=b0+b1x - the error in the prediction are called residuals - The line of the best fit will minimize the error -The residuals are squared and added up which is Residual sum of squares(RSS) -Mean square errror(MSE) -Root Mean Square error (RMSE) -Fitted model is the model(from a family of models) that is the closest to your data. -All models are wrong, but some are useful - The goal of a model is not to uncover the truth, but to discover a simple approximation that is still useful. #### OLS regression

Regressing the response variable on the predictor variables) the goal: To select model parameters (intercept and slopes) that minimize the difference between actual response and the predicted by the model.

Assumption 1. Normality: For fixed value of the independent values, the dependent variable is normally distributed 2. Independence: The Yi values are independent of each other 3. Linearity: The dependent variable is linearly related to the independent variables 4. Homoscedasticity: The variance of the dependent variable does not vary with the levels of the independent variables.

Note: If you violate these assumptions, your statistical significance tests, and confidence may not be accurate.

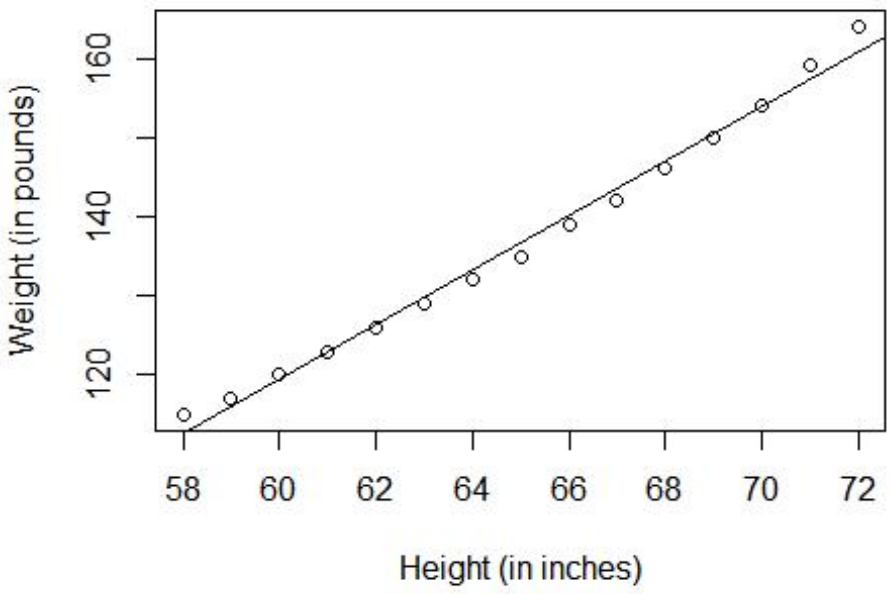
Fitting regression models with lm()

Simple linear regression

```
data("women")
fit<-lm(weight~height, data=women)
summary(fit)
```

```
##
## Call:
## lm(formula = weight ~ height, data = women)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7333 -1.1333 -0.3833  0.7417  3.1167
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -87.51667    5.93694  -14.74 1.71e-09 ***
## height        3.45000    0.09114   37.85 1.09e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.525 on 13 degrees of freedom
## Multiple R-squared:  0.991,  Adjusted R-squared:  0.9903
## F-statistic:  1433 on 1 and 13 DF,  p-value: 1.091e-14

women$weight

##  [1] 115 117 120 123 126 129 132 135 139 142 146 150 154 159 164

fitted(fit)

##        1        2        3        4        5        6        7
  8
## 112.5833 116.0333 119.4833 122.9333 126.3833 129.8333 133.2833 136.7
333
##        9       10       11       12       13       14       15
## 140.1833 143.6333 147.0833 150.5333 153.9833 157.4333 160.8833

residuals(fit)

##           1           2           3           4           5
  6
##  2.41666667  0.96666667  0.51666667  0.06666667 -0.38333333 -0.83333
333
##           7           8           9          10          11
 12
## -1.28333333 -1.73333333 -1.18333333 -1.63333333 -1.08333333 -0.53333
333
##          13          14          15
##  0.01666667  1.56666667  3.11666667

plot(women$height,women$weight,xlab="Height (in inches)",ylab="Weight
(in pounds)")
abline(fit)
```

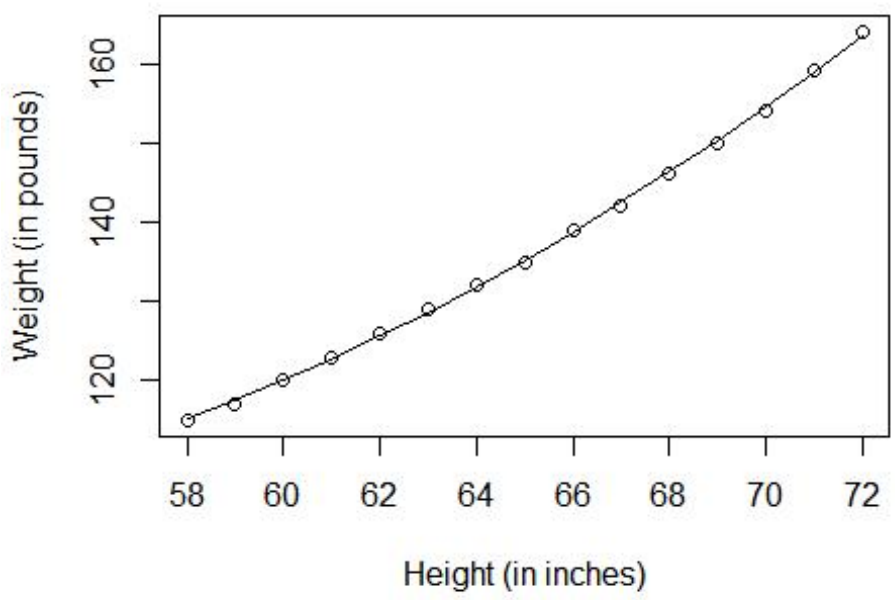


The plot suggests that you might be able to improve on the prediction by using a line with one bend. WE can fit the polynomial regression. #### Polynomial regression

```
fit2<- lm(weight~height+I(height^2), data=women)
summary(fit2)
```

```
##
## Call:
## lm(formula = weight ~ height + I(height^2), data = women)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.50941 -0.29611 -0.00941  0.28615  0.59706
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 261.87818   25.19677  10.393 2.36e-07 ***
## height       -7.34832    0.77769  -9.449 6.58e-07 ***
## I(height^2)   0.08306    0.00598  13.891 9.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3841 on 12 degrees of freedom
## Multiple R-squared:  0.9995, Adjusted R-squared:  0.9994
## F-statistic: 1.139e+04 on 2 and 12 DF,  p-value: < 2.2e-16
```

```
plot(women$height,women$weight,xlab="Height (in inches)",ylab="Weight
(in pounds)")
lines(women$height,fitted(fit2))
```



####

Multiple linear regression

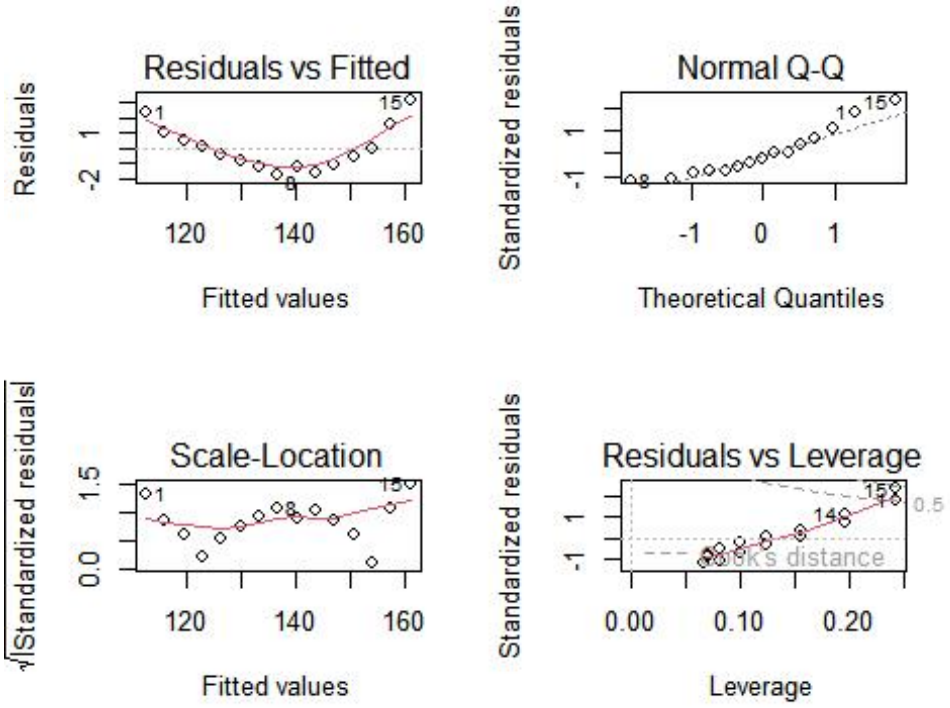
When we have more than one predictor variable, simple linear regression becomes multiple linear regression, and the analysis grows more involved.

HW: Find a data set of which you can fit multiple linear regression and interpret your results

Regression diagnostics

In this section you want to know if the model you have applied is appropriate. The most common approach is to apply the plot() function to the object returned by the lm(). This produces four graphs that are useful for evaluating the model fit.

```
fit<- lm(weight~height,data=women)
par(mfrow=c(2,2))
plot(fit)
```



Normality: the residual plot should be normally distributed with mean 0. The normal QQ plot (upper right) is probability plot of the standardized residuals against the value that would be expected under normality. If you have met the the normality assumption, the points on this graph should fall on the straight 45 degree line. Because they don't, you have clearly violated the normality assumption

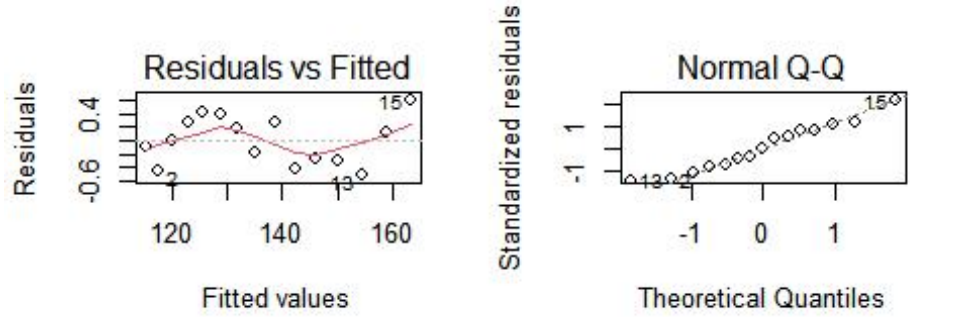
Independence: Judge based on how data were collected

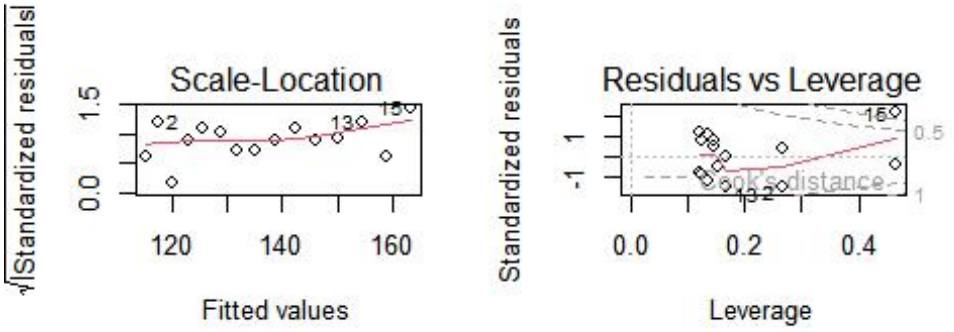
Linearity: If the dependent variable is linear related to the independent variables. there should be no systematic relationship between the residuals and the predicted (That is fitted) values. In the residuals vs. fitted graph (upper left), you see clear evidence of a curved relationship, which suggests that you may want to add a quadratic term to the regression.

The other graph helps to identify outliers, high laverage points, influential observations

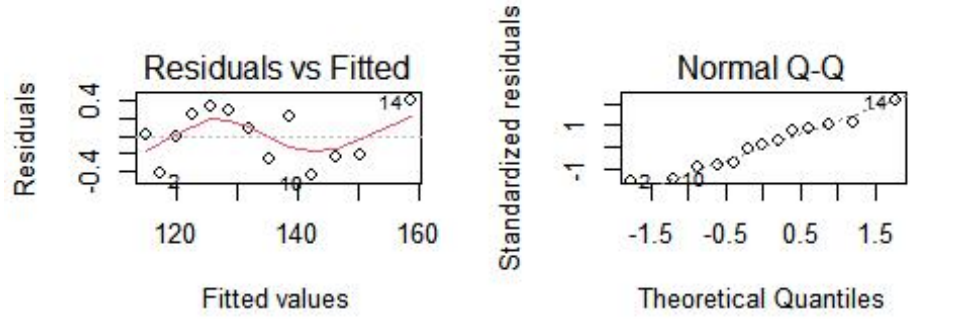
Homoscedasticity: If you have met the constant variance assumption, the point in the scale-location graph(bottom left) should be a random band around a horizontal line. you seem to meet this assumption.

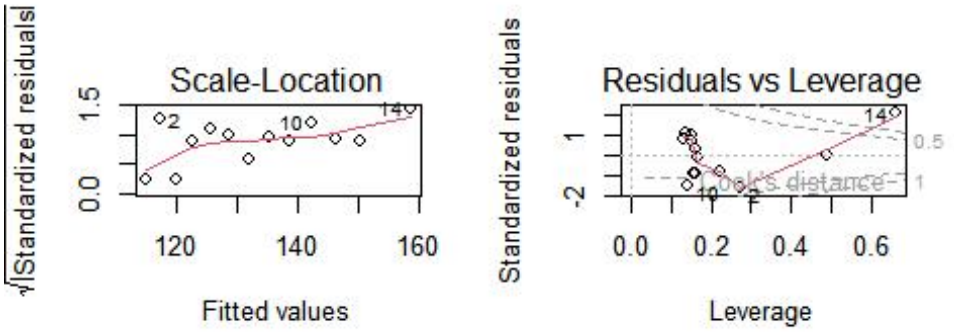
```
fit2<- lm(weight~height+I(height^2), data=women)
par(mfrow=c(2,2))
plot(fit2)
```






```
newfit<-lm(weight~height+I(height^2),data=women[-c(13,15),])
par(mfrow=c(2,2))
plot(newfit)
```





Outliers

The car package also provides a statistical test outliers. the function is "outlierTest()"

Corrective measures

What do you do if you identify problems? They are four approaches to dealing with violations of regression assumptions:

1. Deleting observations (influentials observation like outliers)
2. Transforming variables (transform response like y^(r))
3. Adding or deleting variables (sometimes to deal with multicolinearity)
4. Using another regression approach (Like robust regression, etc…)

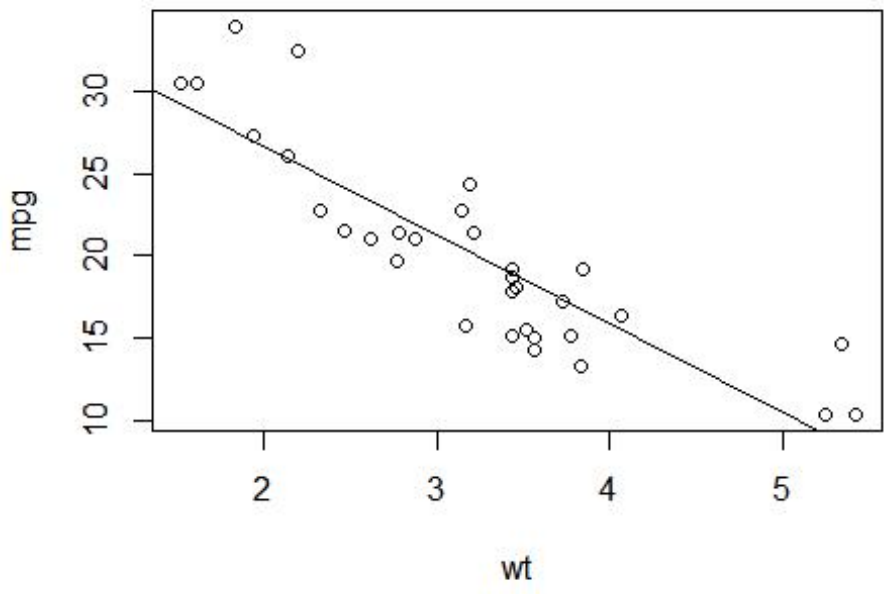
Selecting the best regression model

Shoud you include all variables under study? should you add polynomial or interactions terms to improve the fit? you make a decision based on predictive accuracy and simple and replicable model.

HW: Read about variable selection methods

Exploration of mtcars data

```
plot(mpg~wt,data=mtcars)
model<- lm(mpg~wt, data=mtcars)
#plot(model)
abline(model)
```




```
summary(model)
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

The results show three blocks Block 1: How the model was built Block 2: Five number summary of residuals Block 3: Coefficients and estimates

These are beta coefficients that minimize the RSS b0=37.285 and b1=-5.345

y=37.285+(-5.345)x

interpretation of b: for every unit of independent variable, the dependent variable goes down (Because its negative) 5.345 which are miles per gallon).

Multiple R-squared is like MSE, measure how good of a fit the model is. R^2 of 1 indicates a perfect fit with no residual error, 0 indicate the worst possible fit.

```
predict(model,newdata=data.frame(wt=6))
```

```
##        1
## 5.218297
```

Simulation in R

Simulation is an important (and big) topic for both statistics and for a variety of other areas where there is a need to introduce randomness. Sometimes you want to implement a statistical procedure that requires random number generation or sample (i.e. Markov chain Monte Carlo, the bootstrap, random forests, bagging) and sometimes you want to simulate a system and random number generators can be used to model random inputs

Generation of random numbers

We can simulate from probability distribution

dnorm(x, mean = 0, sd = 1, log = FALSE)

pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)

qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)

This gives a normal probability plot, The points in this plot will lie approximately on a straight line if the distribution is normal.

rnorm(n, mean = 0, sd = 1) or rnorm() This generate random number from standard normal distribution.

```
x <- rnorm(10)

pnorm(2)
```

```
## [1] 0.9772499
```

```
x <- rnorm(10, 20, 2)
```

Setting the random number seed

When simulating any random numbers it is essential to set the random number seed. Setting the random number seed with set.seed() ensures reproducibility of the sequence of random numbers.

```
set.seed(1)
rnorm(5)
```

```
## [1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
```

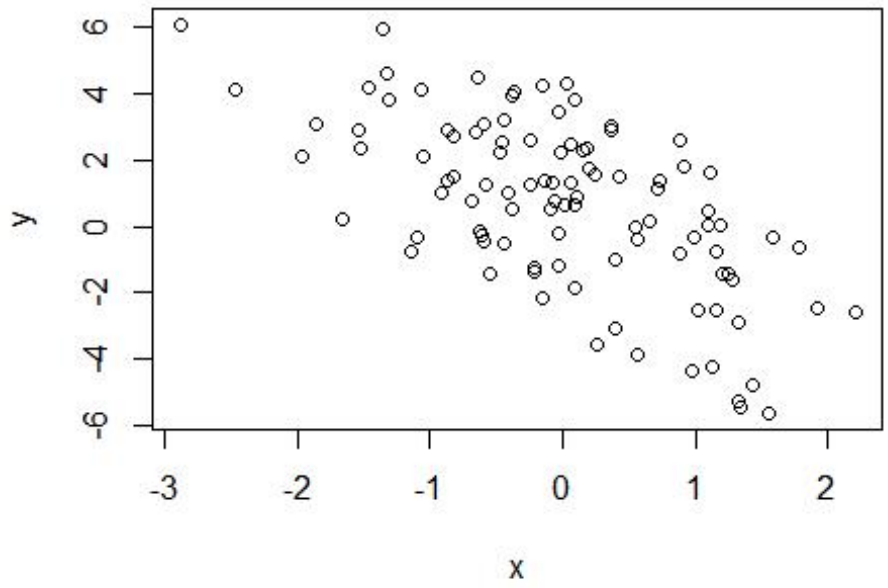
```
rnorm(5)
```

```
## [1] -0.8204684  0.4874291  0.7383247  0.5757814 -0.3053884
```

```
set.seed(1)
```

Simulaton a linear model

```
## Always set your seed!
 set.seed(20)

 ## Simulate predictor variable
 x <- rnorm(100)

 ## Simulate the error term
 e <- rnorm(100, 0, 2)

 ## Compute the outcome via the model
 y <- 0.5 - 2 * x + e

plot(x, y)
```




```
set.seed(1)
sample(1:10, 4)
```

```
## [1] 9 4 7 1
```

```r
sample(1:10, 4)
```

```
## [1] 2 7 3 6
```

```r
## Doesn't have to be numbers
sample(letters, 5)
```

```
## [1] "r" "s" "a" "u" "w"
```

```r
 ## Do a random permutation
 sample(1:10)
```

```
##  [1] 10  6  9  2  1  5  8  4  3  7
```

```r
sample(1:10)
```

```
##  [1]  5 10  2  8  6  1  4  3  9  7
```

```r
## Sample w/replacement
 sample(1:10, replace = TRUE)
```

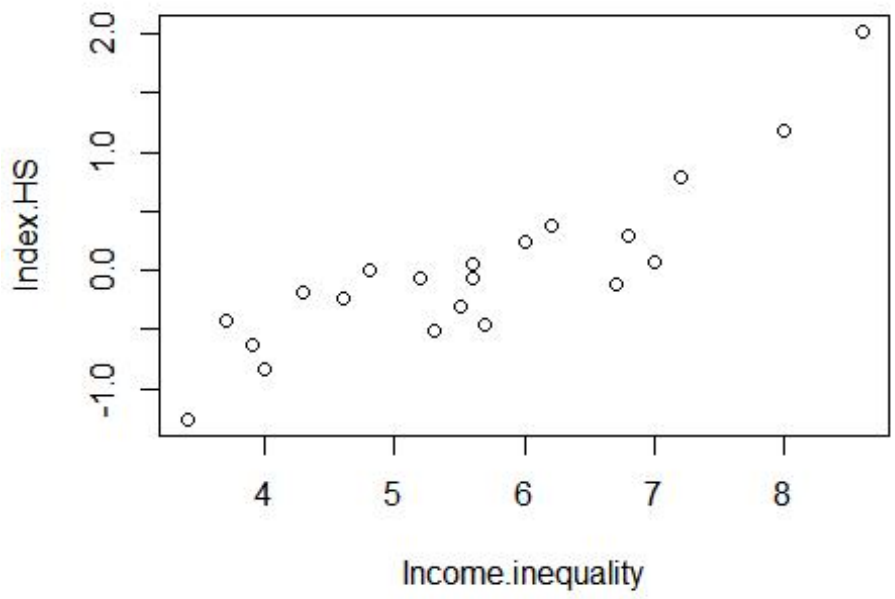
```
##  [1]  3  6 10 10  6  4  4 10  9  7
```

Explore the modelr package.

Data Visualization

Data source:
https://en.wikipedia.org/wiki/List_of_countries_by_GDP_(PPP)_per_capita

```r
country<-c("Australia", "Austria", "Belgium","Canada",
           "Denmark", "Finland","France","Germany",
           "Greece","Ireland","Italy","Japan", "Netherland",
           "New Zealand", "Norway", "Portugal", "Spain","Sweden",
           "Switzerland","UK","USA")

Income.inequality<-c(7.0,4.8,4.6,5.6,4.3,3.7,5.6,5.2,6.2,6.0,6.7,3.4,5.
3,6.8,3.9,8.0,5.5,4.0,5.7,7.2,8.6)

Index.HS<-c(0.07,0.01,-0.23,-0.07,-0.19,-0.43,0.05,-0.06,0.38,0.25,-0.1
2,-1.26,-0.51,0.29,-0.63
            ,1.18,-0.30,-0.83,-0.46,0.79,2.02)

GDP_WB<-c(45926,47682,43435,45066,45537,30676,39328,46401,26851,49393,3
5463,36319,48253,37679,65615
          ,28760,33629,45297,59540,40233,54630)

data.21<-data.frame(country,Income.inequality,Index.HS,GDP_WB)

plot(data.21[c("Income.inequality","Index.HS")])
```




```
Index_inequality.df<-data.21[c("Income.inequality","Index.HS")]
str(Index_inequality.df)

## 'data.frame':    21 obs. of  2 variables:
##  $ Income.inequality: num  7 4.8 4.6 5.6 4.3 3.7 5.6 5.2 6.2 6 ...
##  $ Index.HS         : num  0.07 0.01 -0.23 -0.07 -0.19 -0.43 0.05 -0.
06 0.38 0.25 ...

(country<-data.21[,"country"])

##  [1] "Australia"   "Austria"     "Belgium"     "Canada"      "Denmar
k"
##  [6] "Finland"     "France"      "Germany"     "Greece"      "Irelan
d"
## [11] "Italy"       "Japan"       "Netherland"  "New Zealand" "Norway
"
## [16] "Portugal"    "Spain"       "Sweden"      "Switzerland" "UK"

## [21] "USA"

(country.2<-data.21["country"])

##        country
## 1    Australia
## 2      Austria
## 3      Belgium
## 4       Canada
## 5      Denmark
```

```
## 6      Finland
## 7       France
## 8      Germany
## 9       Greece
## 10     Ireland
## 11       Italy
## 12       Japan
## 13  Netherland
## 14 New Zealand
## 15      Norway
## 16    Portugal
## 17       Spain
## 18      Sweden
## 19 Switzerland
## 20          UK
## 21         USA
```

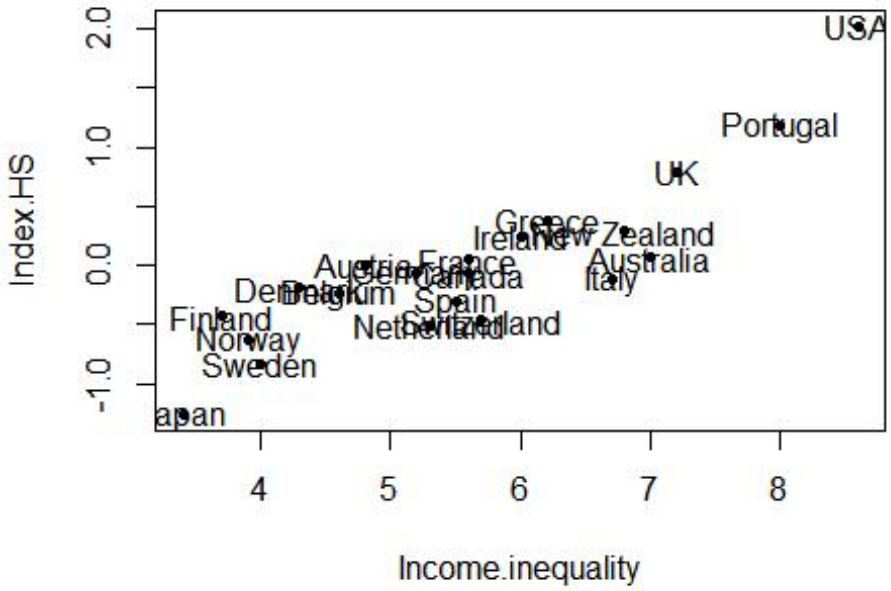
```
str(country)
```

```
##  chr [1:21] "Australia" "Austria" "Belgium" "Canada" "Denmark" "Finl
and" ...
```

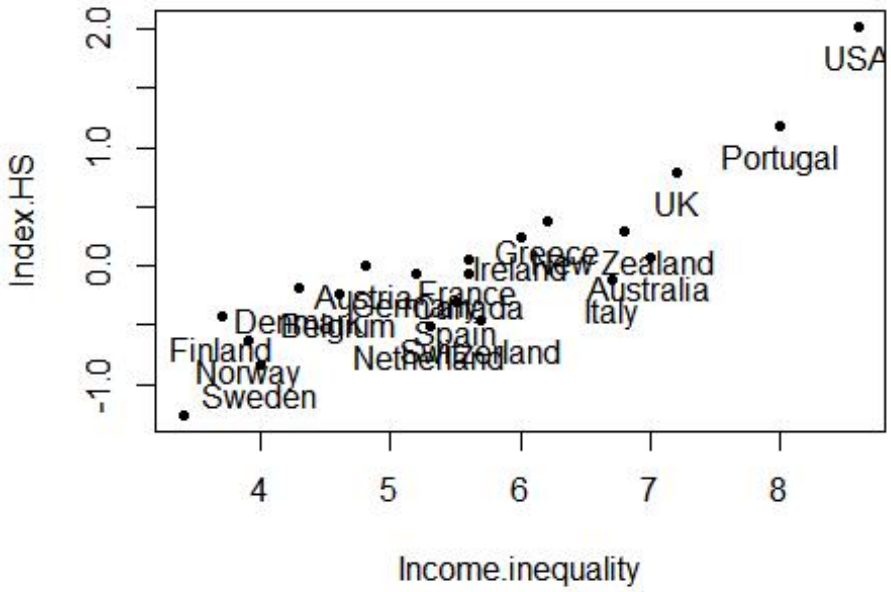
```
str(country.2)
```

```
## 'data.frame':    21 obs. of  1 variable:
##  $ country: chr  "Australia" "Austria" "Belgium" "Canada" ...
```

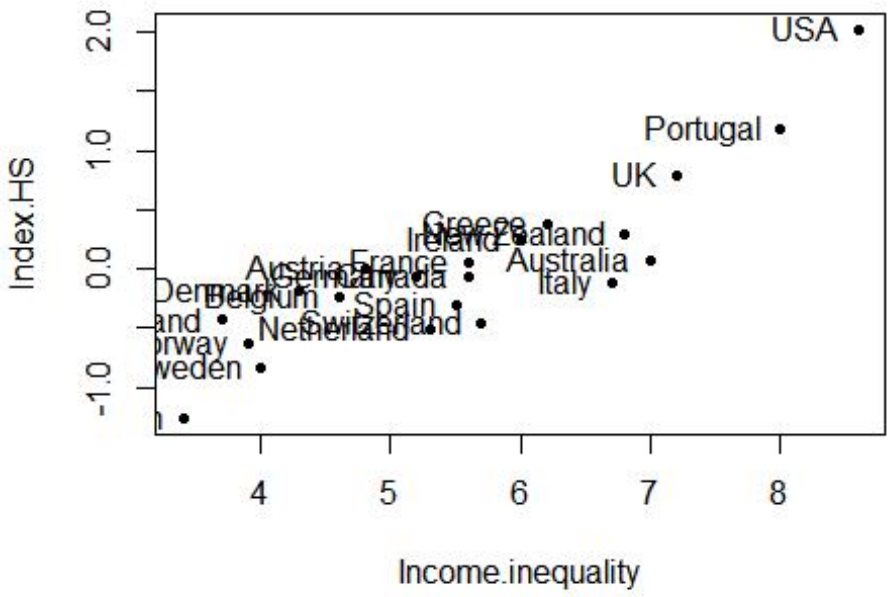
```
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country)
```



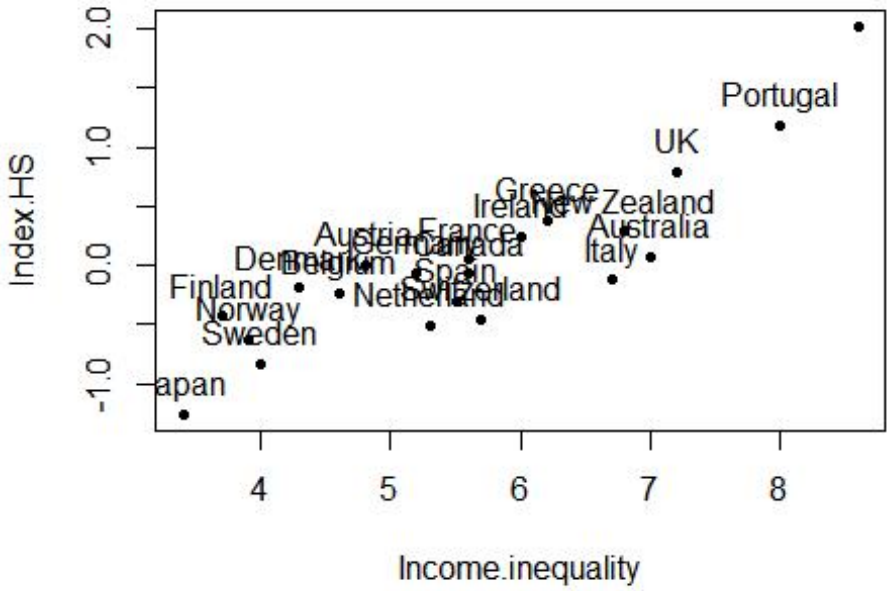

```r
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country,pos=1)
```



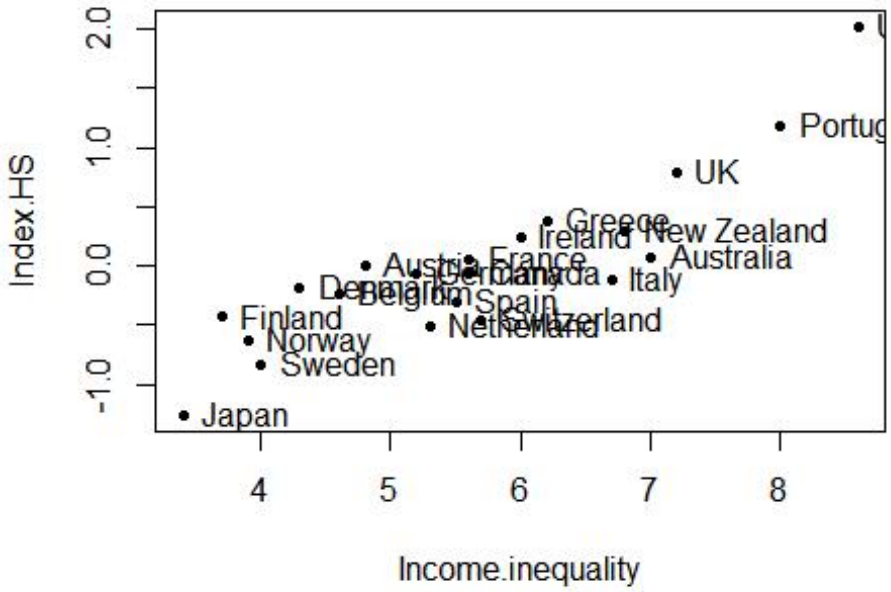
```
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country,pos=2)
```



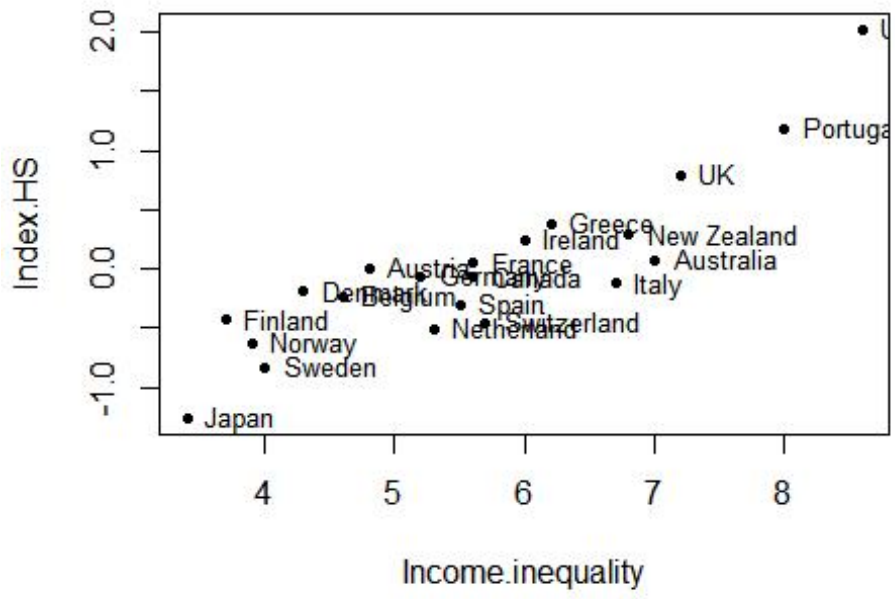

```
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country,pos=3)
```




```
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country,pos=4)
```

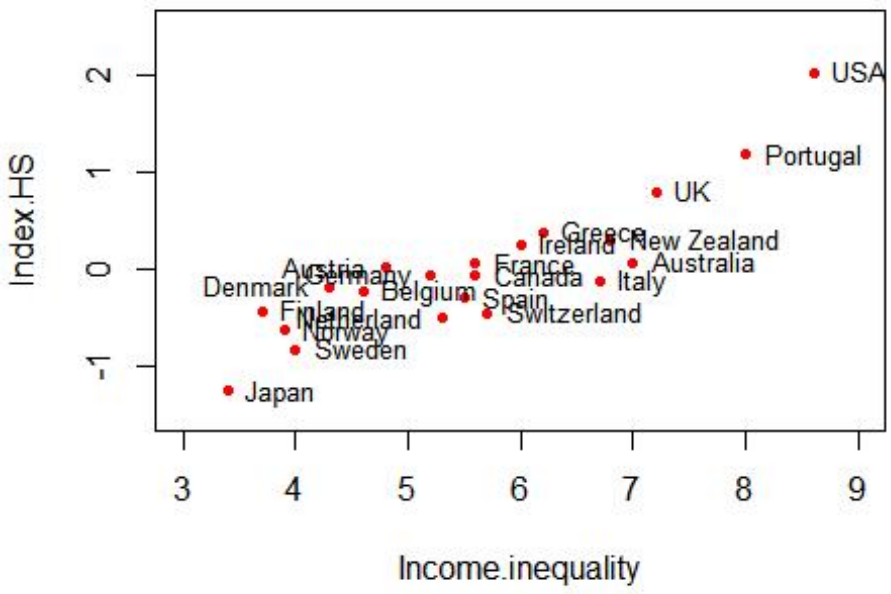


```
plot(Index_inequality.df,pch=20)
text(Index_inequality.df,labels=country,pos=4,cex=0.8)
```



Overlapping text

```
which(country %in% c("Austria","Denmark","Germany","Netherland"))

## [1]  2  5  8 13

text.left<-which(country %in% c("Austria","Denmark","Germany","Netherla
nd"))
text.left

## [1]  2  5  8 13

text.right<-setdiff(1:nrow(data.21),text.left)
text.right

##  [1]  1  3  4  6  7  9 10 11 12 14 15 16 17 18 19 20 21

pos.text<-ifelse(1:nrow(data.21)%in% text.left,2,4)

plot(Index_inequality.df,pch=20,col="red",xlim=c(3,9),ylim=c(-1.5,2.5))
text(Index_inequality.df,labels=country,pos=pos.text,cex=0.8)
```




```
which(country %in% "Germany")
```

```
## [1] 8
```

```
text.up<-which(country %in% "Germany")
text.up
```

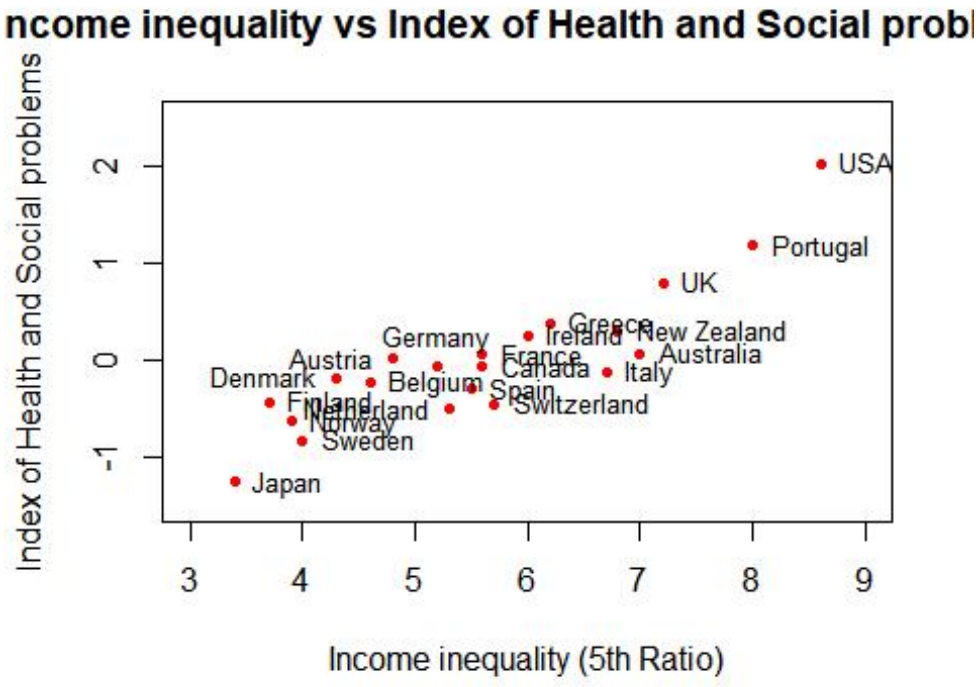
```
## [1] 8
```

```
text.left<-setdiff(1:nrow(data.21),c(text.right,text.up))
text.left
```

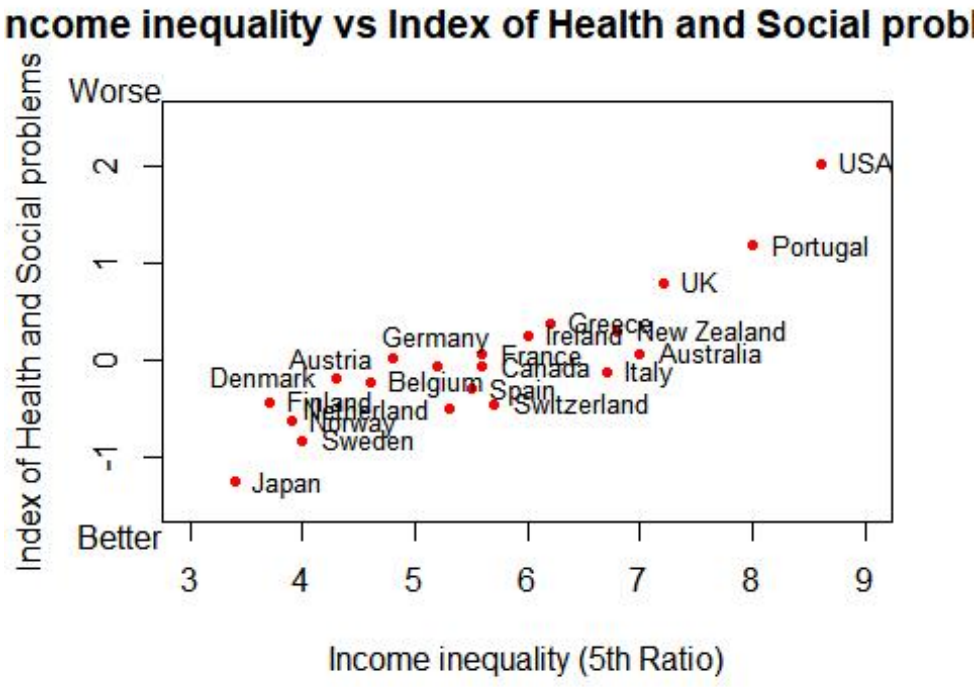
```
## [1]  2  5 13
```

```
pos.text<-ifelse(1:nrow(data.21) %in% text.up,3,ifelse(1:nrow(data.21)%
in% text.left,2,4))
```

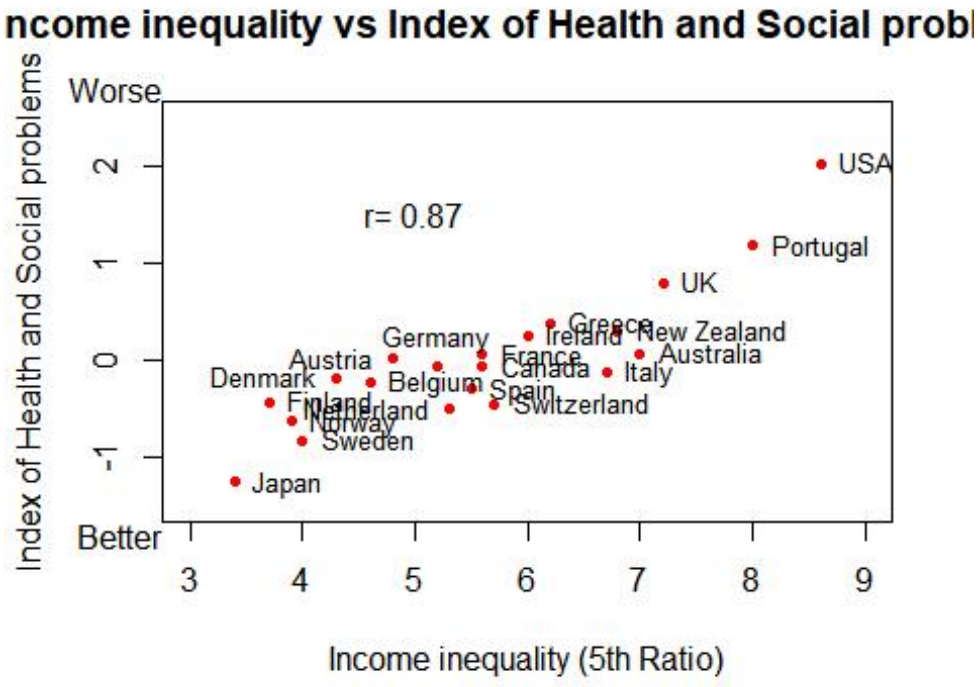
```
plot(Index_inequality.df,pch=20,col="red",xlim=c(3,9),ylim=c(-1.5,2.5),
ann=FALSE)
text(Index_inequality.df,labels = country,pos=pos.text,cex=0.8)
main.title<-"Income inequality vs Index of Health and Social problems"
x.lab<-"Income inequality (5th Ratio)"
y.lab<-"Index of Health and Social problems"
title(main=main.title,xlab=x.lab,ylab=y.lab)
```



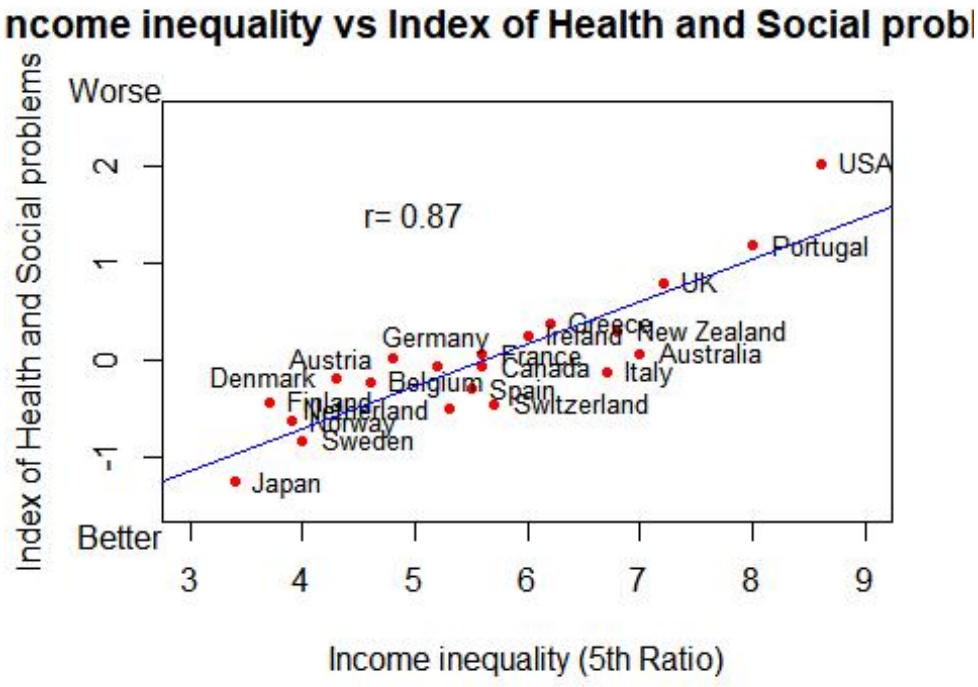

```
plot(Index_inequality.df,pch=20,col="red",xlim=c(3,9),ylim=c(-1.5,2.5),
ann=FALSE)
text(Index_inequality.df,labels = country,pos=pos.text,cex=0.8)
main.title<-"Income inequality vs Index of Health and Social problems"
x.lab<-"Income inequality (5th Ratio)"
y.lab<-"Index of Health and Social problems"
title(main=main.title,xlab=x.lab,ylab=y.lab)
mtext(c("Better","Worse"),side=2,at=c(-1.8,2.8),las=1)
```




```
plot(Index_inequality.df,pch=20,col="red",xlim=c(3,9),ylim=c(-1.5,2.5),
ann=FALSE)
text(Index_inequality.df,labels = country,pos=pos.text,cex=0.8)
main.title<-"Income inequality vs Index of Health and Social problems"
x.lab<-"Income inequality (5th Ratio)"
y.lab<-"Index of Health and Social problems"
title(main=main.title,xlab=x.lab,ylab=y.lab)
mtext(c("Better","Worse"),side=2,at=c(-1.8,2.8),las=1)
text(x=5,y=1.5,labels=paste("r=",round(cor(Index_inequality.df[1],Index
_inequality.df[2]),digits=2)))
```




```
plot(Index_inequality.df,pch=20,col="red",xlim=c(3,9),ylim=c(-1.5,2.5),
ann=FALSE)
text(Index_inequality.df,labels = country,pos=pos.text,cex=0.8)
main.title<-"Income inequality vs Index of Health and Social problems"
x.lab<-"Income inequality (5th Ratio)"
y.lab<-"Index of Health and Social problems"
title(main=main.title,xlab=x.lab,ylab=y.lab)
mtext(c("Better","Worse"),side=2,at=c(-1.8,2.8),las=1)
text(x=5,y=1.5,labels=paste("r=",round(cor(Index_inequality.df[1],Index
_inequality.df[2]),digits=2)))
lm.ineq<-lm(Index.HS~Income.inequality,data=Index_inequality.df)
abline(lm.ineq$coef,col="blue")
```




```
rm(list=ls(all=TRUE))
###set.seed(1002)
n <-30  ## number of observations
M <- 1 ## number of pathways
p<-5
z <- matrix(runif(n * p, 0, 1), nrow=n, ncol=p)
x <- 3*cos(z[, 1]) + 2*rnorm(n)
x<-as.matrix(x)
beta.true <- rep(1,ncol(x))
## pathway-response function
hfun1 <- function(zvec) (10*cos(zvec[1]) - 15*(zvec[2])^2+10*exp(-zvec
[3])*zvec[4]-8*sin(zvec[5])*cos(zvec[3])+20*(zvec[1]*zvec[5]))
h1 <- apply(z, 1, hfun1) ## only depends on z1,z2,z3,z4,z5
eps <- rnorm(n)
y <-x * beta.true + h1+ eps
```