Regressões em R

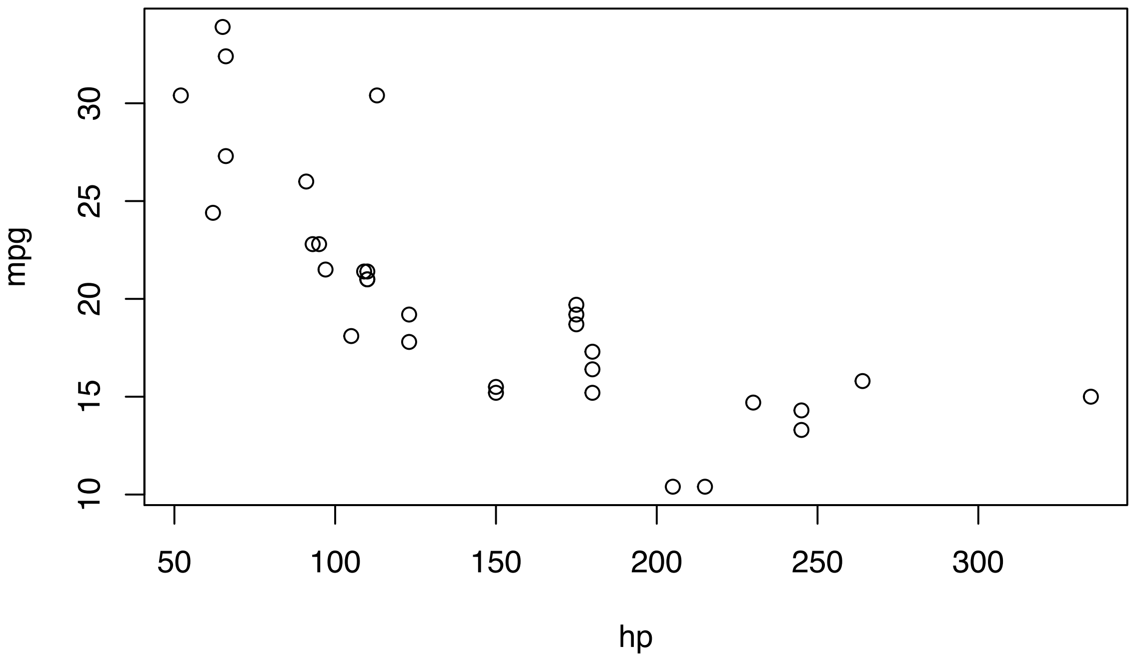
Danilo A C Souto

Regressão Linear

Para regressão linear no R utilizamos a função `lm()`

Mas primeiro vamos olhar os dados da primeira aula.

```
plot(mpg ~ hp, data = mtcars)
```



Verificar a correlação entre os dados.

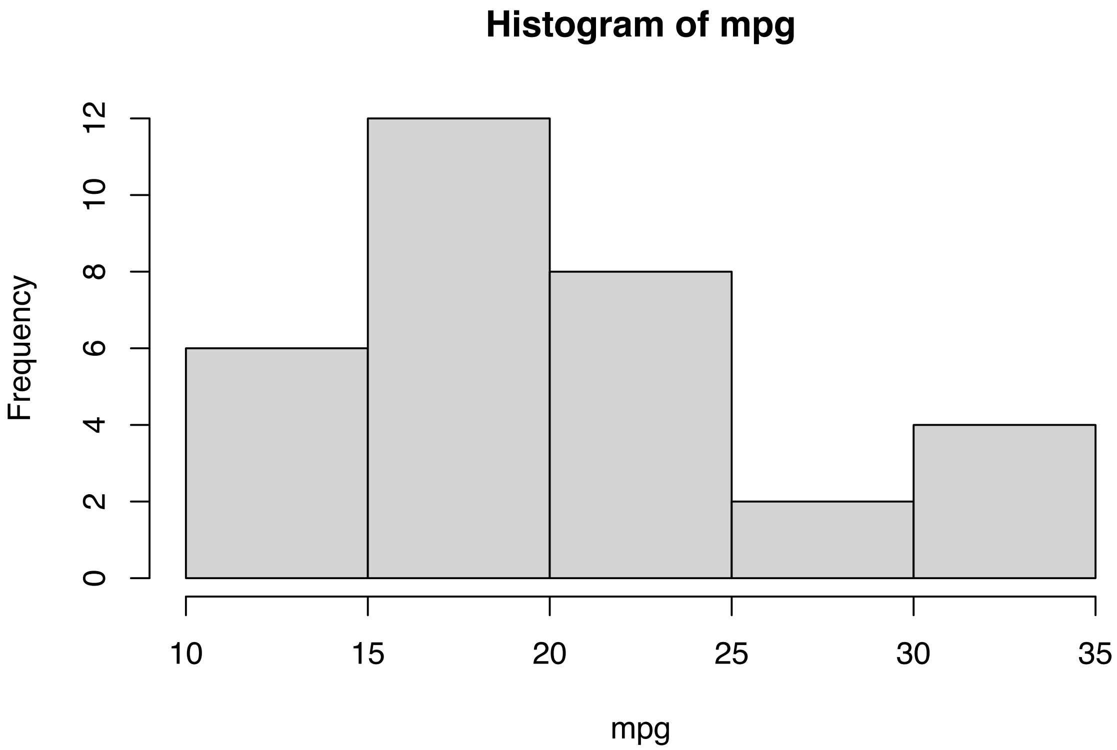
```
#funcao para evitar digitar mtcars todo momento
attach(mtcars)
#covarianca
covaricanca <- cov(mpg,hp)
#correlacao
correlacao <- cor(mpg,hp)
sprintf("Cov:%f  Cor:%f",covaricanca,correlacao)
```

```
## [1] "Cov:-320.732056  Cor:-0.776168"
```

Como pode ser verificado, há sim uma correlação forte entre os dados Sim, há correlação

Observando a distribuição da variável resposta não parece muito bem com relação há Normal.

```
hist(mpg)
```




```r
#verifica se há normalidade
shapiro.test(mpg)$p.value
```

```
## [1] 0.1228814
```

```r
#verifica se há igualdade nas variancas
fligner.test(mpg ~ hp, data = mtcars)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  mpg by hp
## Fligner-Killeen:med chi-squared = 25.126, df = 21, p-value = 0.2417
```

Mas quando realizando Shapiro verificamos que não rejeitamos H0

Portanto Agora vamos fazer a regressão linear com `lm()`.

No R, passamos uma formula para a funçao para dizer que o valor de y é em função de x, ou neste caso, mpg ~ hp

```r
modelo.linear <- lm(mpg ~ hp, data = mtcars)
```

Então, quase que magicamente, o modelo surge. Ou seja todas as operações vistas na última aula foram calculados somente com uma linha.

Temos então todos os coeficientes calculados. Pode-se utilizar a função `coefficients(modelo)` para extrair os betas do modelo.

```r
coefficients(modelo.linear)
```

```
## (Intercept)          hp
## 30.09886054 -0.06822828
```

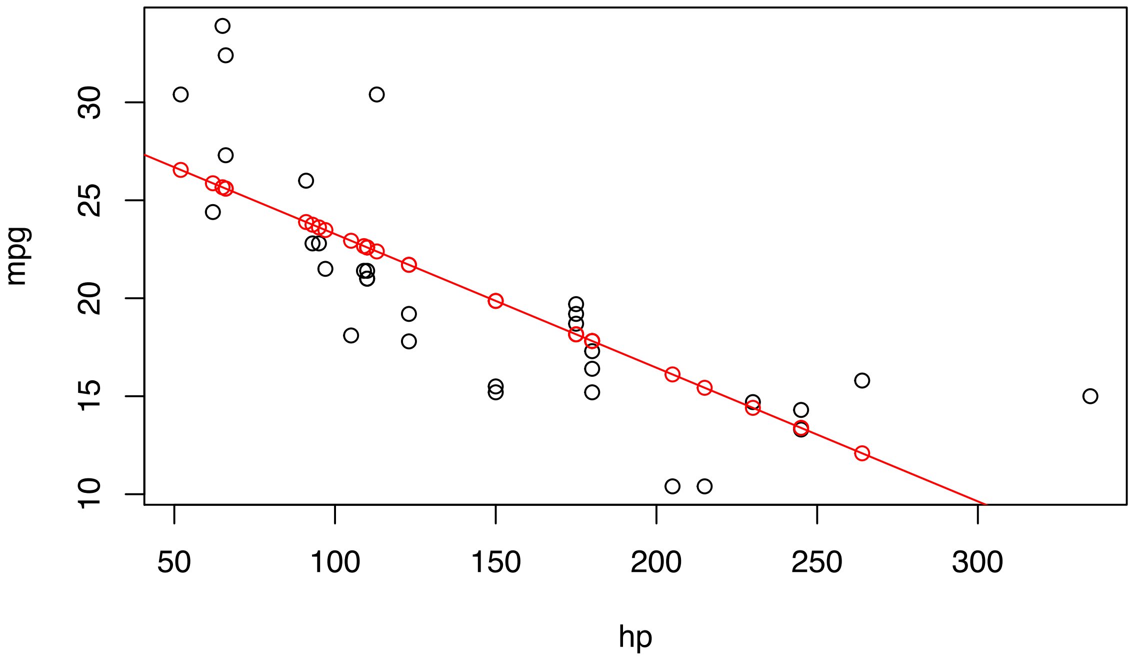
```r
modelo.linear$coefficients
```

```
## (Intercept)          hp 
## 30.09886054 -0.06822828
```

Dessa maneira fácil temos os coeficientes da reta: **f(x) = 30.0988605 + -0.0682283 * x**

Mas, para ficar mais fácil ainda não precisamos criar a função para mostrar o gráfico. Basta utilizar a funcão de `abline()` para plotar a reta.

```r
#plotar mpg(y) em funcao de hp (x)
plot(mpg ~ hp)
#plotar a reta do modelo
abline(modelo.linear, col = "red")
#plotar os valores estimados de mpg para o modelo
points(modelo.linear$fitted.values~ hp, col = "red", data = mtcars)
```



Aqui podemos ver os valores de y estimados para mpg pelo modelo,

`fitted.values(modelo.linear)`

ou

```r
modelo.linear$fitted.values
```

```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive 
##           22.593750           22.593750           23.753631           22.593750 
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D 
##           18.158912           22.934891           13.382932           25.868707 
##            Merc 230            Merc 280           Merc 280C          Merc 450SE 
##           23.617174           21.706782           21.706782           17.817770 
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental 
##           17.817770           17.817770           16.112064           15.429781 
##   Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla 
##           14.406357           25.595794           26.550990           25.664022 
##       Toyota Corona    Dodge Challenger          AMC Javelin          Camaro Z28
```

```
##          23.480718           19.864619           19.864619           13.382932
##    Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
##           18.158912           25.595794           23.890087           22.389065
##      Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
##           12.086595           18.158912            7.242387           22.661978
```

O R ainda calcula mais informações sobre o modelo. Basta usar a função `summary()` para exibir.

Aqui sao exibidos o p-value para cada coeficiente. Com ele é possível ver a sua importãoncia para o modelo.

Assim como o t-value. exibe tambem o Coeficiente de Determinação (r^2) em R-squared e o Coeficiente de Determinação Ajustado em Ajusted R-squared.

Também é possível ver informações dos resíduos

```r
summary(modelo.linear)
```

```
##
## Call:
## lm(formula = mpg ~ hp, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7121 -2.1122 -0.8854  1.5819  8.2360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
## hp          -0.06823    0.01012  -6.742 1.79e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.863 on 30 degrees of freedom
## Multiple R-squared:  0.6024, Adjusted R-squared:  0.5892
## F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07
```

Quer obter somente o coeficiente de determinação ?

```r
#coeficiente de determinacao
summary(modelo.linear)$r.squared
```

```
## [1] 0.6024373
```

```r
#coeficiente de determinacao Ajustado
summary(modelo.linear)$adj.r.squared
```

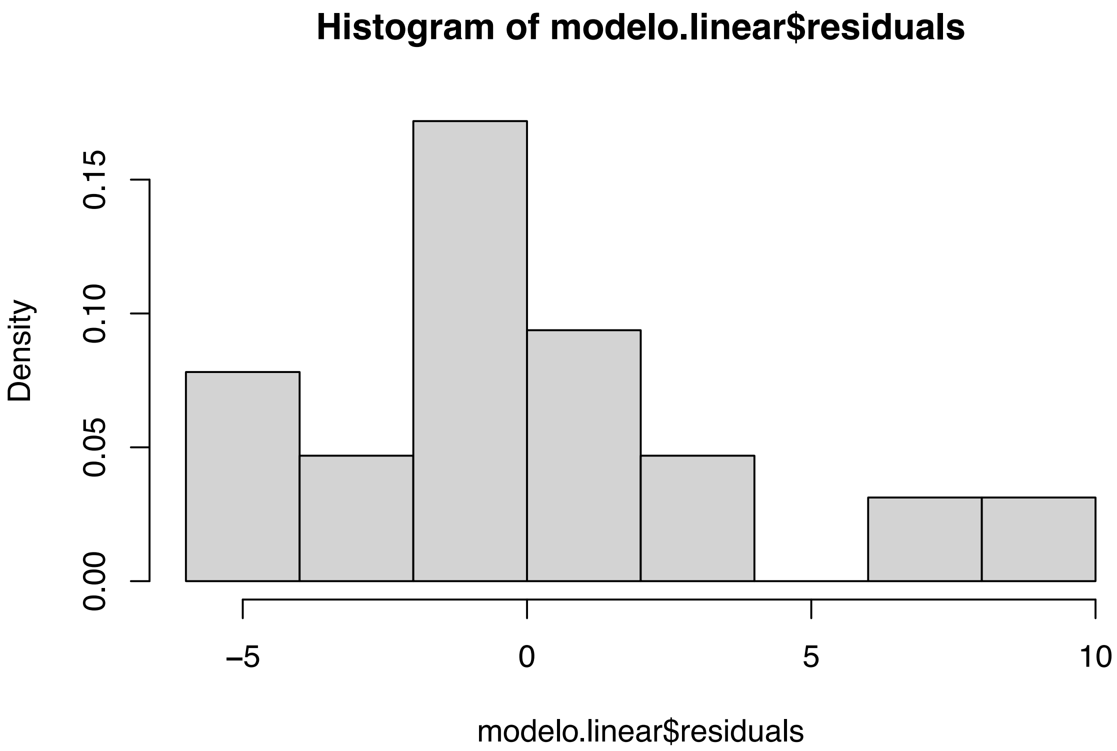
```
## [1] 0.5891853
```

Histograma dos Desvios

```r
shapiro.test(modelo.linear$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelo.linear$residuals
## W = 0.92337, p-value = 0.02568
```

```r
hist(modelo.linear$residuals,freq = F)
```



Agora vamos propositalmente utilizar o modelo e fazer algumas predições e extrapolações.

Para prever valores temos nossa função `predict(modelo, newdata = novos)`

Preste atenção pois o campo newdata **somente** aceita Data Frame

```
#novos valores de HP para a predição
novos_hp <- seq(from = 50 , to = 550 , by = 50)
#predicao
predicao <- predict(modelo.linear,data.frame(hp=novos_hp) )
#exibir os valores
predicao
```

```
##          1          2          3          4          5          6          7
## 26.6874466 23.2760327 19.8646188 16.4532049 13.0417910  9.6303771  6.2189632
##          8          9         10         11
##  2.8075493 -0.6038646 -4.0152785 -7.4266924
```

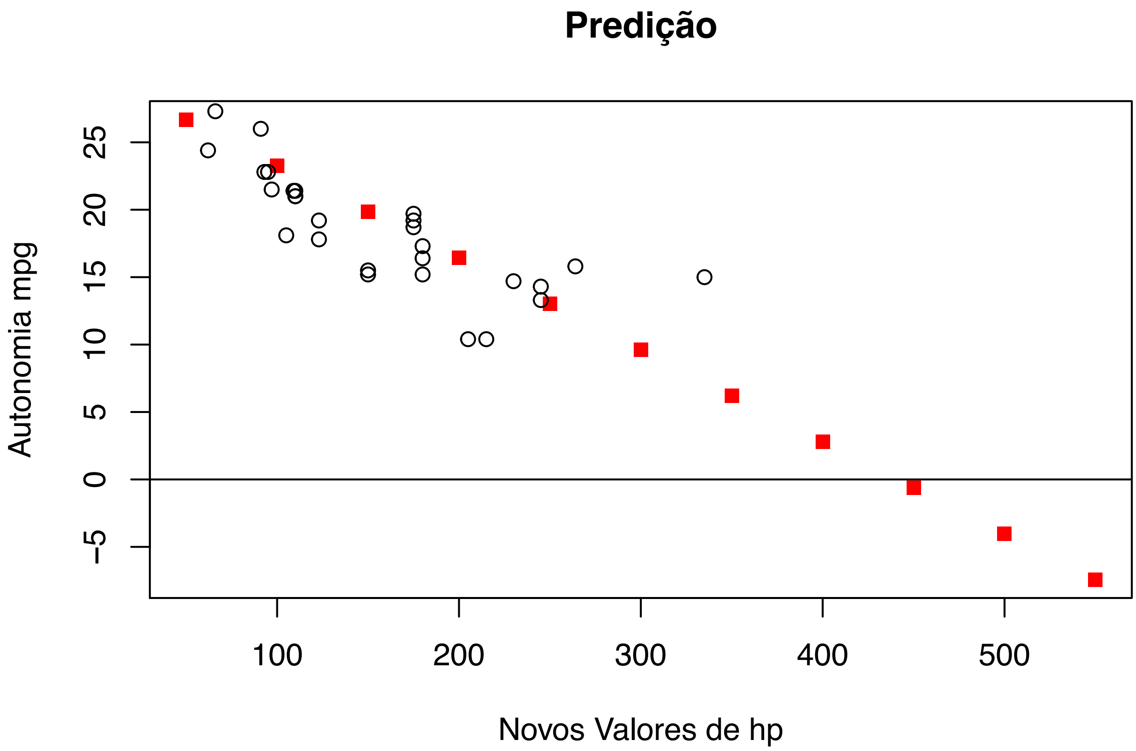
Como já mencionado, extrapolações além dos dados podem gerar problemas, ainda mais quando o problema não é bem conhecido ou o modelo foi escolhido de forma inadequada.

Como podem ver, apartir do ponto i=9, a quantidade de milhas por galão fica negativa. O que seria isso? Produção de combustível?

Isto é mais perceptível no gráfico.

```
#gráfico com os estimados
plot(predicao ~ novos_hp, col = "red", pch = 15,
     ylab = "Autonomia mpg", xlab = "Novos Valores de hp",
     main = "Predição")
#dados observados
points(mpg~hp)
#uma linha horizontal em 0
```

```r
abline(h=0)
```



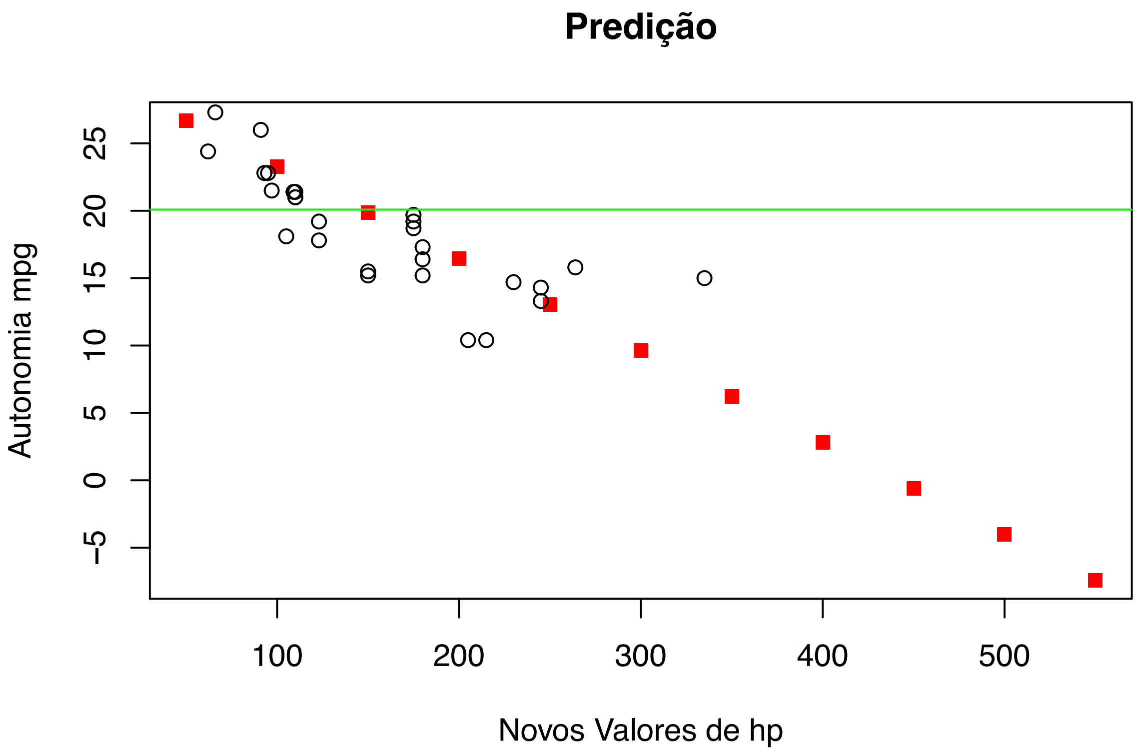
Mas esta estimativa é melhor que a média?

```r
#calcula a média.
media.mpg <- mean(mtcars$mpg)
#faz a modelagem, utiliza-se apenas um paramentro, somente o interceptor
modelo.const <- lm(mpg ~ 1, data = mtcars )
summary(modelo.const)
```

```
##
## Call:
## lm(formula = mpg ~ 1, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.6906 -4.6656 -0.8906  2.7094 13.8094
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   20.091      1.065   18.86   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.027 on 31 degrees of freedom
```

```r
#visualizar no gráfico
#gráfico com os estimados
plot(predicao ~ novos_hp, col = "red", pch = 15,
     ylab = "Autonomia mpg", xlab = "Novos Valores de hp",
     main = "Predição")
```

```r
#dados observados
points(mpg~hp)
#linha da média
abline(modelo.const, col = "green")
```



Ambos os modelos tem relação com os dados como visto no gráfico, mas qual melhor? Há outra forma de comparar os modelos por meio do anova.

```r
anova(modelo.const, modelo.linear,test = "Chisq")
```

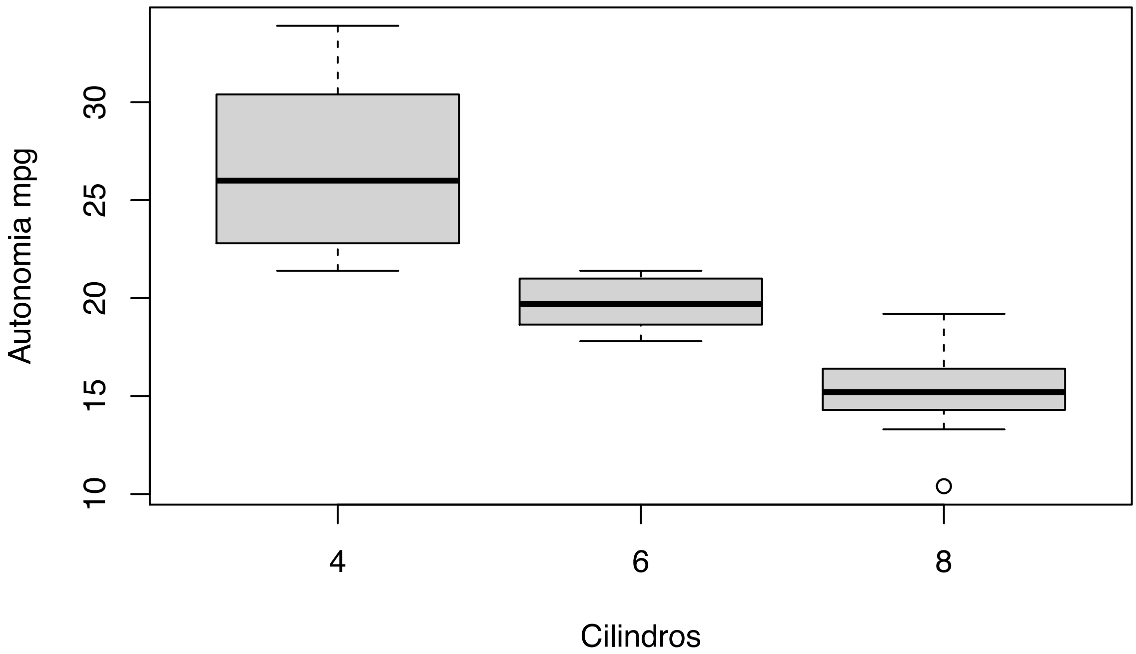
```
## Analysis of Variance Table
##
## Model 1: mpg ~ 1
## Model 2: mpg ~ hp
##   Res.Df     RSS Df Sum of Sq  Pr(>Chi)
## 1     31 1126.05
## 2     30  447.67  1    678.37 1.558e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Portanto pelo p-value obtido na comparação, rejeitamos H0, portanto os modelos realmente diferem. Então vamos escolher o que possui o menor erro.

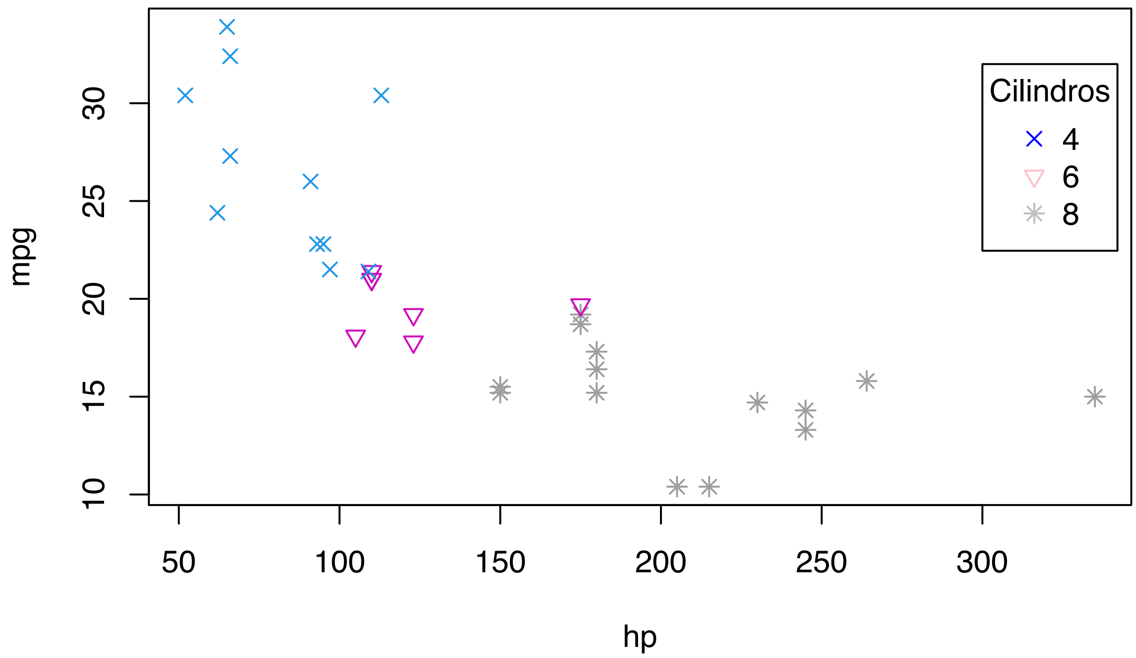
Regressão Linear Multivariada

Mas como visto, nosso exemplo ainda não está bom. Existem outras formas de melhorar. Para isso ao observar melhor nossos dados vemos que ele possui outras colunas, uma delas é cyl que é a quantidade de cilindros nos motores dos carros.

```r
boxplot(mpg ~ cyl, ylab="Autonomia mpg", xlab="Cilindros", data = mtcars)
```




```
plot(mpg ~ hp , pch =cyl, col = cyl, data = mtcars)
legend(300,32,legend = c("4","6","8"),
       pch = c(4,6,8),  col = c("blue","pink","grey"),
       title = "Cilindros")
```



Como pode perceber, parece ter relação entre o cyl e mpg, pois quanto maior , cyl menor mpg

podemos fazer a correlação de várias variaveis ao mesmo tempo com `cor()`

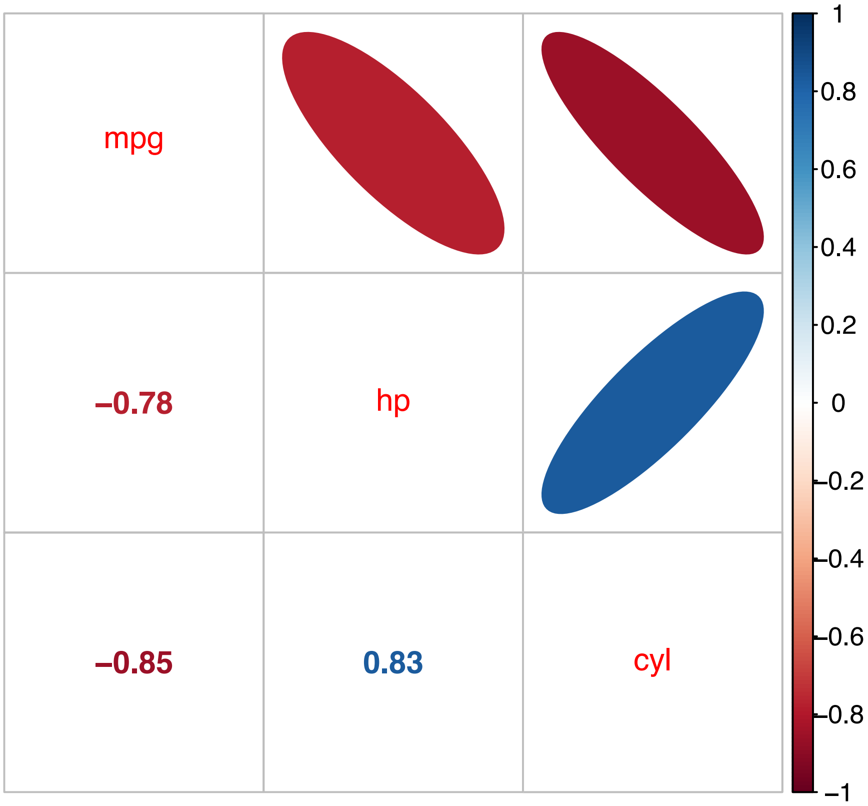
```
#verificando a correlação
multi.cor <- cor(mtcars[,c("mpg","hp","cyl")], method = "pearson")
#correlação base
multi.cor
```

```
##             mpg         hp        cyl
## mpg  1.0000000 -0.7761684 -0.8521620
## hp  -0.7761684  1.0000000  0.8324475
## cyl -0.8521620  0.8324475  1.0000000
```

```r
#install.packages("corrplot")

#aqui está uma forma de exibir a correlação
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
corrplot.mixed(multi.cor, upper = "ellipse")
```



Então vamos fazer uma regressão multivariada.

Da mesma forma que a regressão linear simples, utilizamos `lm()` mas agora dicionamos mais uma variável na fórmula.

A nova váriavel cyl tem uma correlação ainda mais forte com a autonomia Então podemos utilizar uma regressão linear multivariada.

```r
#regressão linear multivariada
modelo.multi <- lm(mpg ~ hp + cyl, data = mtcars)

#aqui temos nossa regressão depois vamos melhorar isso. Por hora apenas vamos deixar assim.
summary(modelo.multi)
```

```
##
## Call:
## lm(formula = mpg ~ hp + cyl, data = mtcars)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4948 -2.4901 -0.1828  1.9777  7.2934
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.90833    2.19080  16.847  < 2e-16 ***
## hp          -0.01912    0.01500  -1.275  0.21253
## cyl         -2.26469    0.57589  -3.933  0.00048 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.173 on 29 degrees of freedom
## Multiple R-squared:  0.7407, Adjusted R-squared:  0.7228
## F-statistic: 41.42 on 2 and 29 DF,  p-value: 3.162e-09
```

Vamos analisar o coeficiente de determinação **ajustado**, neste caso devemos sempre observar o coeficiente de determinação Ajustado que minimiza o aumento provocado apenas por se incluir uma nova variável ao modelo

```r
#r^2 do modelo linear
summary(modelo.linear)$adj.r.squared
```

```
## [1] 0.5891853
```

```r
#r^2 do modelo multi variado
summary(modelo.multi)$adj.r.squared
```

```
## [1] 0.7228263
```

```r
#coeficiente de determinacao simples
summary(modelo.linear)$r.squared
```

```
## [1] 0.6024373
```

```r
#aqui é só para ver como sempre há um aumento maior no coeficiente de determinacao e que devemos utiliz
summary(modelo.multi)$r.squared
```

```
## [1] 0.7407084
```

E os coeficientes?

Da mesma maneira que a regressão linear simples basta consultar

```r
coefficients(modelo.multi)
```

```
## (Intercept)          hp         cyl
##  36.9083305  -0.0191217  -2.2646936
```

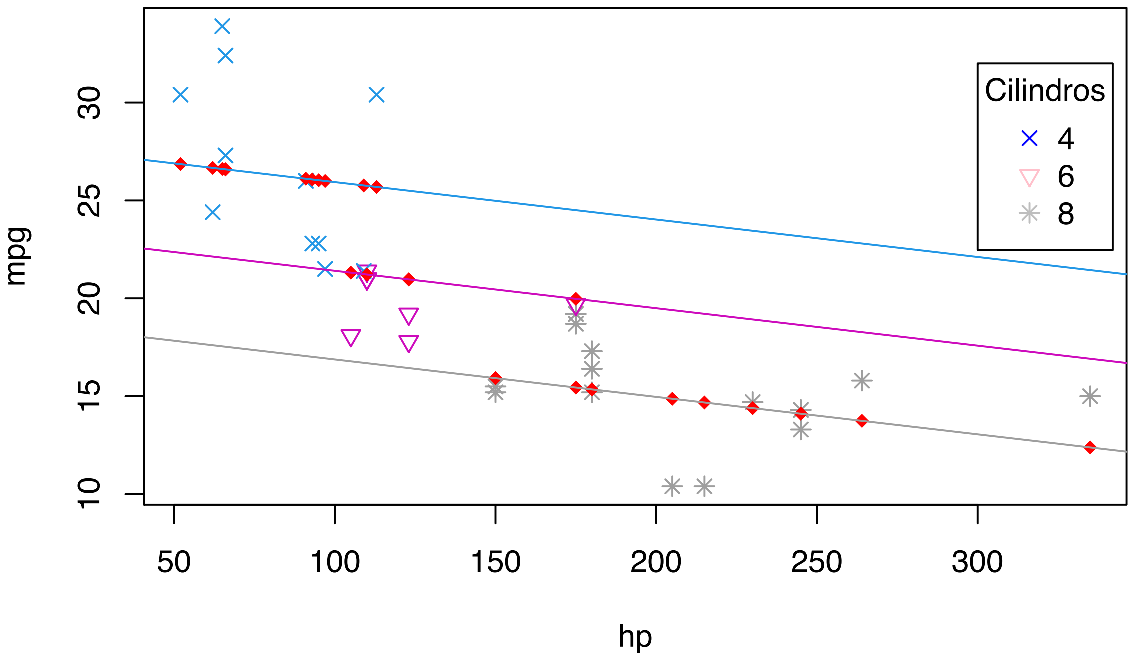
Temos nossa reta Y = 36.9083305 + -0.0191217 * hp + -0.0191217 * cyl

Como pode perceber, o erro diminuiu e a nossa regressão multivariada representa melhor nossos dados

Quero ver, Mas e os gráficos?

```r
plot(mpg ~ hp , pch =cyl, col = cyl, data = mtcars)

points(modelo.multi$fitted.values ~ hp, col = "red", pch = 18, data = mtcars)
fun.reta <-  function(x, cyl){
  modelo.multi$coefficients[1]+
```

```r
    modelo.multi$coefficients["cyl"]*cyl +
    modelo.multi$coefficients["hp"]*x
  }
curve(expr = fun.reta(x,4)  , from = 0, to = 500, add = T, col = 4)
curve(expr = fun.reta(x,6)  , from = 0, to = 500, add = T, col = 6)
curve(expr = fun.reta(x,8)  , from = 0, to = 500, add = T, col = 8)
legend(300,32,legend = c("4","6","8"),
       pch = c(4,6,8),  col = c("blue","pink","grey"),
       title = "Cilindros")
```




```r
#Intervalo de confianca
intervalo_confianca <- confint.default(modelo.multi, level = 0.95)
cbind(intervalo_confianca,modelo.multi$coefficients)
```

```
##                    2.5 %      97.5 %
## (Intercept) 32.61444405 41.20221691 36.9083305
## hp          -0.04852259  0.01027919 -0.0191217
## cyl         -3.39341577 -1.13597142 -2.2646936
```

Parece melhor. Mas será que difere muito dos outros modelos?

```r
anova(modelo.multi,test = "Chisq")
```

```
## Analysis of Variance Table
##
## Response: mpg
##           Df Sum Sq Mean Sq F value    Pr(>F)
## hp         1 678.37  678.37  67.379 4.742e-09 ***
## cyl        1 155.70  155.70  15.465 0.0004804 ***
## Residuals 29 291.97   10.07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(modelo.const, modelo.linear, modelo.multi,test = "Chisq")
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ 1
## Model 2: mpg ~ hp
## Model 3: mpg ~ hp + cyl
##   Res.Df     RSS Df Sum of Sq  Pr(>Chi)
## 1     31 1126.05
## 2     30  447.67  1    678.37 2.241e-16 ***
## 3     29  291.97  1    155.70 8.406e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Podese rejeitar H0, portanto os modelos são diferentes e devese escolher o melhor. Mas qual é o melhor? O mais simples ou com o menor erro?

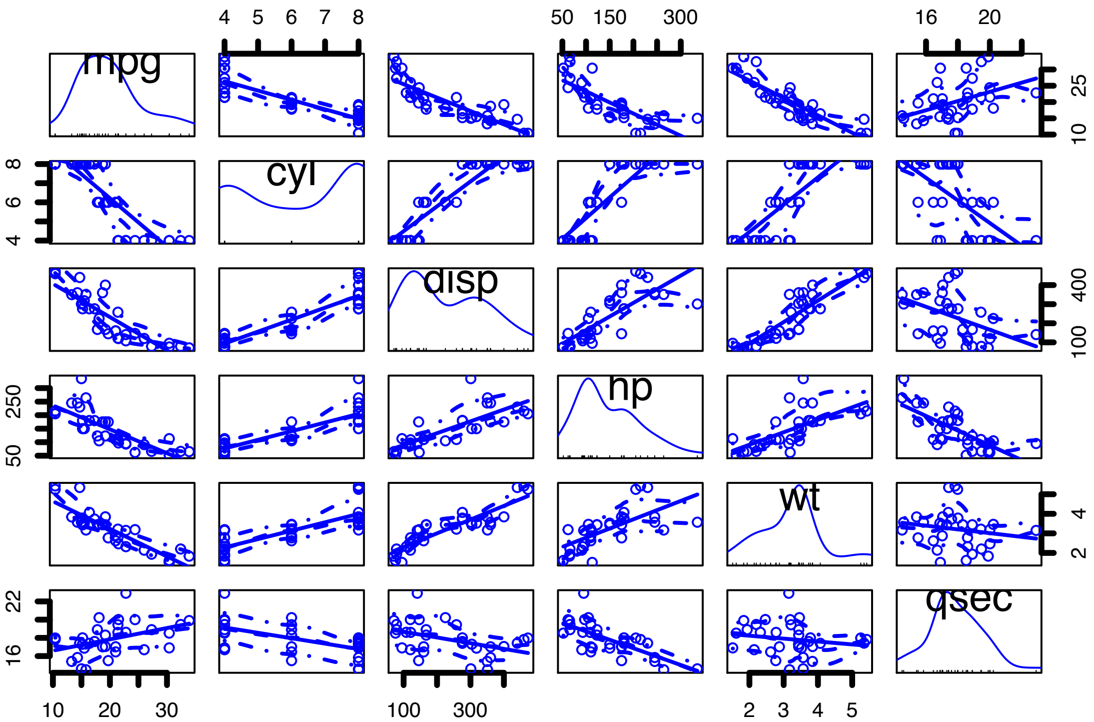
Interprete o resultado obtido.

Seleção de variáveis

Backward

No modo de seleção Backward, adiciona-se todas as variáveis e vai retirando as menos significativas. Para efeito didático vamos trabalhar apenas com algumas das colunas do mtcars

```r
#install.packages("car")
library(car)
scatterplotMatrix(mtcars[ , c("mpg" , "cyl" , "disp","hp" , "wt" ,"qsec")], lwd = 3)
```




```r
#listando as colunas para escolher as que interessam
colnames(mtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear"
## [11] "carb"
```

```r
#Escolha um grau de significancia

#modelo linear multi variado
modelo.multi2.0 <- lm(mpg ~ cyl + disp + hp + wt + qsec, data = mtcars)

#observa-se a importancia das variáveis no modelo
summary(modelo.multi2.0)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + disp + hp + wt + qsec, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3117 -1.3483 -0.4352  1.2603  5.6094
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 35.87361    9.91809   3.617  0.00126 **
## cyl         -1.15608    0.71525  -1.616  0.11809
## disp         0.01195    0.01191   1.004  0.32484
## hp          -0.01584    0.01527  -1.037  0.30908
## wt          -4.22527    1.25239  -3.374  0.00233 **
## qsec         0.25382    0.48746   0.521  0.60699
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.547 on 26 degrees of freedom
## Multiple R-squared:  0.8502, Adjusted R-squared:  0.8214
## F-statistic: 29.51 on 5 and 26 DF,  p-value: 6.182e-10
```

```r
variavel_menor_backward <- function(modelo, dados, coef = 0.975){
  #modelo <- modelo.multi2
  #summary(modelo)
  #coef <- 0.975
  #dados<- mtcars

  #obtém-se os graus de liberdade
  modelo.coef <- summary(modelo)$coefficients;modelo.coef
  if(nrow(summary(modelo)$coefficients) == 2 ){
    return(sprintf("Só há uma variável, nada a remover"))
  }
  gl<-nrow(dados) - length(coefficients(modelo));gl

  #calcula o quantil
  quantile  <- qt(coef,gl);quantile

  modelo.coef <- modelo.coef[-1,];modelo.coef
  #menor_valor <- min (abs(summary(modelo)$coefficients[, "t value"])) ; menor_valor
  menor_valor <- min (abs(modelo.coef[, "t value"])); menor_valor

  #obtém os valores que são menor que o quantil
```

```r
  nome <- names(which(abs(modelo.coef[,"t value"]) == menor_valor))

  if (quantile > menor_valor) {
    return(sprintf("Remova %s, t %f  < quantil: %f", nome,menor_valor,quantile))
  }
  return(sprintf("Remova %s, t %f  < quantil: %f", nome,menor_valor,quantile))
}

variavel_menor_backward(modelo.multi2.0, mtcars)
```

```
## [1] "Remova qsec, t 0.520695  < quantil: 2.055529"
```

```r
#nova formula sem qsec
modelo.multi2.1 <- lm(mpg ~ cyl + disp + hp + wt , data = mtcars)
summary(modelo.multi2.1)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + disp + hp + wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0562 -1.4636 -0.4281  1.2854  5.8269
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40.82854    2.75747  14.807 1.76e-14 ***
## cyl         -1.29332    0.65588  -1.972 0.058947 .
## disp         0.01160    0.01173   0.989 0.331386
## hp          -0.02054    0.01215  -1.691 0.102379
## wt          -3.85390    1.01547  -3.795 0.000759 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.513 on 27 degrees of freedom
## Multiple R-squared:  0.8486, Adjusted R-squared:  0.8262
## F-statistic: 37.84 on 4 and 27 DF,  p-value: 1.061e-10
```

```r
variavel_menor_backward(modelo.multi2.1, mtcars)
```

```
## [1] "Remova disp, t 0.989122  < quantil: 2.051831"
```

```r
#nova formula sem disp
modelo.multi2.2 <- lm(mpg ~ cyl + hp + wt , data = mtcars)
summary(modelo.multi2.2)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + hp + wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9290 -1.5598 -0.5311  1.1850  5.8986
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 38.75179    1.78686  21.687  < 2e-16 ***
## cyl         -0.94162    0.55092  -1.709 0.098480 .
## hp          -0.01804    0.01188  -1.519 0.140015
## wt          -3.16697    0.74058  -4.276 0.000199 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.512 on 28 degrees of freedom
## Multiple R-squared:  0.8431, Adjusted R-squared:  0.8263
## F-statistic: 50.17 on 3 and 28 DF,  p-value: 2.184e-11
```

```r
variavel_menor_backward(modelo.multi2.2, mtcars)
```

```
## [1] "Remova hp, t 1.518838  < quantil: 2.048407"
```

```r
#nova formula sem hp
modelo.multi2.3 <- lm(mpg ~ cyl + wt , data = mtcars)
summary(modelo.multi2.3)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2893 -1.5512 -0.4684  1.5743  6.1004
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.6863     1.7150  23.141  < 2e-16 ***
## cyl          -1.5078     0.4147  -3.636 0.001064 **
## wt           -3.1910     0.7569  -4.216 0.000222 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.568 on 29 degrees of freedom
## Multiple R-squared:  0.8302, Adjusted R-squared:  0.8185
## F-statistic: 70.91 on 2 and 29 DF,  p-value: 6.809e-12
```

```r
variavel_menor_backward(modelo.multi2.3, mtcars)
```

```
## [1] "Remova cyl, t 3.635972  < quantil: 2.045230"
```

```r
#nova formula sem cyl
modelo.multi2.4 <- lm(mpg ~  wt , data = mtcars)
summary(modelo.multi2.4)
```

```
##
## 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
```

```
variavel_menor_backward(modelo.multi2.4, mtcars)
```

```
## [1] "Só há uma variável, nada a remover"
```

```
#falar sobre covarianca
vcov(modelo.multi2.0)
```

```
##             (Intercept)          cyl          disp            hp           wt
## (Intercept) 98.36851673 -3.780794108  0.0188985496 -0.0838208307  4.959981095
## cyl         -3.78079411  0.511577454 -0.0041079682 -0.0012510199 -0.169858483
## disp         0.01889855 -0.004107968  0.0001417866 -0.0000243896 -0.008850177
## hp          -0.08382083 -0.001251020 -0.0000243896  0.0002330882 -0.004825560
## wt           4.95998110 -0.169858483 -0.0088501773 -0.0048255598  1.568470147
## qsec        -4.63866916  0.128479190  0.0003281529  0.0043993353 -0.347663758
##                     qsec
## (Intercept) -4.6386691643
## cyl          0.1284791895
## disp         0.0003281529
## hp           0.0043993353
## wt          -0.3476637582
## qsec         0.2376180891
```

```
vcov(modelo.multi2.3)
```

```
##             (Intercept)        cyl         wt
## (Intercept)   2.9411702 -0.2738532 -0.3234747
## cyl          -0.2738532  0.1719664 -0.2456100
## wt           -0.3234747 -0.2456100  0.5729074
```

```
#analisando a diferença entre os modelos
anova(modelo.multi2.0, modelo.multi2.1, modelo.multi2.2, modelo.multi2.3, modelo.multi2.4,test = "Chisq
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ cyl + disp + hp + wt + qsec
## Model 2: mpg ~ cyl + disp + hp + wt
## Model 3: mpg ~ cyl + hp + wt
## Model 4: mpg ~ cyl + wt
## Model 5: mpg ~ wt
##   Res.Df    RSS Df Sum of Sq  Pr(>Chi)
## 1     26 168.69
## 2     27 170.44 -1    -1.759 0.6025793
## 3     28 176.62 -1    -6.176 0.3292234
## 4     29 191.17 -1   -14.551 0.1342325
## 5     30 278.32 -1   -87.150 0.0002473 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note que o último modelo difere dos outros modelos.

Vamos observar melhor

```
library(hnp)
```

```
## Loading required package: MASS
```

```
layout(matrix(c(1,2,3,4),2,2))
#Grafico Normal das Probabilidades dos modelos
hnp(modelo.multi2.0, xlab = 'N(0,1)', ylab = 'Resíduos', main = modelo.multi2.0$call$formula)
```

```
## Gaussian model (lm object)
```

```
hnp(modelo.multi2.1, xlab = 'N(0,1)', ylab = 'Resíduos', main = modelo.multi2.1$call$formula)
```

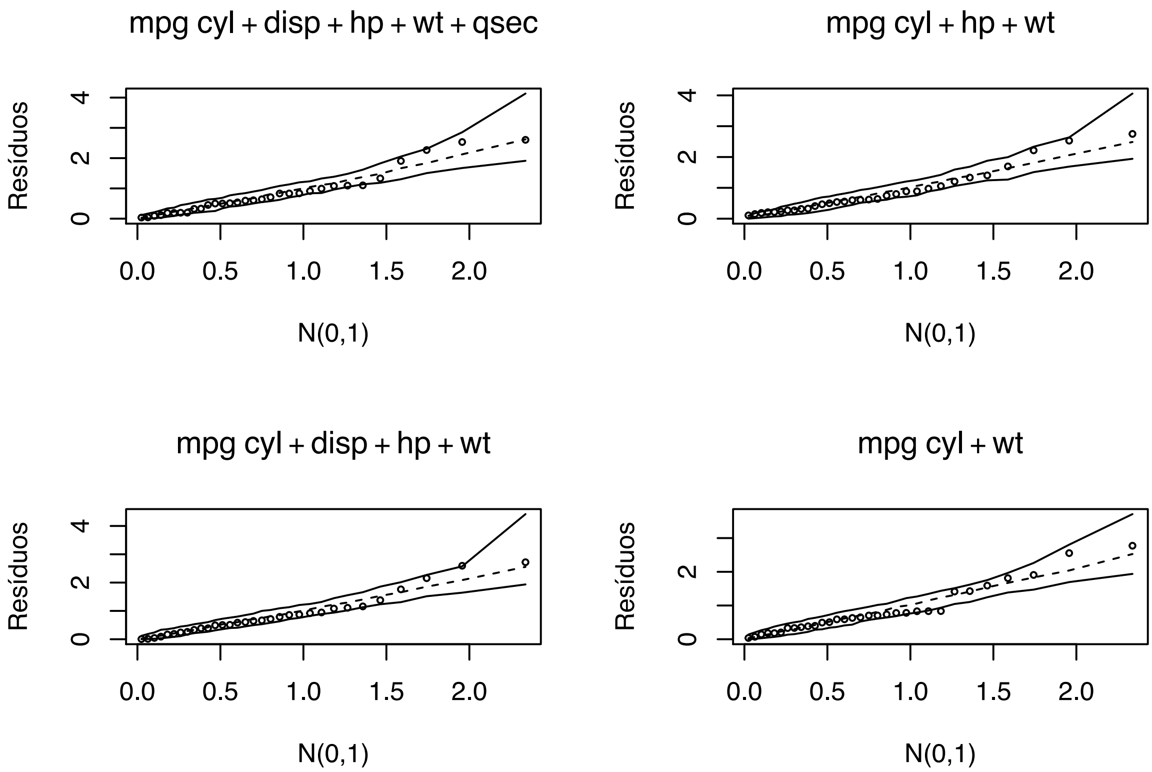
```
## Gaussian model (lm object)
```

```
hnp(modelo.multi2.2, xlab = 'N(0,1)', ylab = 'Resíduos', main = modelo.multi2.2$call$formula)
```

```
## Gaussian model (lm object)
```

```
hnp(modelo.multi2.3, xlab = 'N(0,1)', ylab = 'Resíduos', main = modelo.multi2.3$call$formula)
```

```
## Gaussian model (lm object)
```



Confirmou-se a menor falta de aderência do modelo 2.3.

```
data.frame(
modelos = c('2.0', '2.1', '2.2', '2.3'),
aic = c(AIC(modelo.multi2.0), AIC(modelo.multi2.1),AIC(modelo.multi2.2),AIC(modelo.multi2.3)),
verossimilhança = c(logLik(modelo.multi2.0), logLik(modelo.multi2.1),logLik(modelo.multi2.2),logLik(mod
)
```

```
##   modelos      aic verossimilhança
## 1     2.0 158.0056       -72.00282
```

```
## 2      2.1 156.3376       -72.16880
## 3      2.2 155.4766       -72.73831
## 4      2.3 156.0101       -74.00503
```

Aqui temos que procurar o modelo com menor AIC e maior Verossimilhança.

Muito difícil, não tem nada mais fácil? Tem sim =)

Mas para isso vamos precisar da biblioteca MASS que não vem instalada por padrão. Vamos começar com o modelo 2.0 e ver os resultados.

```r
library(MASS)
modelo.multi.step <- step(modelo.multi2.0, direction = "both")
```

```
## Start:  AIC=65.19
## mpg ~ cyl + disp + hp + wt + qsec
##
##        Df Sum of Sq    RSS    AIC
## - qsec  1     1.759 170.44 63.526
## - disp  1     6.534 175.22 64.410
## - hp    1     6.983 175.67 64.492
## <none>              168.69 65.194
## - cyl   1    16.950 185.63 66.258
## - wt    1    73.848 242.53 74.813
##
## Step:  AIC=63.53
## mpg ~ cyl + disp + hp + wt
##
##        Df Sum of Sq    RSS    AIC
## - disp  1     6.176 176.62 62.665
## <none>              170.44 63.526
## - hp    1    18.048 188.49 64.746
## + qsec  1     1.759 168.69 65.194
## - cyl   1    24.546 194.99 65.831
## - wt    1    90.925 261.37 75.206
##
## Step:  AIC=62.66
## mpg ~ cyl + hp + wt
##
##        Df Sum of Sq    RSS    AIC
## <none>              176.62 62.665
## - hp    1    14.551 191.17 63.198
## + disp  1     6.176 170.44 63.526
## - cyl   1    18.427 195.05 63.840
## + qsec  1     1.401 175.22 64.410
## - wt    1   115.354 291.98 76.750
```

```r
anova(modelo.multi2.0, modelo.multi.step)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ cyl + disp + hp + wt + qsec
## Model 2: mpg ~ cyl + hp + wt
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     26 168.69
## 2     28 176.62 -2   -7.9352 0.6115 0.5501
```

Forward

Ao contrário do Backward começa pelo modelo mais simples e começa a adicionar variáveis Basta usar `step(modelo, direction = "forward")`

StepWise

Utiliza-se o Forward para adicionar uma variável ao modelo e o backward para ver se houve alteração de importancia para as outras variáveis a cada inserção. Pode-se utilizar intervalos diferentes para adição e remoção de variáveis.

Basta usar `step(modelo, direction = "both")`

Alavancagem

O que é alavancagem?

Pode-se entender alavancagem como a influência de cada dado/observação sobre o modelo.

ela pode ser obtida com a funcção `influence(modelo)`

Com isso fica fácil identificar algum valor que possa ser um outlier.

Mas muito cuidado ao remover os valores

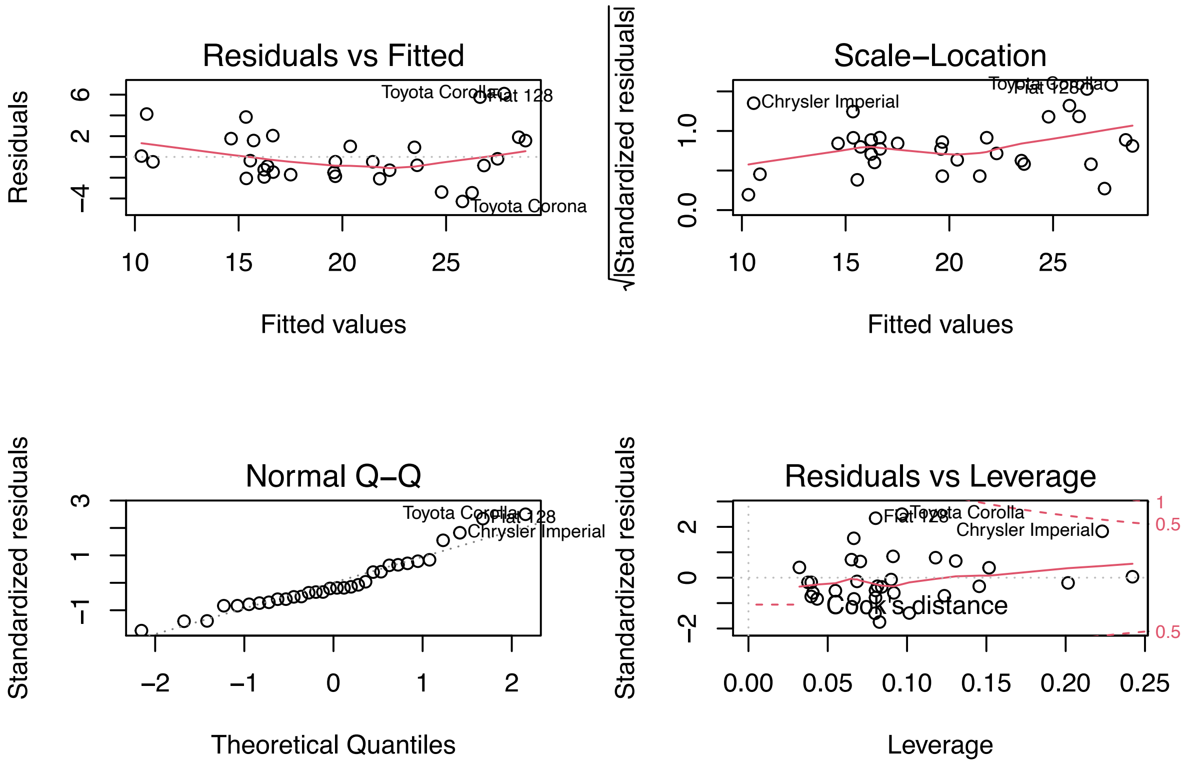
```
sort(influence(modelo.multi2.3)$hat)
```

```
##      Hornet 4 Drive       Mazda RX4 Wag            Merc 280           Merc 280C 
##          0.03213611          0.03756521          0.03959146          0.03959146 
##             Valiant        Ferrari Dino           Mazda RX4          Merc 450SE 
##          0.04068001          0.04330261          0.05482311          0.06497359 
##    Pontiac Firebird          Camaro Z28         Merc 450SLC          Merc 450SL 
##          0.06641213          0.06654404          0.06846591          0.07054547 
##          Datsun 710          Duster 360       Maserati Bora            Fiat 128 
##          0.07978891          0.08012014          0.08012014          0.08019462 
##       Porsche 914-2       Toyota Corona    Dodge Challenger           Fiat X1-9 
##          0.08133608          0.08263810          0.08402476          0.08995733 
##    Hornet Sportabout         AMC Javelin      Toyota Corolla          Volvo 142E 
##          0.09117599          0.09165987          0.09681350          0.10142065 
##         Honda Civic      Ford Pantera L        Lotus Europa            Merc 230 
##          0.11801538          0.12352416          0.13069974          0.14550945 
##           Merc 240D  Cadillac Fleetwood   Chrysler Imperial Lincoln Continental 
##          0.15170109          0.20151356          0.22303287          0.24212252
```

```
#A soma das alavancagens é igual ao número de variáveis
sum(influence(modelo.multi2.3)$hat)
```

```
## [1] 3
```

```
layout(matrix(c(1,2,3,4),2,2))
plot(modelo.multi2.3)
```



Regressões GLM

São Modelos Lineares Generalizados. A função para se obter esses modelos é 'glm()

Regressão Binomial

Utiliza a sigmóide para fazer regressão de probabilidade. Analisa casos onde a variável resposta é lógica (0 ou 1) ou a probilidade de que um evento ocorra.

Neste exemplo vamos fazer a análise de dose e resposta. Para fazer a regressão de insetos mortos por dose de inseticida. Análise de resposta.

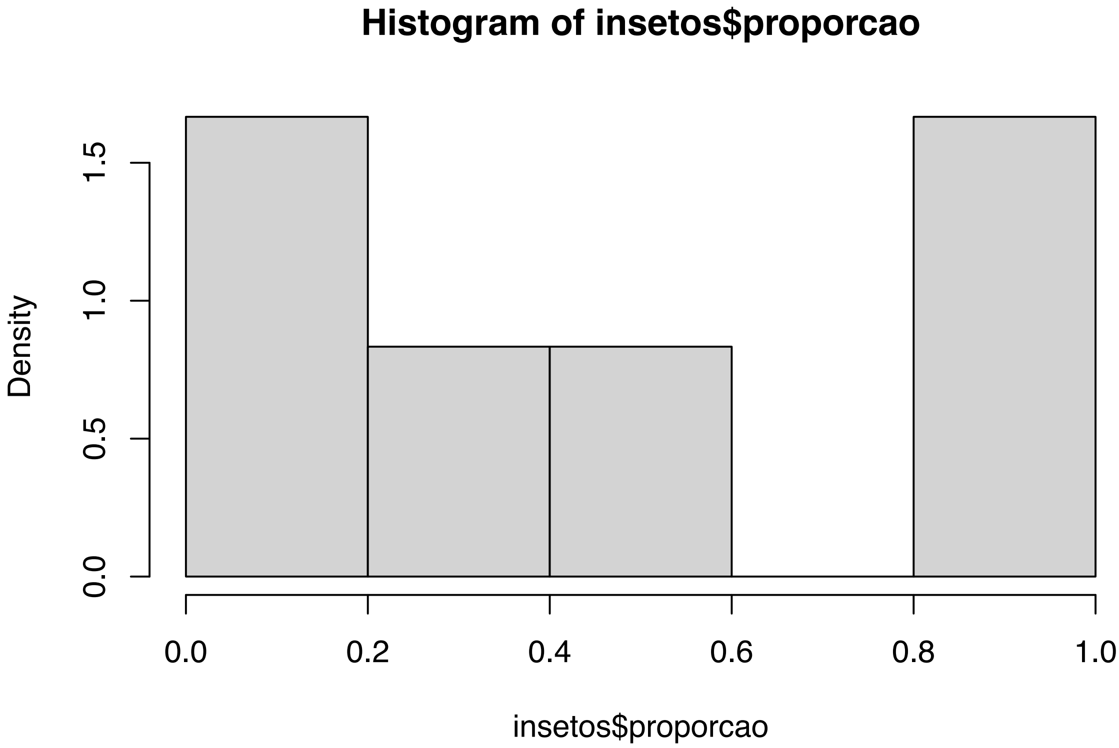
```
#Análise de insetos mortos

#criando o data frame
insetos <- data.frame(
  dose   = c(0.0, 2.6, 3.8, 5.1, 7.7, 10.2),
  total  = c(49,  50,  48,  46,  49,  50),
  mortos = c(0,   6,   16,  24,  42,  44))

#Nossa variável resposta
insetos$proporcao <- NULL
insetos$proporcao <- insetos$mortos / insetos$total
#insetos <- cbind(insetos, proporcao)

#limpando
head(insetos)
```

```
##   dose total mortos proporcao
## 1  0.0    49      0 0.0000000
```

```
## 2  2.6    50      6 0.1200000
## 3  3.8    48     16 0.3333333
## 4  5.1    46     24 0.5217391
## 5  7.7    49     42 0.8571429
## 6 10.2    50     44 0.8800000
```

Agora vamos ver o comportamento da nossa variável depensente

```r
#Olhar o comportamentomento do nosso Y
hist(insetos$proporcao, freq = F)
```



Como demonstrado no gráfico, mesmo com poucas amostras, pode-se perceber uma grande concentração nas extremidades. Além disso nossa resposta tem os limites em 0 e 1.

Agora vamos ver a dispersão

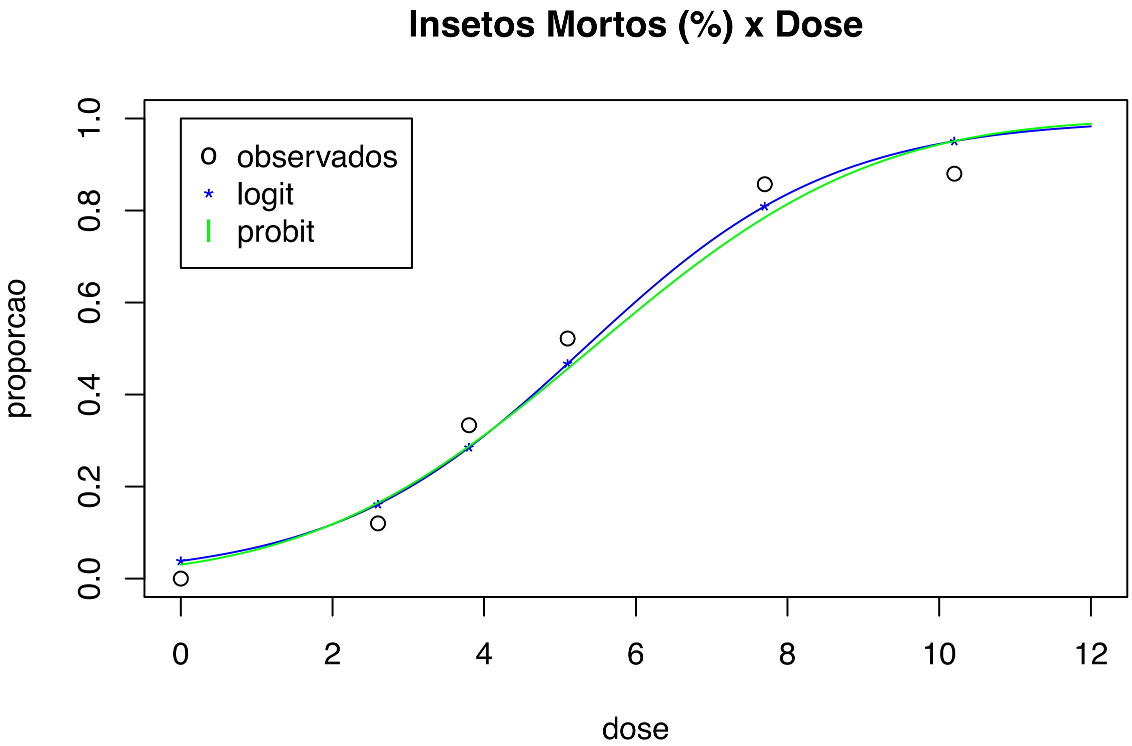
```r
# Plotando o gráfico da variável resposta por dose
plot(proporcao ~ dose, data = insetos , xlim = c(0,12), ylim = c(0,1), main = "Insetos Mortos (%) x Dos
```

Neste caso a regressão linear não é adequada. Iremos utilizar a regressão glm da família binomial. `glm(family = binomial )`

```
# Fazendo a regressão do modelo binomial
modelo.logit <- glm(proporcao ~ dose, family = binomial(link = "logit"), data = insetos)
# Utilizar outra função de ligação probit
modelo.probit <- glm(proporcao ~ dose, family = binomial(link = "probit"), data = insetos)

#aqui os coeficientes de beta
coef(modelo.logit)
```

```
## (Intercept)        dose
##  -3.2232221   0.6062865
```

```
#novas dados para estimar a proporção de insetos mortos
new.doses <- seq(0,12, by=0.1)

#estimando os valores
new.prob.logit <- predict(modelo.logit,data.frame(dose=new.doses))
new.prob.probit <- predict(modelo.probit,data.frame(dose=new.doses), type = "response")

#funcao inversa
mu <- function(t) {exp(t)/(1+exp(t))}

#vendo os gráficos
plot(proporcao ~ dose, data = insetos , xlim = c(0,12), ylim = c(0,1), main = "Insetos Mortos (%) x Dos
lines(new.doses, mu(new.prob.logit),  col = "blue")
points(insetos$dose, modelo.logit$fitted.values, pch = "*", col = "blue")
lines(new.doses, new.prob.probit, col = "green")
legend(0,1, legend = c("observados","logit","probit"),
       col = c("black","blue","green"), pch=c("o","*","line")  )
```




```r
#comparando os modelos
anova(modelo.logit, modelo.probit , test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: proporcao ~ dose
## Model 2: proporcao ~ dose
##   Resid. Df Resid. Dev Df  Deviance Pr(>Chi)
## 1         4    0.20850
## 2         4    0.21687  0 -0.008366
```

```r
exp(modelo.logit$coefficients[2])
```

```
##    dose
## 1.83361
```

```r
exp(confint.default(modelo.logit))
```

```
##                    2.5 %   97.5 %
## (Intercept) 0.0002468564 6.425405
## dose        0.7189248366 4.676601
```

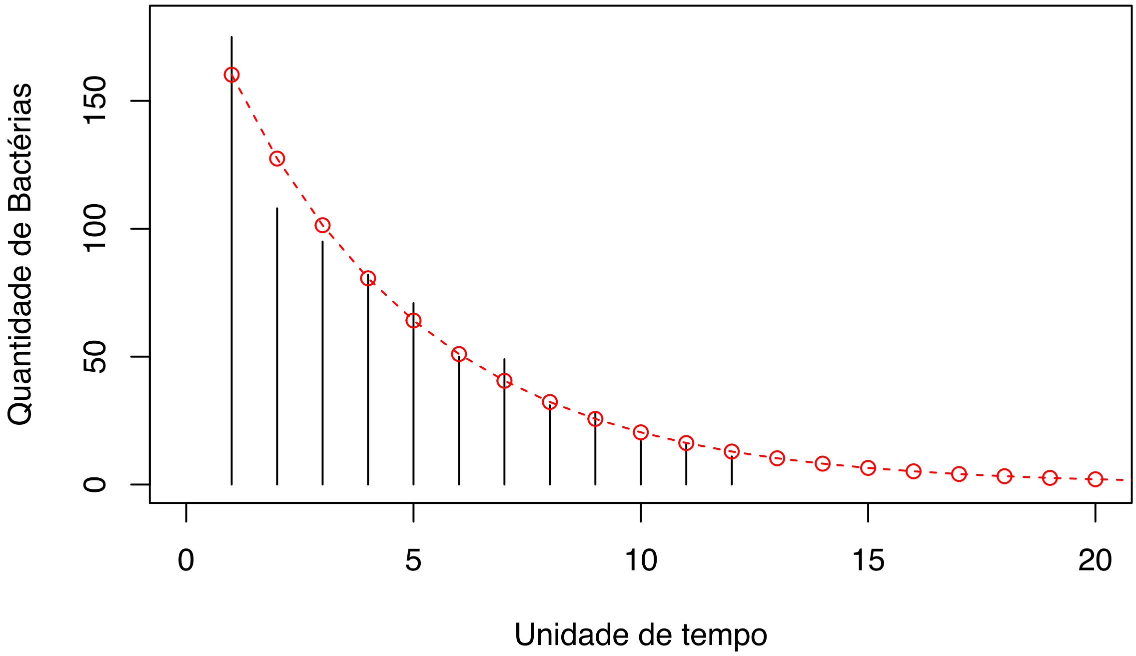
Temos que a cada aumento de 1 unidade de dose o número de insetos mortos aumenta em 83.3609659 %

A probabilidade de acontecer sucesso é nula para valores pequenos e praticamente certa para valores altos de dose.

Poisson

A taxa de ocorrência é constante no tempo Os intervalos são independentes. Um intervalo não pode interferir no outro.

```r
bacterias <- c(175, 108,95,82,71,50,49,31,28,17,16,11)
tempo <- 1:12
plot(bacterias ~ tempo , xlim = c(0,20) ,
     ylim= c(0,180), type = "h",
     ylab = "Quantidade de Bactérias", xlab = "Unidade de tempo")

modelo <- glm(bacterias ~ tempo, family = poisson)
x <- seq(0:20)
preditos <- predict(modelo, data.frame(tempo = x))

f<- function(l){exp(l)}
points(x , f(preditos), col = "red")
lines(x , f(preditos), col = "red", lty = 2)
```




```r
preditos <- predict(modelo, data.frame(tempo = x),
                    type = "response", interval = "prediction")

summary(modelo)
```

```
##
## Call:
## glm(formula = bacterias ~ tempo, family = poisson)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7703  -0.5715  -0.1019   0.5496   1.2794
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.30557    0.06348   83.58   <2e-16 ***
## tempo       -0.22890    0.01270  -18.02   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 393.6292  on 11  degrees of freedom
## Residual deviance:   8.4215  on 10  degrees of freedom
## AIC: 80.182
##
## Number of Fisher Scoring iterations: 4
```

```
int.confianca <- exp(confint.default(modelo))
int.confianca
```

```
##                   2.5 %     97.5 %
## (Intercept) 177.8868390 228.148180
## tempo         0.7758571   0.815459
```

Gama e Gaussiana Inversa

A distribuição Gama pode ser obtida com a funçaão `glm(mpg ~ hp ,family = Gamma, data = mtcars)`

Muito útil para dados positivos e assimétricos.

Lembra dos nossos carros? Então vamos ver como fica.

```
#Direto para o modelo
modelo.g.mpg <- glm(mpg ~ hp ,family = Gamma, data = mtcars)

#Preditor linear multivariado
modelo.g.mpg.2 <- glm(mpg ~ hp + cyl,family = Gamma, data = mtcars)

#Aqui temos a gaussiana inversa, Outra distribuição para dados assimétricos
modelo.ig.mpg <- glm(mpg ~ hp + cyl,family = inverse.gaussian, data = mtcars)
```

Para o modelo linear tínhamos o problema quando prevíamos a autonomia uma potência muito alta. Neste caso tínhamos uma autonomia negativa.

Neste caso vamos prever para hp = 1000 e ver o que acontece.

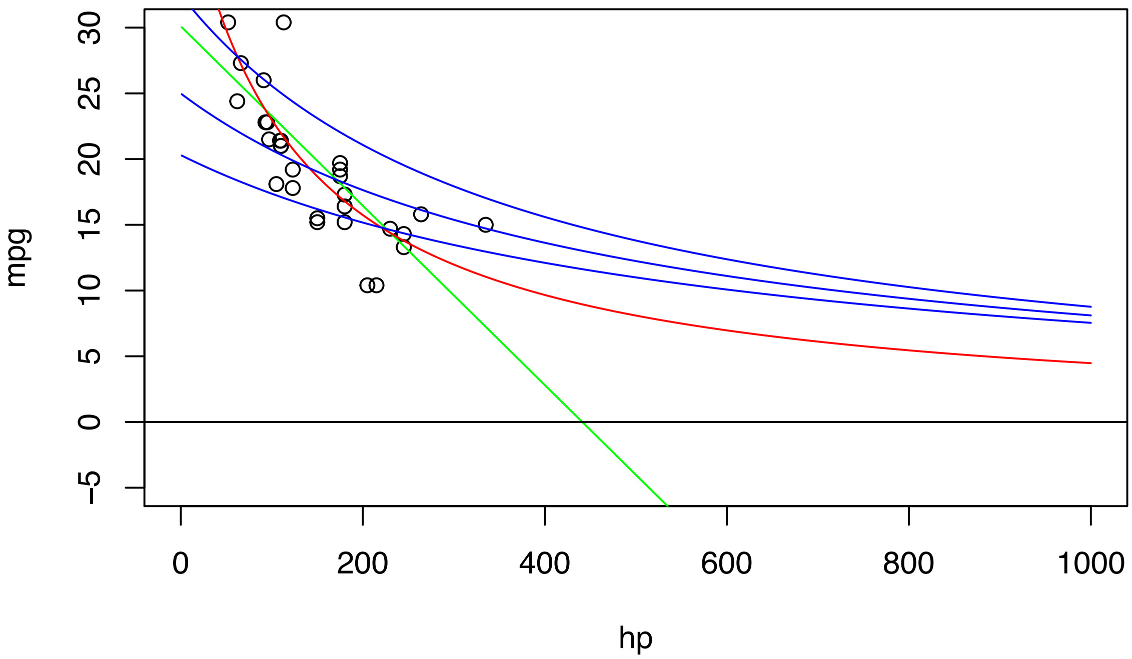
Linear(hp = 1000): -38.1294175 | Gama(hp = 1000): 4.4756265 | Gama(hp = 100000): 0.0499693

Como pode observar, agora mesmo extrapolando o valor dos dados coletados, não temos mais a autonomia negativa.

Os gráficos =)

```
#valores de predição de 1 até 1000
n.hp <- 1:1000

#vamos apra os gráficos
plot(mpg ~ hp , data = mtcars, ylim = c(-5,30), xlim=c(0,1000))
lines(n.hp,
      predict(modelo.linear,
                   newdata = data.frame(hp = n.hp),
                   type = "response"), col = "green")
#points(mtcars$hp,modelo.g.mpg$fitted.values, col = "red", pch = 14)
abline(h = 0)
lines(n.hp,
      predict(modelo.g.mpg,
```

```r
              newdata = data.frame(hp = n.hp),
              type = "response"),
      col = "red")

lines(n.hp,
      predict(modelo.g.mpg.2,
              newdata = data.frame(hp = n.hp, cyl = 4),
              type = "response"),
      col = "blue")

lines(n.hp,
      predict(modelo.g.mpg.2,
              newdata = data.frame(hp = n.hp, cyl = 6),
              type = "response"),
      col = "blue")

lines(n.hp,
      predict(modelo.g.mpg.2,
              newdata = data.frame(hp = n.hp, cyl = 8),
              type = "response"),
      col = "blue")
```



Mas será que o modelo tem um desempenho melhor?

```r
data.frame(
  colnames = c("linear", "gama 1", "gama 2", "I. gaussian"),
  aic = c(AIC( modelo.linear),
          AIC(modelo.g.mpg),
          AIC(modelo.g.mpg.2),
          AIC(modelo.ig.mpg)),
  verossimilhança = c(logLik(modelo.linear),
                      logLik(modelo.g.mpg),
                      logLik(modelo.g.mpg.2),
```

```r
                      logLik(modelo.ig.mpg))
  )
```

```
##      colnames      aic verossimilhança
## 1      linear 181.2386       -87.61931
## 2      gama 1 170.2344       -82.11719
## 3      gama 2 162.9300       -77.46498
## 4 I. gaussian 165.9610       -78.98049
```

```r
anova(modelo.g.mpg,modelo.g.mpg.2,modelo.ig.mpg ,modelo.linear, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: mpg ~ hp
## Model 2: mpg ~ hp + cyl
## Model 3: mpg ~ hp + cyl
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        30    0.85275
## 2        29    0.63831  1  0.21444 0.001387 **
## 3        29    0.03657  0  0.60173
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
layout(matrix(c(1,2,3,4),c(2,2)))

hnp(modelo.linear, xlab = 'N(0,1)', ylab = 'Res Linear')
```

```
## Gaussian model (lm object)
```

```r
hnp(modelo.g.mpg, xlab = 'N(0,1)', ylab = 'Res Gama')
```

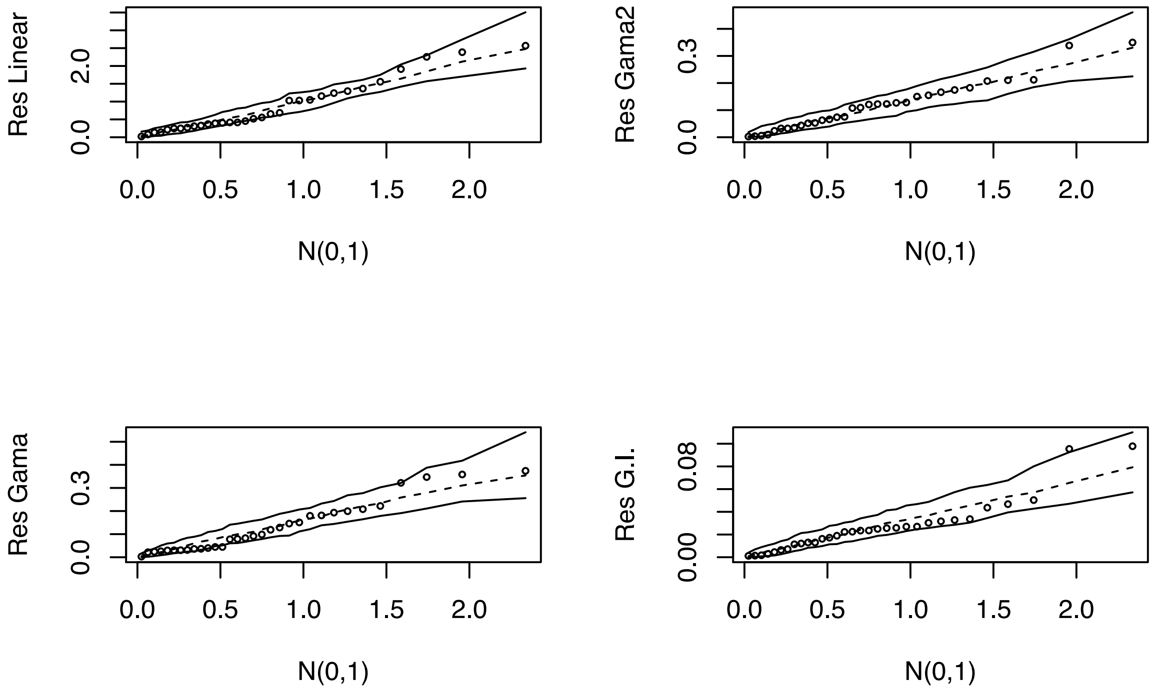
```
## Gamma model
```

```r
hnp(modelo.g.mpg.2, xlab = 'N(0,1)', ylab = 'Res Gama2')
```

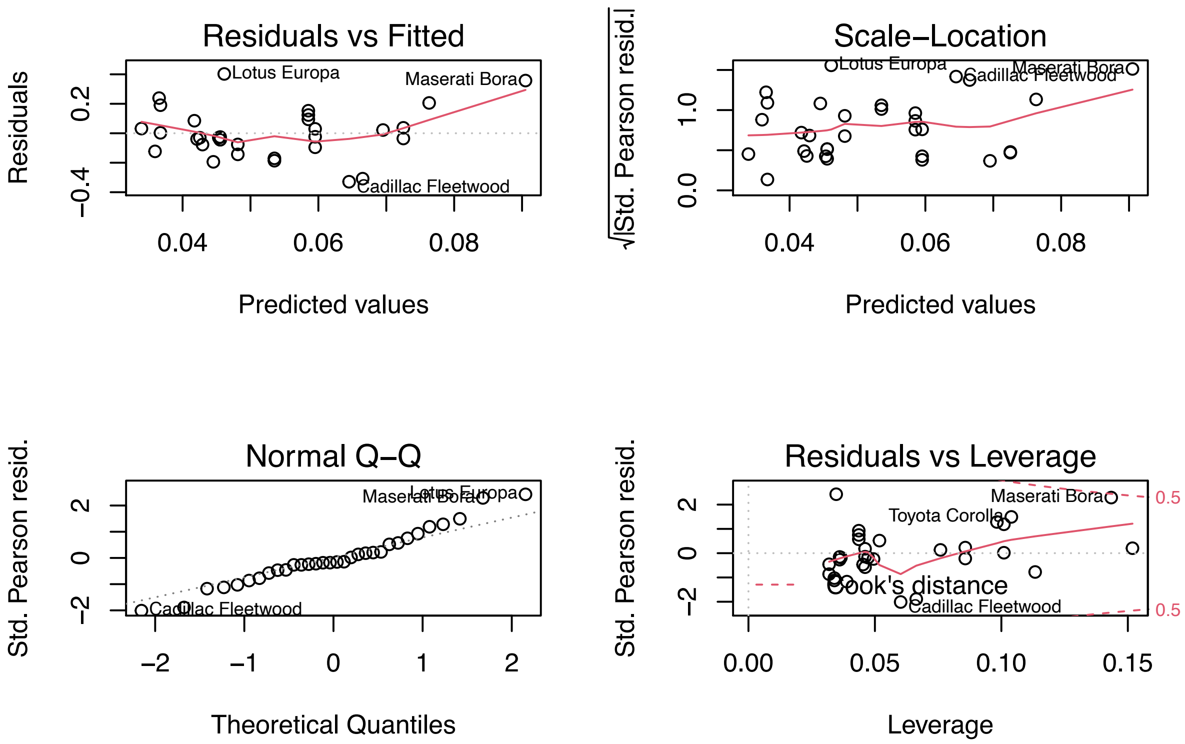
```
## Gamma model
```

```r
hnp(modelo.ig.mpg, xlab = 'N(0,1)', ylab = 'Res G.I.')
```

```
## Inverse gaussian model
```




```r
layout(matrix(c(1,2,3,4),c(2,2)))
plot(modelo.g.mpg)
```

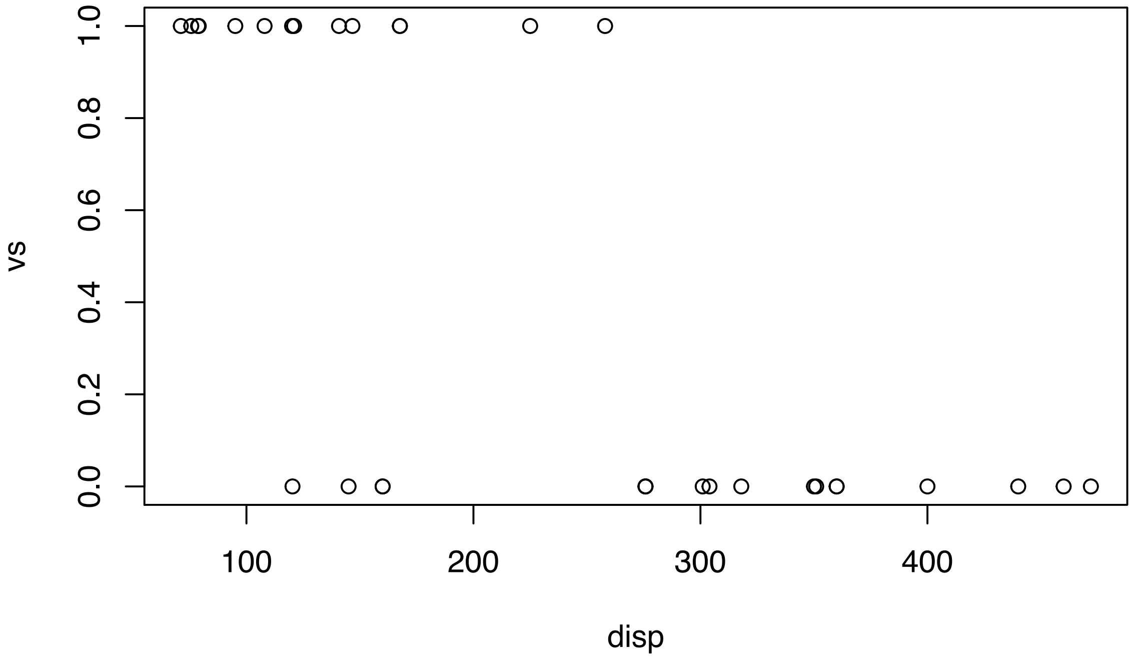


Outro exemplo regressão binomial

MTCARS tem um campo que diz respeito ao tipo de motor. Em V(0) ou em Linha vs(1).

Vamos tentar criar uma regressão que nos diga o tipo de motor baseado em seu peso (wt) e volume (disp).

```
#dispersao do tipo do motor
plot(vs ~ disp)
```




```
#esses são as variações para o modelo proposto
modelo.wt <- glm(vs ~ wt , family = binomial, data = mtcars)
modelo.disp <- glm(vs ~ disp ,  family = binomial, data = mtcars)
modelo.ambos <- glm(vs ~ disp + wt , family = binomial , data = mtcars)

#mas e se eu puder escolher outras variáveis
#modelo que será utilizada pra seleçao automática das variáveis
modelo.selecao <- glm(vs ~ . , family = binomial , data = mtcars)
#selecao automática
modelo.selecao <- stepAIC(modelo.selecao,  direction = "both")
```

```
## Start:  AIC=22
## vs ~ mpg + cyl + disp + hp + drat + wt + qsec + am + gear + carb
##
##        Df   Deviance AIC
## - disp  1 7.2539e-10  20
## - wt    1 7.2781e-10  20
## - hp    1 7.4603e-10  20
## - mpg   1 7.5020e-10  20
## - drat  1 7.7912e-10  20
## - gear  1 7.8065e-10  20
## - qsec  1 8.3407e-10  20
## - carb  1 9.2143e-10  20
## - am    1 9.2650e-10  20
## - cyl   1 1.0265e-09  20
## <none>    7.2154e-10  22
##
## Step:  AIC=20
## vs ~ mpg + cyl + hp + drat + wt + qsec + am + gear + carb
```

```
##
##        Df   Deviance AIC
## - mpg   1 7.5471e-10  18
## - hp    1 7.5801e-10  18
## - drat  1 7.9352e-10  18
## - gear  1 8.0497e-10  18
## - wt    1 8.5019e-10  18
## - qsec  1 9.0737e-10  18
## - am    1 9.4011e-10  18
## - cyl   1 1.0518e-09  18
## - carb  1 1.0541e-09  18
## <none>    7.2539e-10  20
## + disp  1 7.2154e-10  22
##
## Step:  AIC=18
## vs ~ cyl + hp + drat + wt + qsec + am + gear + carb
##
##        Df   Deviance AIC
## - hp    1 7.9730e-10  16
## - gear  1 8.0967e-10  16
## - drat  1 8.1165e-10  16
## - qsec  1 9.9068e-10  16
## - carb  1 1.0685e-09  16
## - cyl   1 1.1054e-09  16
## - am    1 1.1288e-09  16
## - wt    1 1.5389e-09  16
## <none>    7.5471e-10  18
## + mpg   1 7.2539e-10  20
## + disp  1 7.5020e-10  20
##
## Step:  AIC=16
## vs ~ cyl + drat + wt + qsec + am + gear + carb
##
##        Df   Deviance AIC
## - gear  1 9.4469e-10  14
## - qsec  1 1.1138e-09  14
## - am    1 1.1556e-09  14
## - cyl   1 1.1904e-09  14
## - drat  1 1.2746e-09  14
## - wt    1 1.5747e-09  14
## - carb  1 1.5927e-09  14
## <none>    7.9730e-10  16
## + hp    1 7.5471e-10  18
## + mpg   1 7.5801e-10  18
## + disp  1 7.8323e-10  18
##
## Step:  AIC=14
## vs ~ cyl + drat + wt + qsec + am + carb
##
##        Df Deviance    AIC
## - drat  1   0.0000 12.000
## - cyl   1   0.0000 12.000
## - am    1   0.0000 12.000
## - carb  1   0.0000 12.000
```

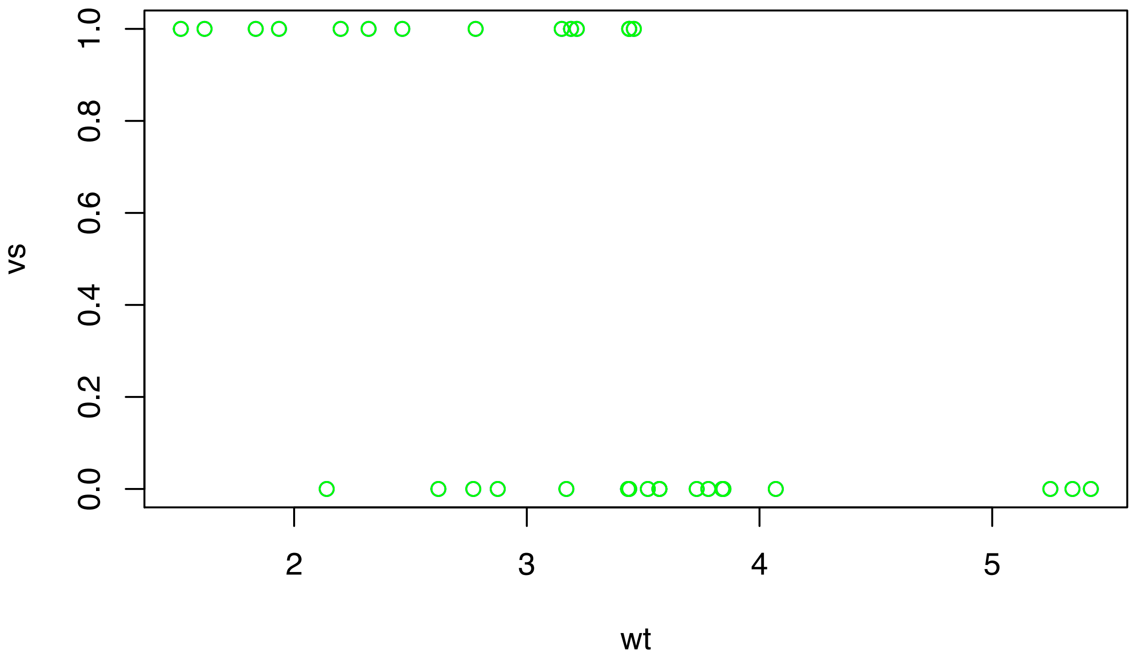
```
## - wt    1   0.0000 12.000
## <none>       0.0000 14.000
## + gear  1   0.0000 16.000
## + hp    1   0.0000 16.000
## + disp  1   0.0000 16.000
## + mpg   1   0.0000 16.000
## - qsec  1   4.2373 16.237
##
## Step:  AIC=12
## vs ~ cyl + wt + qsec + am + carb
##
##        Df Deviance    AIC
## - cyl   1   0.0000 10.000
## - carb  1   0.0000 10.000
## - am    1   0.0000 10.000
## - wt    1   0.0000 10.000
## <none>       0.0000 12.000
## + hp    1   0.0000 14.000
## + drat  1   0.0000 14.000
## + disp  1   0.0000 14.000
## + gear  1   0.0000 14.000
## + mpg   1   0.0000 14.000
## - qsec  1   4.4141 14.414
##
## Step:  AIC=10
## vs ~ wt + qsec + am + carb
##
##        Df Deviance    AIC
## - am    1    0.000  8.000
## - carb  1    0.000  8.000
## - wt    1    0.000  8.000
## <none>       0.000 10.000
## + hp    1    0.000 12.000
## + disp  1    0.000 12.000
## + cyl   1    0.000 12.000
## + drat  1    0.000 12.000
## + mpg   1    0.000 12.000
## + gear  1    0.000 12.000
## - qsec  1   21.023 29.023
##
## Step:  AIC=8
## vs ~ wt + qsec + carb
##
##        Df Deviance    AIC
## - carb  1    0.000  6.000
## <none>       0.000  8.000
## + am    1    0.000 10.000
## + mpg   1    0.000 10.000
## + hp    1    0.000 10.000
## + gear  1    0.000 10.000
## + cyl   1    0.000 10.000
## + disp  1    0.000 10.000
## + drat  1    0.000 10.000
## - wt    1   11.444 17.444
```

```
## - qsec  1   25.337 31.337
##
## Step:  AIC=6
## vs ~ wt + qsec
##
##        Df Deviance    AIC
## <none>       0.000  6.000
## + carb  1    0.000  8.000
## + mpg   1    0.000  8.000
## + am    1    0.000  8.000
## + disp  1    0.000  8.000
## + gear  1    0.000  8.000
## + cyl   1    0.000  8.000
## + drat  1    0.000  8.000
## + hp    1    0.000  8.000
## - wt    1   14.076 18.076
## - qsec  1   31.367 35.367
```

```r
#nova fórmula encontrada automaticamente
modelo.selecao$formula
```

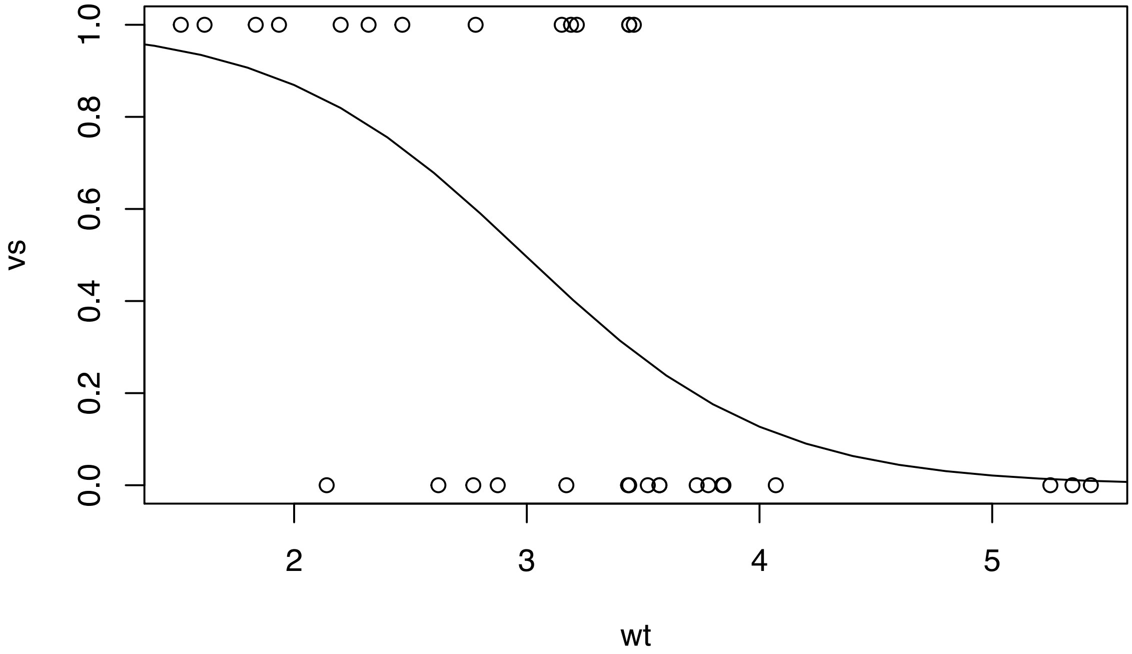
```
## vs ~ wt + qsec
```

```r
plot(vs ~ wt, col = "blue")
points(mtcars$wt,
       predict(modelo.selecao,
               newdata = data.frame(wt = mtcars$wt, qsec = mtcars$qsec),
               type = "response") ,
       data = mtcars, col = "green")
```

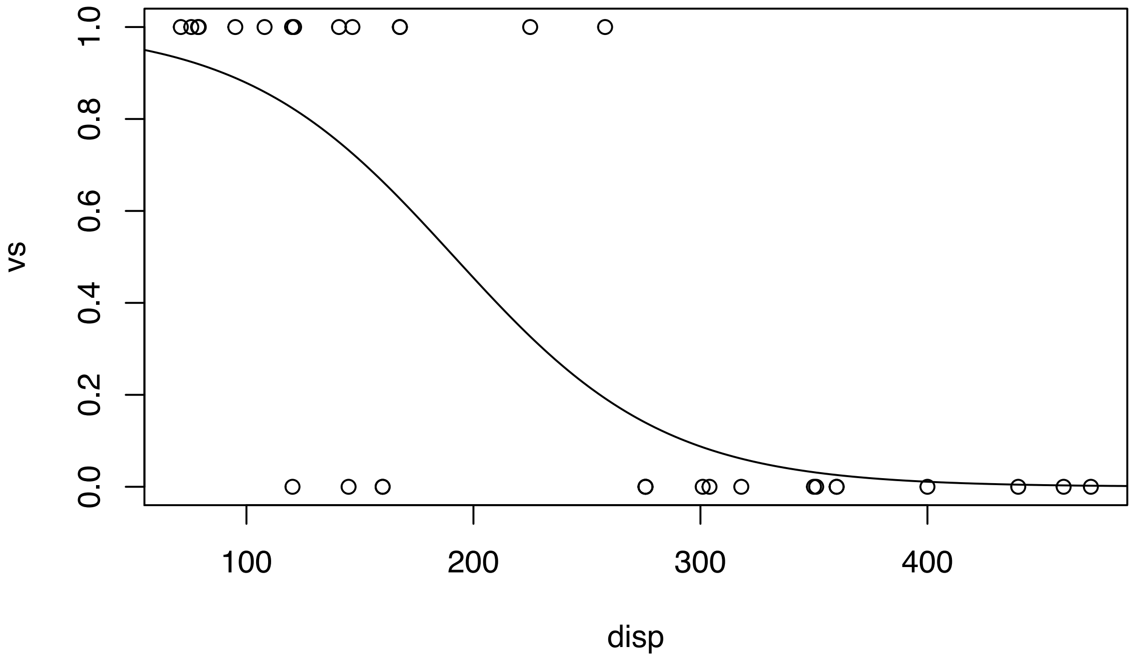



```r
x <- seq(0,6,by = 0.2)
resultado <- predict(modelo.wt,
                     data.frame(wt = x) ,
                     type = "response")
```

```r
plot(vs ~ wt);lines(x,resultado)
```

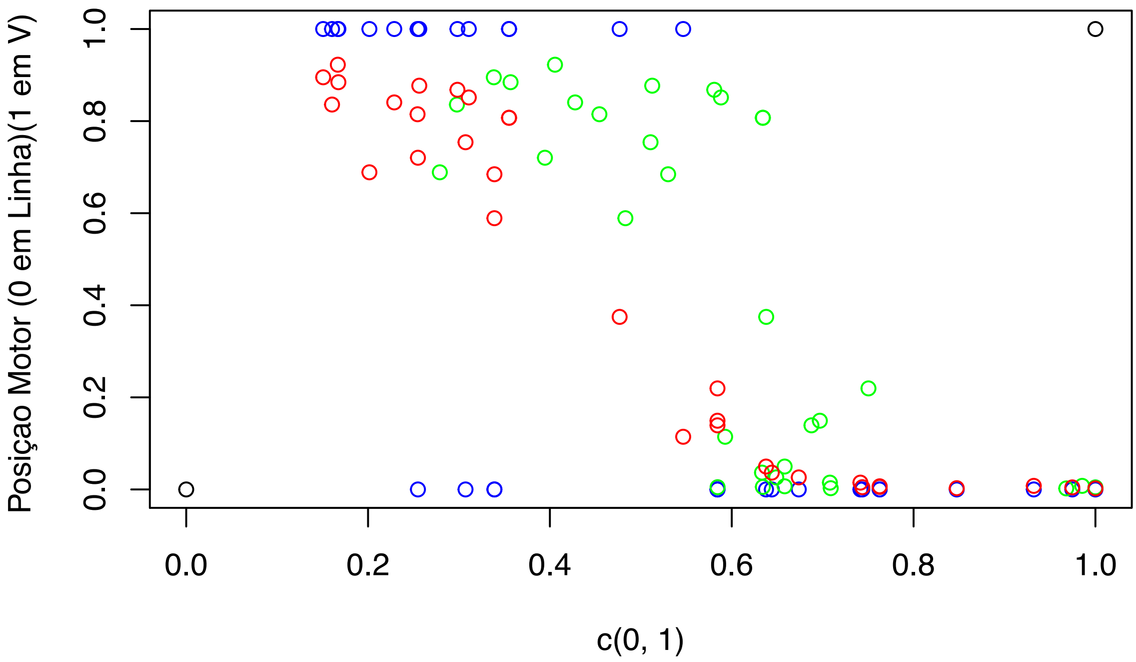



```r
z <- 0:500
resultado <- predict(modelo.disp,
                     data.frame(disp = z) ,
                     type = "response")
plot(vs ~ disp);lines(z,resultado)
```




```r
resultado <- predict(modelo.ambos,
                     data.frame(disp = mtcars$disp, wt = mtcars$wt) ,
                     type = "response")
```

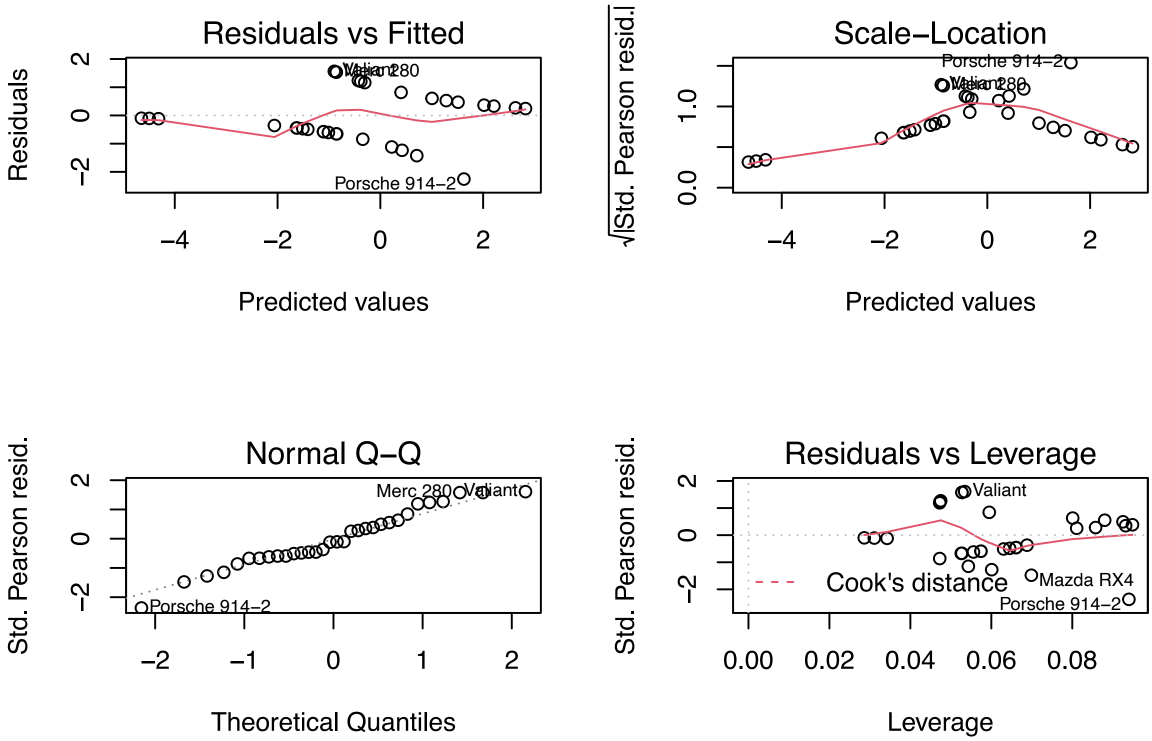
```r
plot(x = c(0,1) ,y=c(0,1),
     ylab = "Posiçao Motor (0 em Linha)(1 em V) ");

points(mtcars$disp/max(mtcars$disp),
       col = "blue");

points(mtcars$wt/max(mtcars$wt),
       resultado,
       col = "green");

points(mtcars$disp/max(mtcars$disp),
       resultado, col = "red")
```




```r
exp(confint.default(modelo.wt))/(1+ exp(confint.default(modelo.wt)))
```

```
##                  2.5 %    97.5 %
## (Intercept) 0.76924559 0.9999638
## wt          0.03431562 0.3813413
```

```r
layout(matrix(c(1,2,3,4),c(2,2)))
plot(modelo.wt)
```


Residuals vs Fitted
Residuals
Predicted values
Porsche 914−2
Scale−Location
√|Std. Pearson resid.|
Predicted values
Porsche 914−2
Normal Q−Q
Std. Pearson resid.
Theoretical Quantiles
Merc 280
Valiant
Porsche 914−2
Residuals vs Leverage
Std. Pearson resid.
Leverage
Valiant
Mazda RX4
Porsche 914−2
Cook's distance