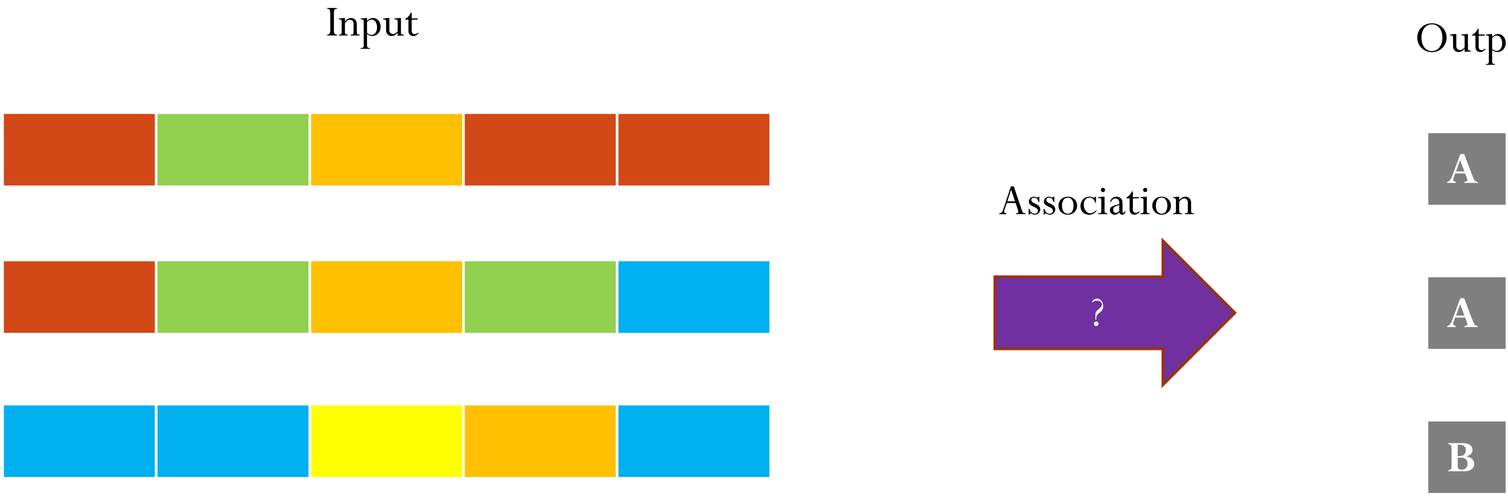
# Supervised learning. Linear models

Ruxandra Stoean

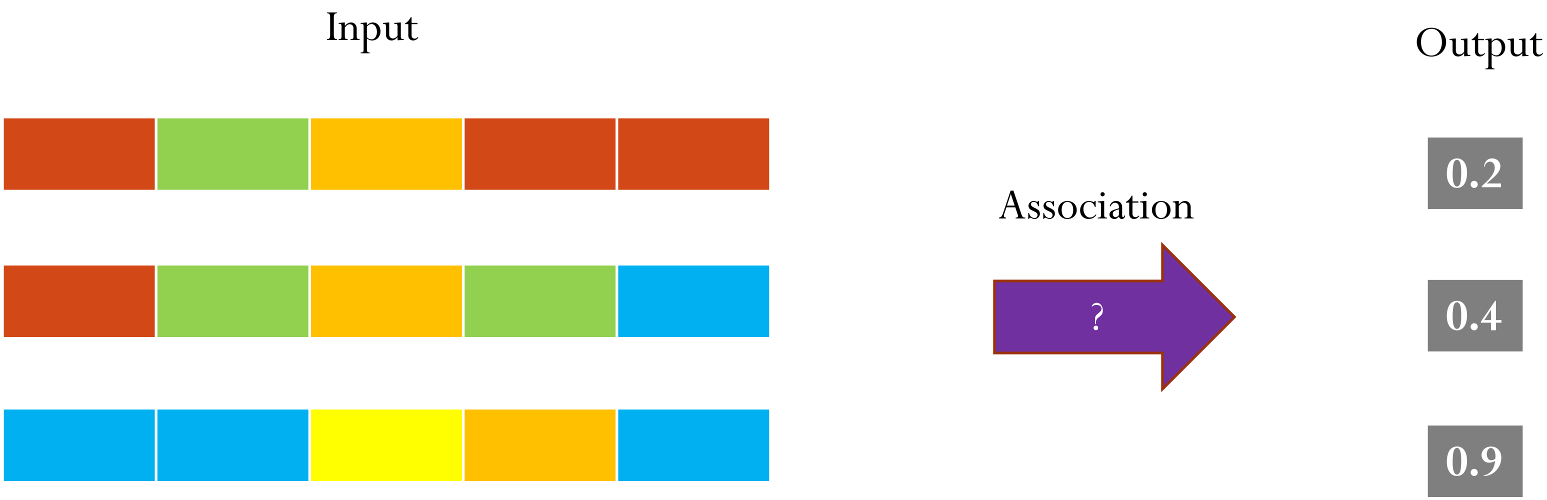
# Supervised learning

- Concept
  - Input data with available output values
  - Learn the input – output association
  - Prediction on the target of new data
- Classification
  - Output grouped in two or more classes
  - Qualitative output (discrete, factor, categorical, ordered categorical)
  - Prediction on the class of new data
- Regression
  - Quantitative output (continuous)
  - Prediction on the target numerical value for new input

# Classification



# Regression



Input

Association

?

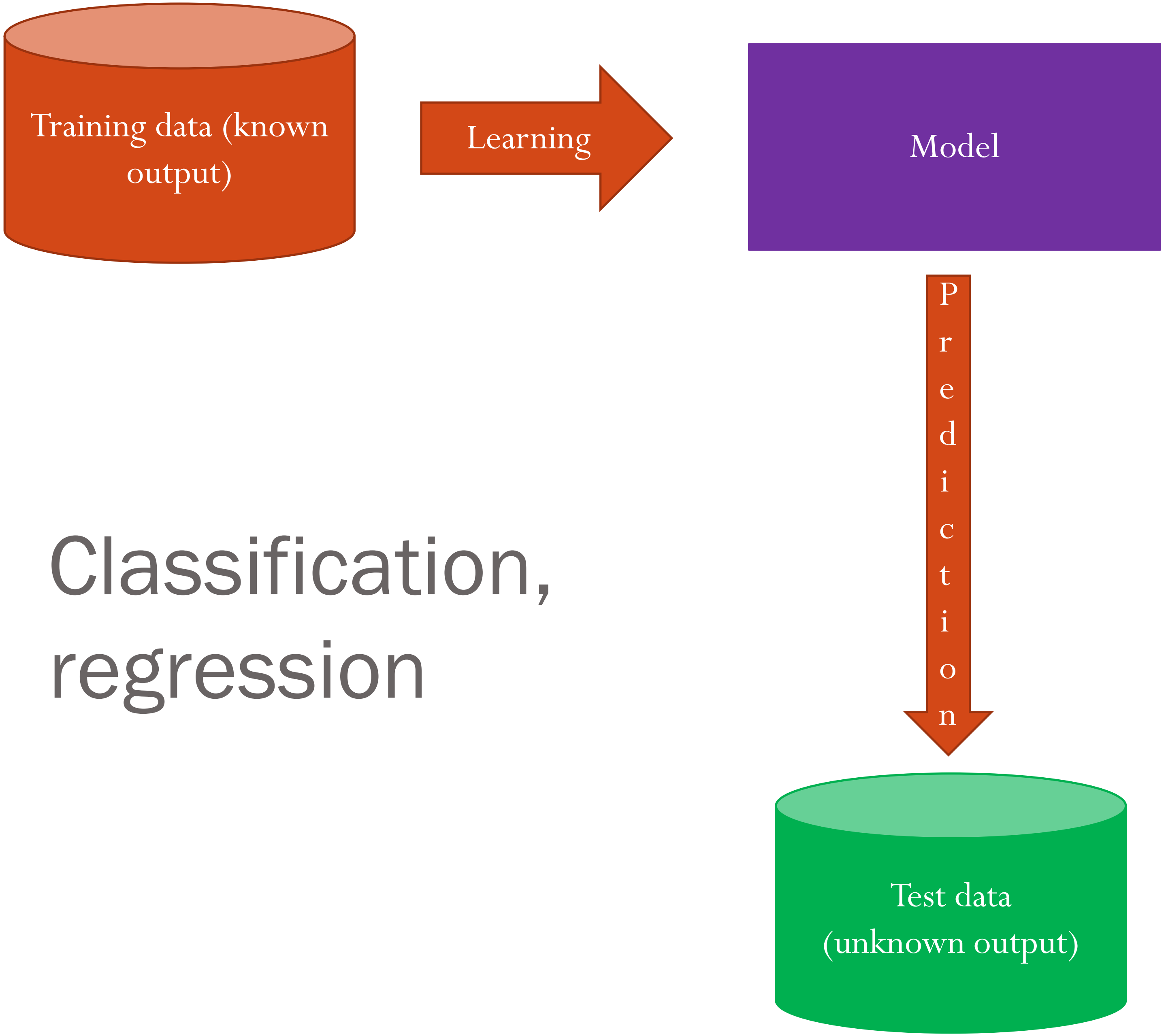
Output

0.2

0.4

0.9

# Classification, regression



Training data (known output)

Learning

Model

Prediction

Test data (unknown output)

# Supervised learning

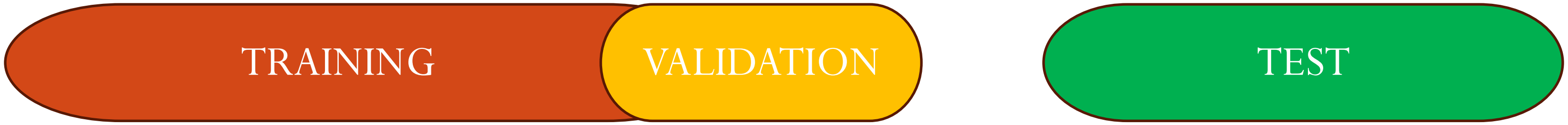
- A data set - pairs of the type (input, output)
  - The input is a sequence of values for the data attributes
  - The output is a confirmed decision
- Every record (object, example, vector) is described by a number of attributes with values from a discrete or continuous domain.
- The targets are either all discrete (classification), or continuous (regression).

# Definitions

- Given a data set of *m* records $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}\}$, where
  - every data point is defined by *n* attributes $\mathbf{x_i} \in R^n$
  - each has an associated outcome $y_1, \ldots, y_m$
  - Discrete outcome *y* -> classification
  - Continuous outcome *y* -> regression
- A supervised learning task is to build a model that
  - learns the association between *x* and *y*
  - and predicts the outcome for new data points.

# The model and the data

- The data set is split in three distinct subsets:
  - The training set
  - The validation set
  - The test set (data output is hidden)
- The algorithm learns the association between every training data point and its output (training phase).
- The obtained model:
  - tested on the validation set to measure its prediction error
    - Parameter tuning
    - Variable selection
  - tested on the test set to assess the generalization ability



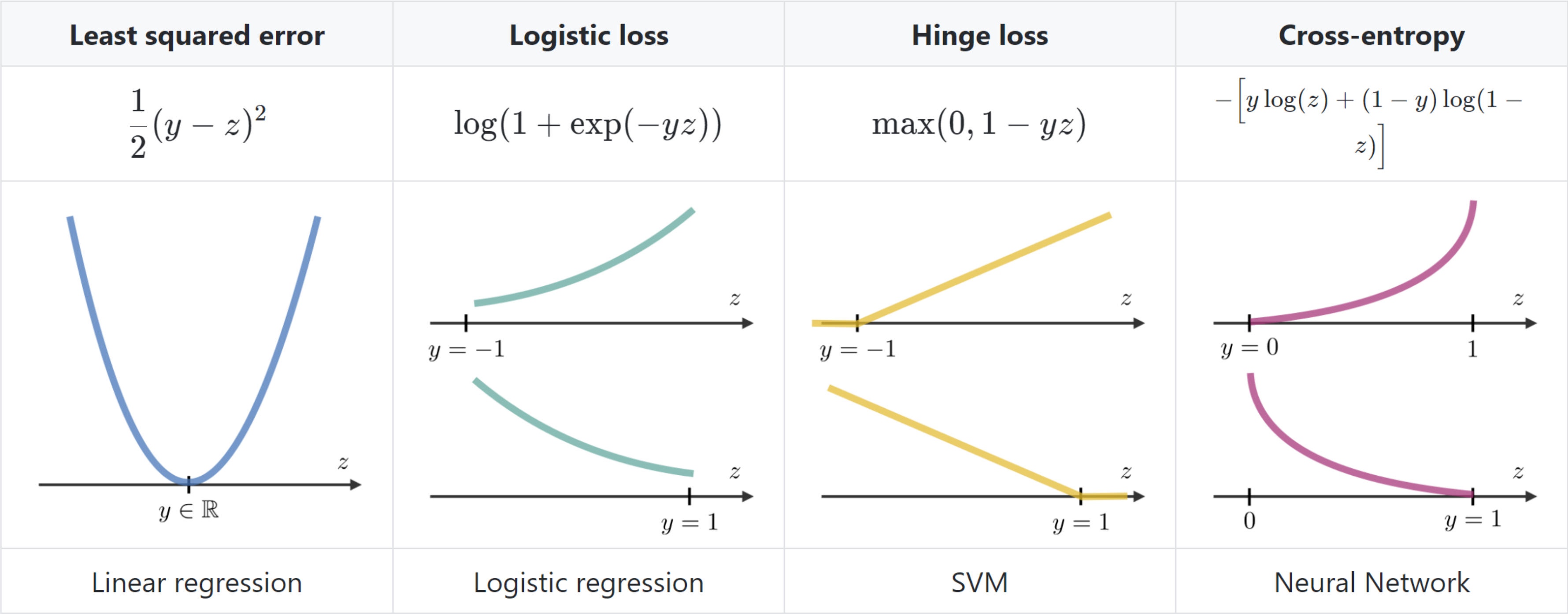
TRAINING

VALIDATION

TEST

# The loss function

- Loss measures how much the model learns the I/O relationship.
- Takes as arguments the predicted value of the model z and the corresponding data real output y and returns their difference.
- Cost function – sum of loss over all training data



| Least squared error | Logistic loss | Hinge loss | Cross-entropy |
|---|---|---|---|
| $\frac{1}{2}(y-z)^2$ | $\log(1+\exp(-yz))$ | $\max(0,1-yz)$ | $-\left[y\log(z)+(1-y)\log(1-z)\right]$ |
| Linear regression | Logistic regression | SVM | Neural Network |

$z$
$y \in \mathbb{R}$
$z$
$y=-1$
$z$
$y=1$
$z$
$y=-1$
$z$
$y=1$
$z$
$y=0$
1
$z$
0
$y=1$

https://stanford.edu/~shervine/teaching/cs-229

# Example

- Classification – Pima Indians diabetes
- Class:
  - positive for diabetes (class 1) – 268 cases
  - negative (class 0) – 500 cases

| Pregnancies | Glucose | Blood pressure | Skin thickness | Insulin | BMI | Diabetes pedigree | Age | Class |
|---|---|---|---|---|---|---|---|---|
| 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 50 | 1 |
| 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 31 | 0 |
| 8 | 183 | 64 | 0 | 0 | 23.3 | 0.672 | 32 | 1 |

# Example

- Regression - Boston housing – 506 instances
- Median value of houses in thousands of American $

| Crime rate | Proportion of residential zones | Proportion of non-retail business | River | Nitric oxide concentration | Average number of rooms | Proportion of building before 1940 | Distance to employment centres | Accessibility to railways | Tax | Pupil-teacher ratio | Proportip [illegible] A[illegible] p[illegible] e | Percent of l[illegible]r [illegible]s [illegible]u [illegible]on | Median value |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.00632 | 18.00 | 2.310 | 0 | 0.5380 | 6.5750 | 65.20 | 4.0900 | 1 | 296.0 | 15.30 | 396[illegible]90[illegible] | [illegible]98 | 24.00 |
| 0.02731 | 0.00 | 7.070 | 0 | 0.4690 | 6.4210 | 78.90 | 4.9671 | 2 | 242.0 | 17.80 | 3[illegible]9[illegible] | [illegible]4 | 21.60 |
| 0.02729 | 0.00 | 7.070 | 0 | 0.4690 | 7.1850 | 61.10 | 4.9671 | 2 | 242.0 | 17.80 | 392.83 | 4.03 | 34.70 |

# Linear models

- Sufficient performance for a low number of training data points

- Regression
  - Linear regression model

- Classification
  - Logistic regression
  - Linear support vector machines

# Multiple linear regression

- $m$ training data of the type $(X,Y)$:
  - $X = (X^1, X^2, \ldots, X^n)$ an input vector
    - n attributes - independent, explanatory, predictor variables
  - $Y$ – outcome
    - Dependent variable, function of the other variables, response variable
    - Y takes continuous values
- The regression equation: $f(X) = b_0 + b_1 X^1 + b_2 X^2 + \ldots + b_n X^n$
  - $b_{1,} b_{2, \ldots,} b_n$ – regression coefficients
  - $b_0$ – intercept

# Parameter estimation

- Least squares method

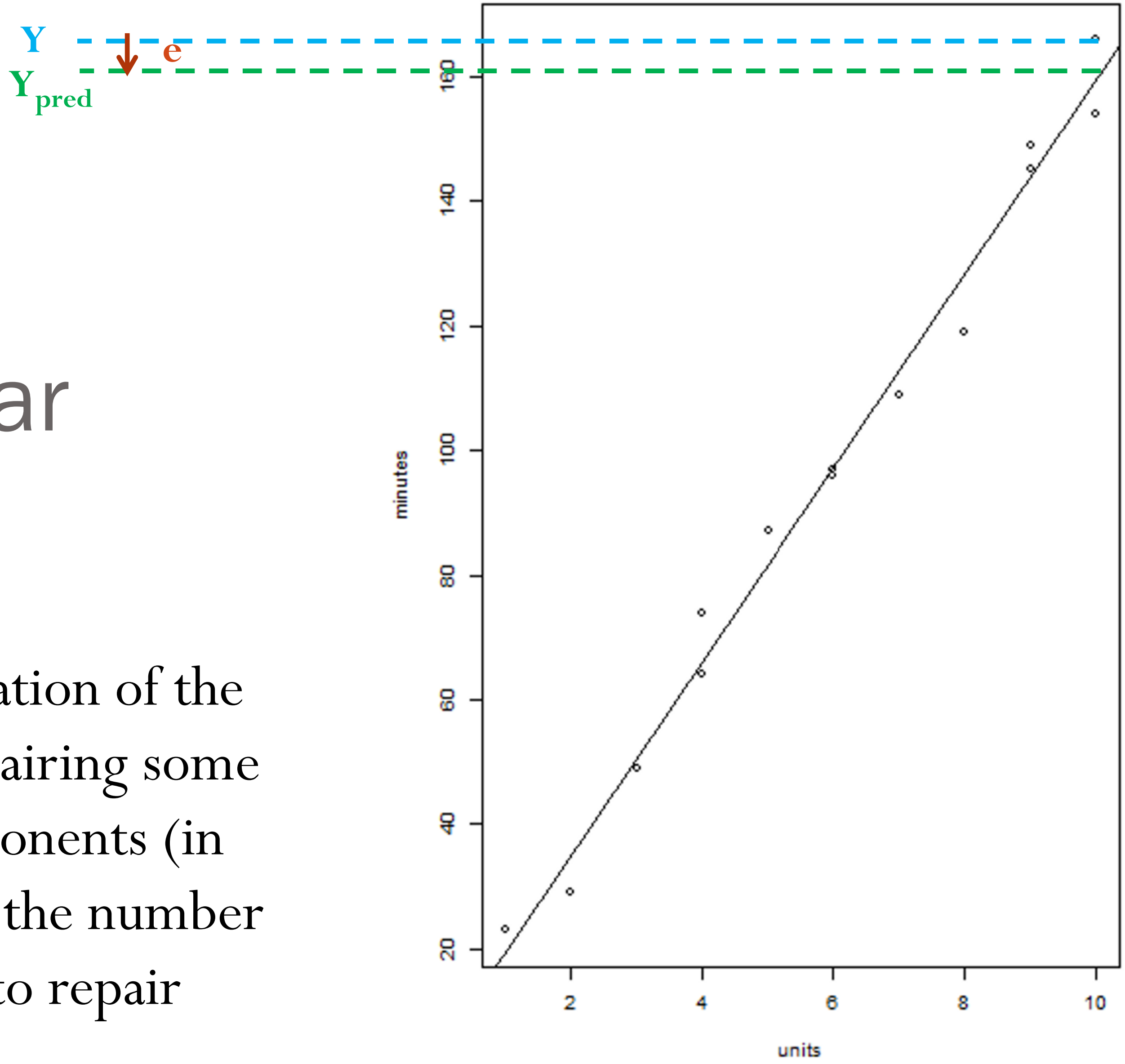
- Determines the coefficients $b_j$, $j = 0, 1, 2, \ldots, n$, such as to minimize the sum of squared residuals ($SR$) for the training data $(x_i, y_i)$, $i = 1, 2, \ldots, m$

- $SR = \sum_{i=1}^{m} (y_i - f(x_i))^2$

- The residual (error) is the difference between the real outcome value and the value predicted through $f$.

# Simple linear regression

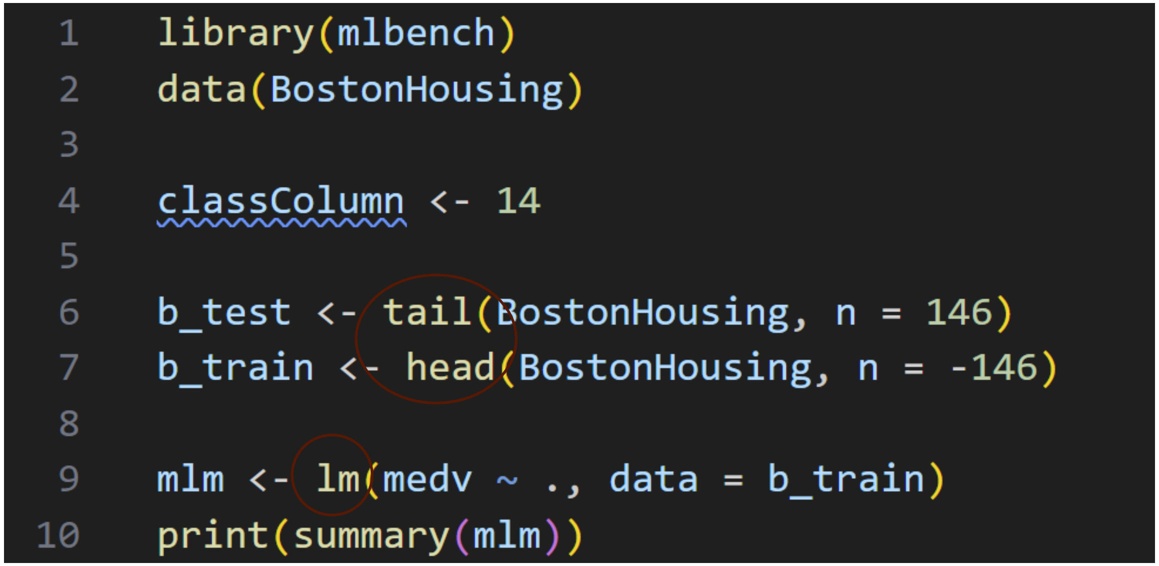
- Example: estimation of the duration for repairing some computer components (in minutes), given the number of components to repair



S. Chatterjee, A.S. Hadi, Regression Analysis by Example (4th ed.), Wiley, 2006.

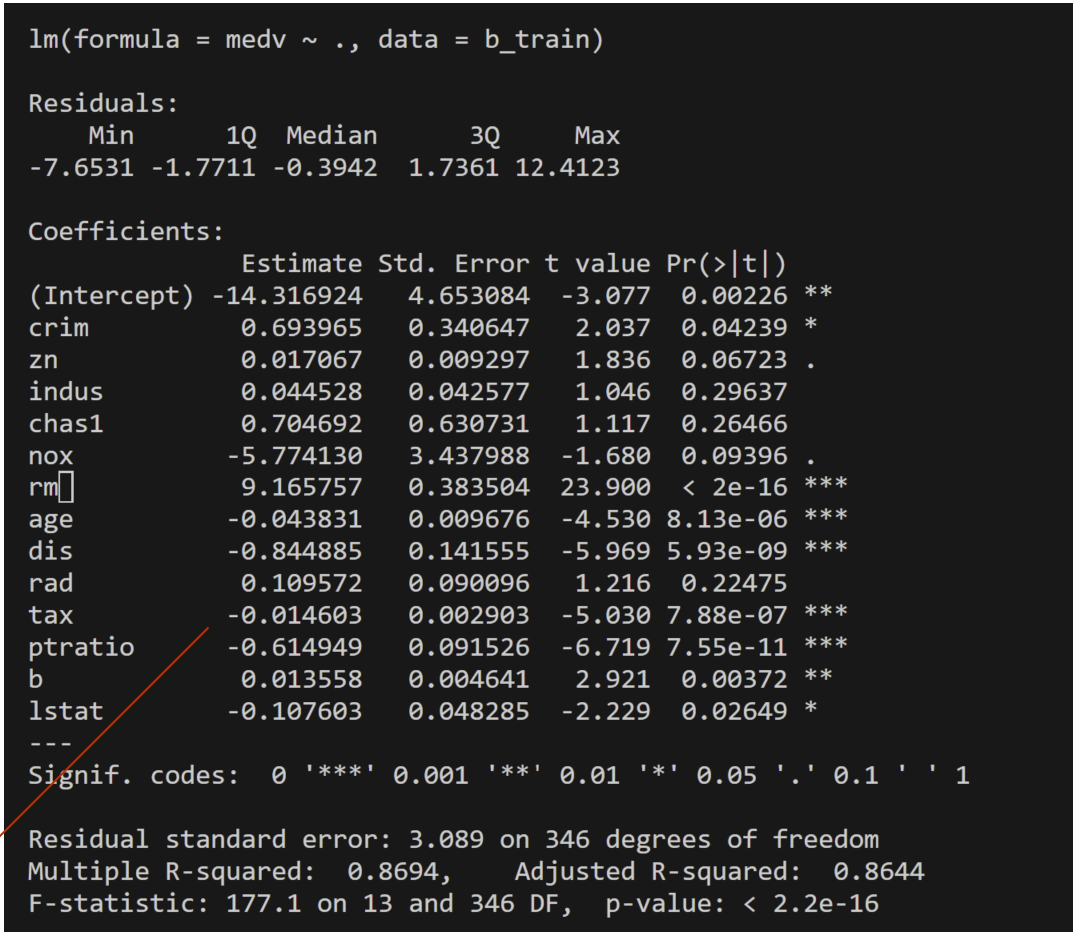
# Multiple linear regression - R

- Problem: Boston housing (506 records, 13 indicators + target)
  - In package mlbench that needs to be installed and included
  - Outcome: medv
- Function lm()



```r
library(mlbench)
data(BostonHousing)

classColumn <- 14

b_test <- tail(BostonHousing, n = 146)
b_train <- head(BostonHousing, n = -146)

mlm <- lm(medv ~ ., data = b_train)
print(summary(mlm))
```

# Results and interpretation (1/4)

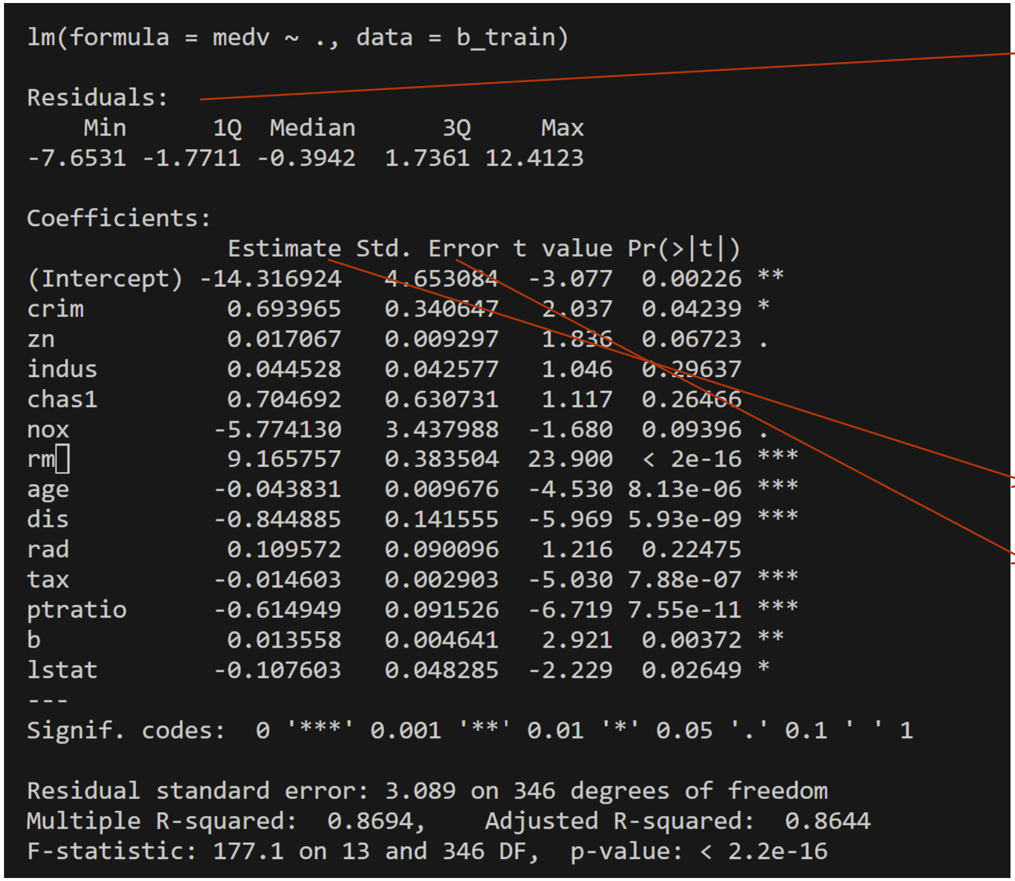


```
lm(formula = medv ~ ., data = b_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.6531 -1.7711 -0.3942  1.7361 12.4123 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -14.316924   4.653084  -3.077  0.00226 ** 
crim          0.693965   0.340647   2.037  0.04239 *  
zn            0.017067   0.009297   1.836  0.06723 .  
indus         0.044528   0.042577   1.046  0.29637    
chas1         0.704692   0.630731   1.117  0.26466    
nox          -5.774130   3.437988  -1.680  0.09396 .  
rm□           9.165757   0.383504  23.900  < 2e-16 ***
age          -0.043831   0.009676  -4.530 8.13e-06 ***
dis          -0.844885   0.141555  -5.969 5.93e-09 ***
rad           0.109572   0.090096   1.216  0.22475    
tax          -0.014603   0.002903  -5.030 7.88e-07 ***
ptratio      -0.614949   0.091526  -6.719 7.55e-11 ***
b             0.013558   0.004641   2.921  0.00372 ** 
lstat        -0.107603   0.048285  -2.229  0.02649 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.089 on 346 degrees of freedom
Multiple R-squared:  0.8694,	Adjusted R-squared:  0.8644 
F-statistic: 177.1 on 13 and 346 DF,  p-value: < 2.2e-16
```

- Model is:

f(X) =Y= -14.31 + 0.69*crim + 0.01*zn + 0.04*indus + 0.7*chas – 5.77*nox + 9.16*rm – 0.04*age – 0.84*dis + 0.1*rad – 0.01*tax – 0.61*ptratio + 0.01*black – 0.1*lstat

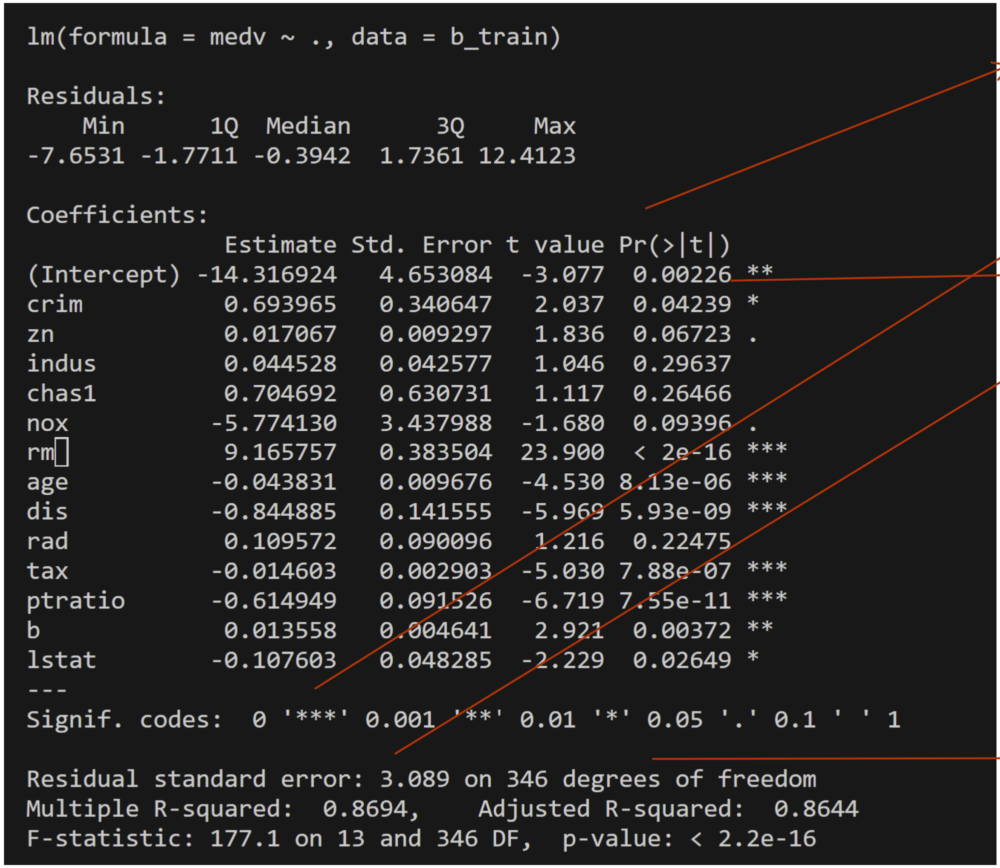
# Results and interpretation (2/4)



```
lm(formula = medv ~ ., data = b_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.6531 -1.7711 -0.3942  1.7361 12.4123 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -14.316924   4.653084  -3.077  0.00226 ** 
crim          0.693965   0.340647   2.037  0.04239 *  
zn            0.017067   0.009297   1.836  0.06723 .  
indus         0.044528   0.042577   1.046  0.29637    
chas1         0.704692   0.630731   1.117  0.26466    
nox          -5.774130   3.437988  -1.680  0.09396 .  
rm▯           9.165757   0.383504  23.900  < 2e-16 ***
age          -0.043831   0.009676  -4.530 8.13e-06 ***
dis          -0.844885   0.141555  -5.969 5.93e-09 ***
rad           0.109572   0.090096   1.216  0.22475    
tax          -0.014603   0.002903  -5.030 7.88e-07 ***
ptratio      -0.614949   0.091526  -6.719 7.55e-11 ***
b             0.013558   0.004641   2.921  0.00372 ** 
lstat        -0.107603   0.048285  -2.229  0.02649 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.089 on 346 degrees of freedom
Multiple R-squared:  0.8694,    Adjusted R-squared:  0.8644 
F-statistic: 177.1 on 13 and 346 DF,  p-value: < 2.2e-16
```

- Hypothesis: Residuals must have normal distribution, with mean at 0 and values closer to the mean and not at margin
- Regression coefficients
- Standard error (StD) of a coefficient – measures the precision of the model in estimating the coefficient unknown value
  - Smaller value is better

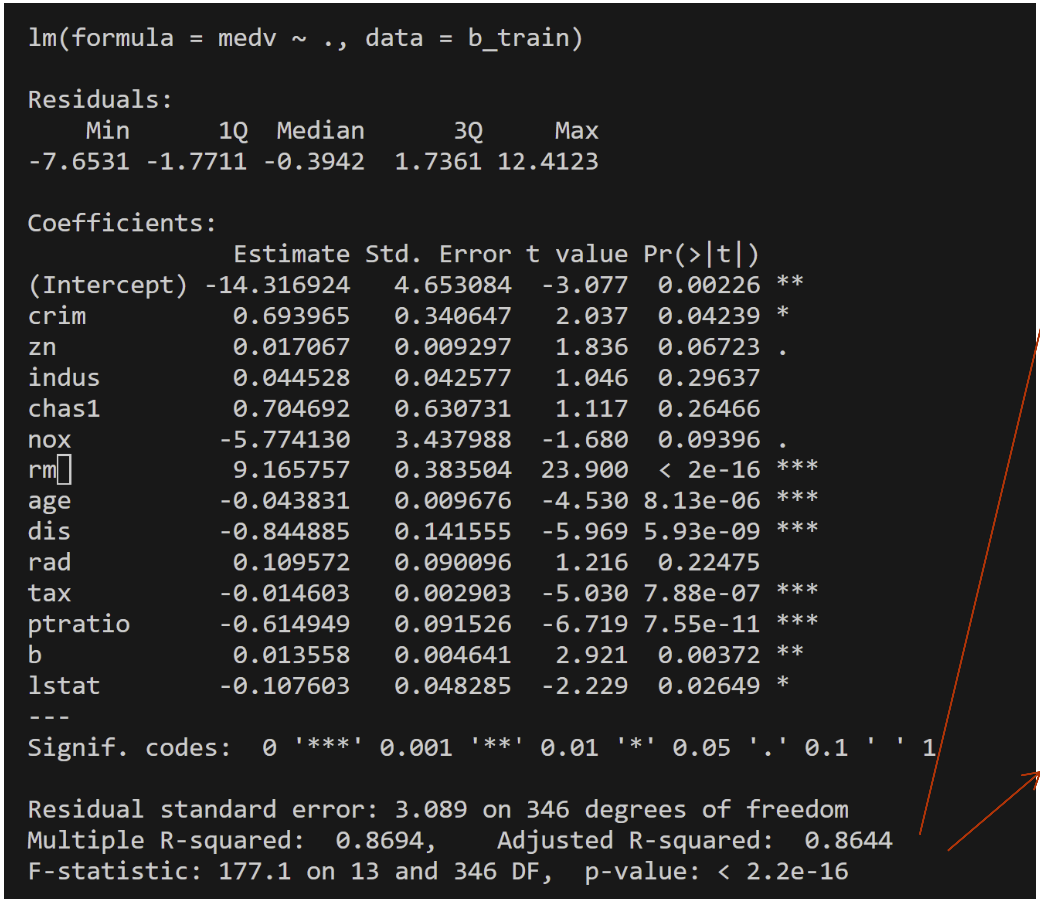
# Results and interpretation (3/4)



```
lm(formula = medv ~ ., data = b_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.6531 -1.7711 -0.3942  1.7361 12.4123 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -14.316924   4.653084  -3.077  0.00226 ** 
crim          0.693965   0.340647   2.037  0.04239 *  
zn            0.017067   0.009297   1.836  0.06723 .  
indus         0.044528   0.042577   1.046  0.29637    
chas1         0.704692   0.630731   1.117  0.26466    
nox          -5.774130   3.437988  -1.680  0.09396 .  
rm▯           9.165757   0.383504  23.900  < 2e-16 ***
age          -0.043831   0.009676  -4.530 8.13e-06 ***
dis          -0.844885   0.141555  -5.969 5.93e-09 ***
rad           0.109572   0.090096   1.216  0.22475    
tax          -0.014603   0.002903  -5.030 7.88e-07 ***
ptratio      -0.614949   0.091526  -6.719 7.55e-11 ***
b             0.013558   0.004641   2.921  0.00372 ** 
lstat        -0.107603   0.048285  -2.229  0.02649 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.089 on 346 degrees of freedom
Multiple R-squared:  0.8694,	Adjusted R-squared:  0.8644 
F-statistic: 177.1 on 13 and 346 DF,  p-value: < 2.2e-16
```

- Measure of the importance of the variable
  - *** high significance
  - p-value as low as possible
- Standard error of residuals – ideal proportional to the first and third quartile (1.5 +/- mean)
- Degrees of freedom – difference between the number of records and number of variables (coefficients + intercept)

# Results and interpretation (4/4)



```
lm(formula = medv ~ ., data = b_train)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.6531 -1.7711 -0.3942  1.7361 12.4123 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -14.316924   4.653084  -3.077  0.00226 ** 
crim          0.693965   0.340647   2.037  0.04239 *  
zn            0.017067   0.009297   1.836  0.06723 .  
indus         0.044528   0.042577   1.046  0.29637    
chas1         0.704692   0.630731   1.117  0.26466    
nox          -5.774130   3.437988  -1.680  0.09396 .  
rm▯           9.165757   0.383504  23.900  < 2e-16 ***
age          -0.043831   0.009676  -4.530 8.13e-06 ***
dis          -0.844885   0.141555  -5.969 5.93e-09 ***
rad           0.109572   0.090096   1.216  0.22475    
tax          -0.014603   0.002903  -5.030 7.88e-07 ***
ptratio      -0.614949   0.091526  -6.719 7.55e-11 ***
b             0.013558   0.004641   2.921  0.00372 ** 
lstat        -0.107603   0.048285  -2.229  0.02649 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.089 on 346 degrees of freedom
Multiple R-squared:  0.8694,	Adjusted R-squared:  0.8644 
F-statistic: 177.1 on 13 and 346 DF,  p-value: < 2.2e-16
```

- $R^2$ evaluates the degree of agreement (fit) of the model with the data – ideally closer to 1.
- The model explains 86% of the original variability, and 14% comes from residual variability.
- The F test compares the model with all parameters against one with less.
  - The low p-value says the model with all is good.

# Other hypotheses on the model

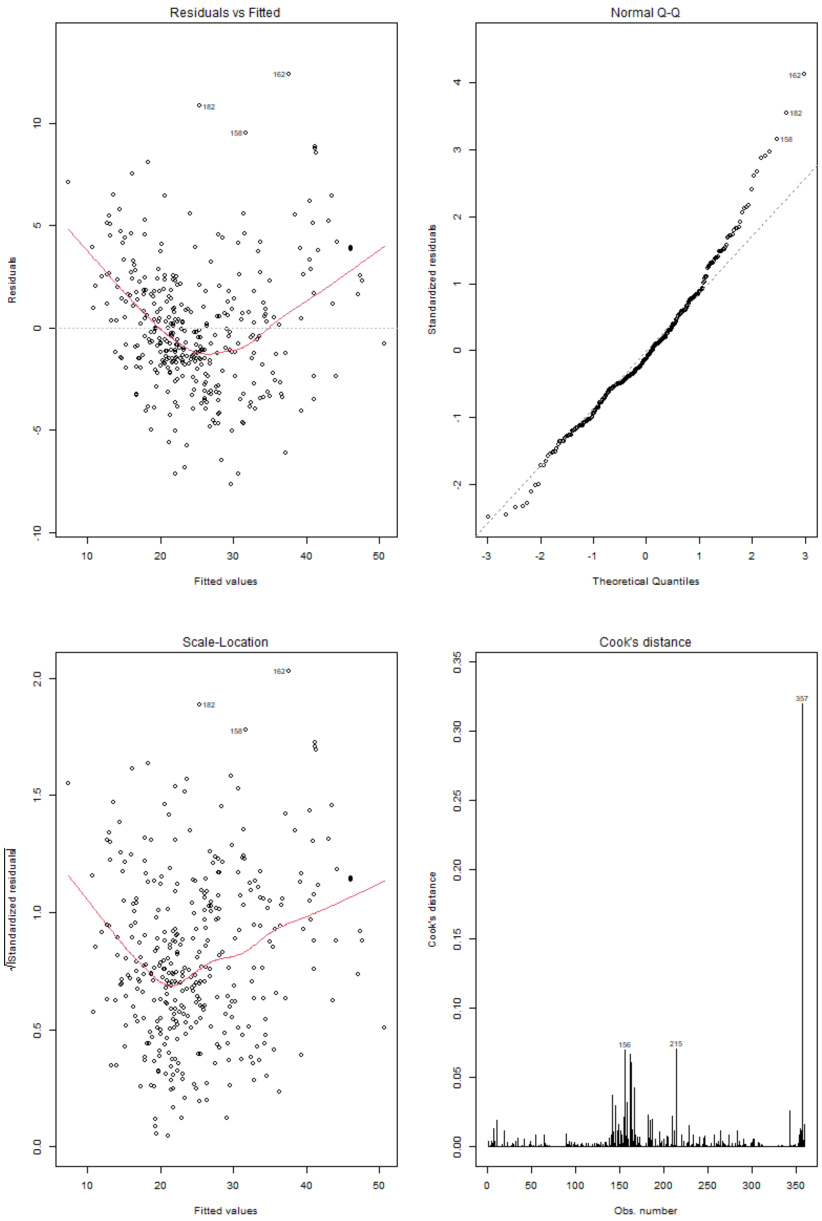
1. The predicted values must be expressed as a linear function of the X variables.

2. The variance of observations around the regression line is constant (homoscedasticity).

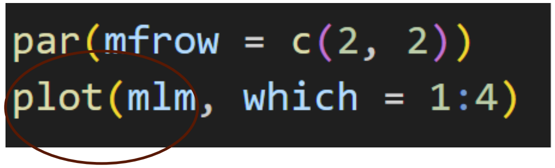
3. The predicted values (or the errors) have normal distribution.

- All these hypotheses can be checked by examining the residuals.

# Diagnostic plots (1/4)

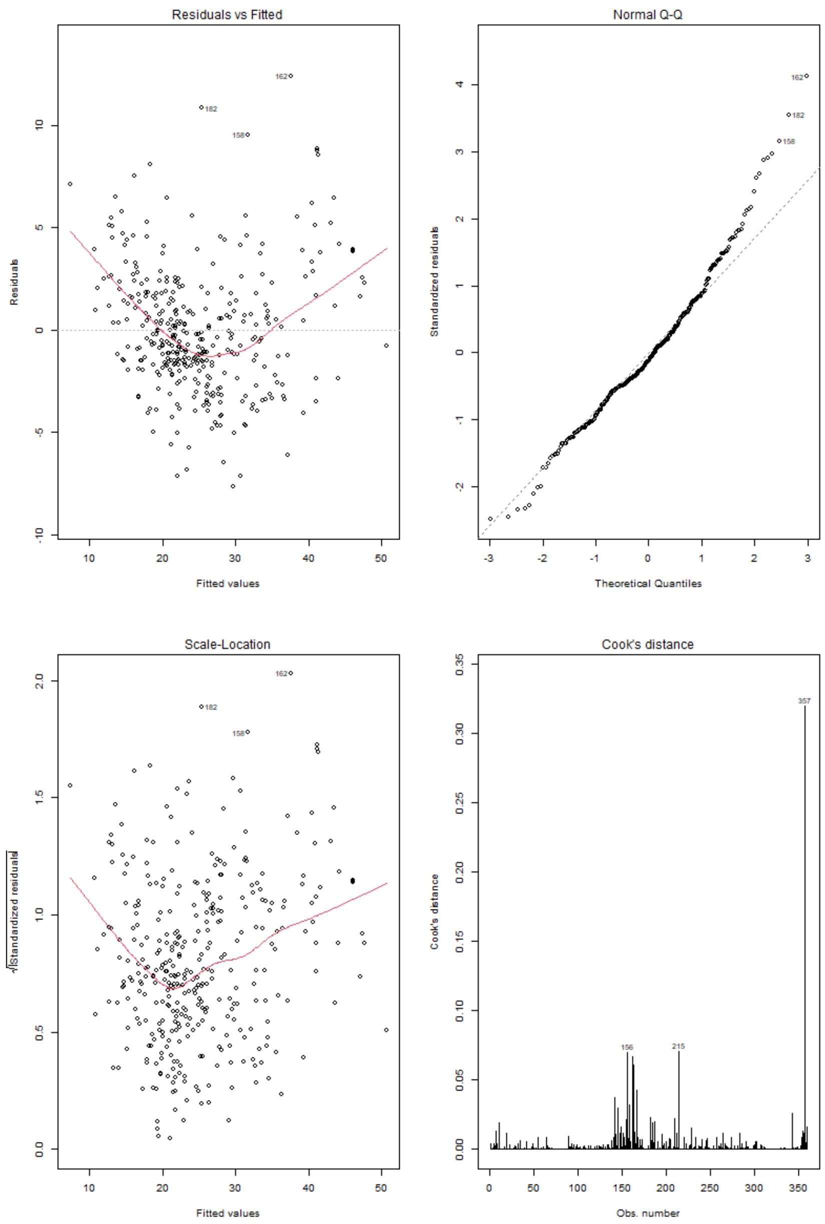


- R can plot diagnostic plots for the model that was fit to the data.
- Useful especially for multiple regression
  - When model visualization is impossible
- Add to the program:



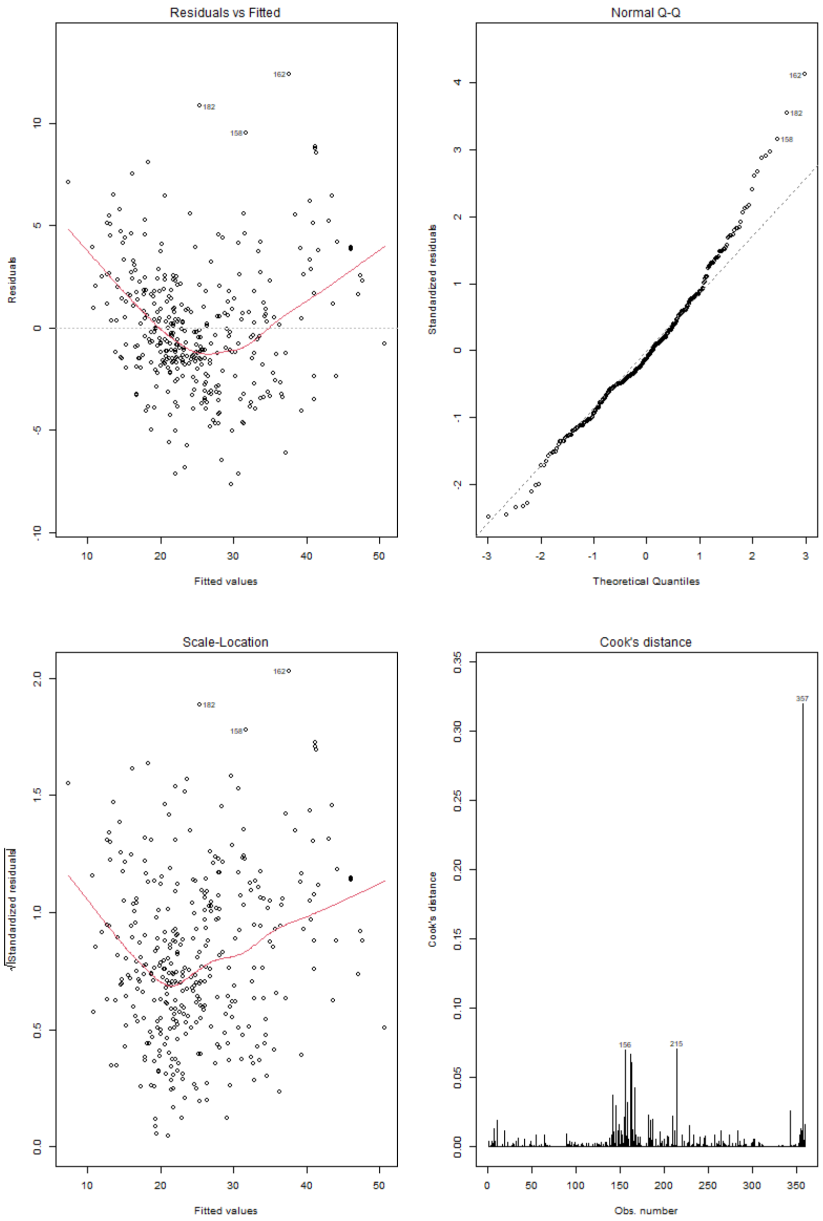
```
par(mfrow = c(2, 2))
plot(mlm, which = 1:4)
```

# Diagnostic plots (2/4)



1. The residual plot vs predicted values verify model <u>linearity</u> and <u>homoscedasticity</u>.
   1. Linearity – the red line should be aprox. horizontal, no curves
   2. Homoscedasticity – the variance of observations around the line y=0 is constant (constant variance).

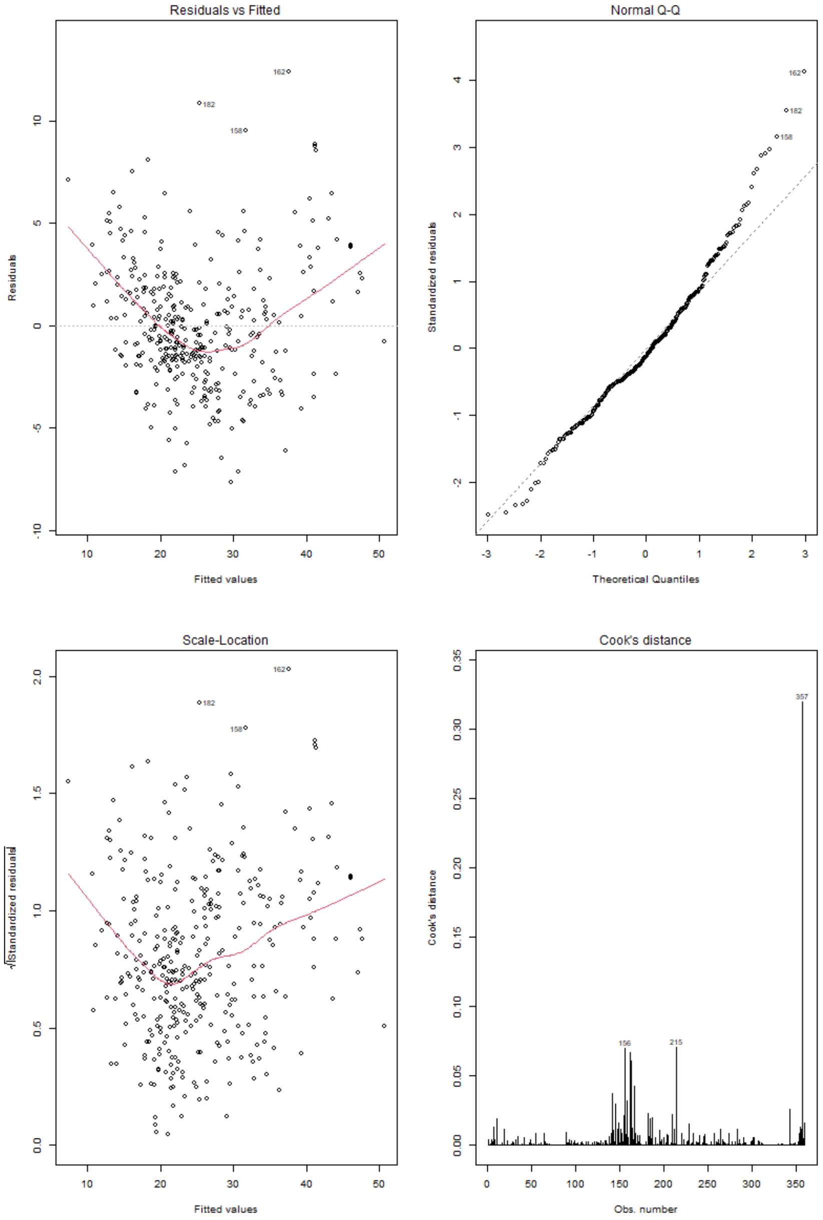
# Diagnostic plots (3/4)



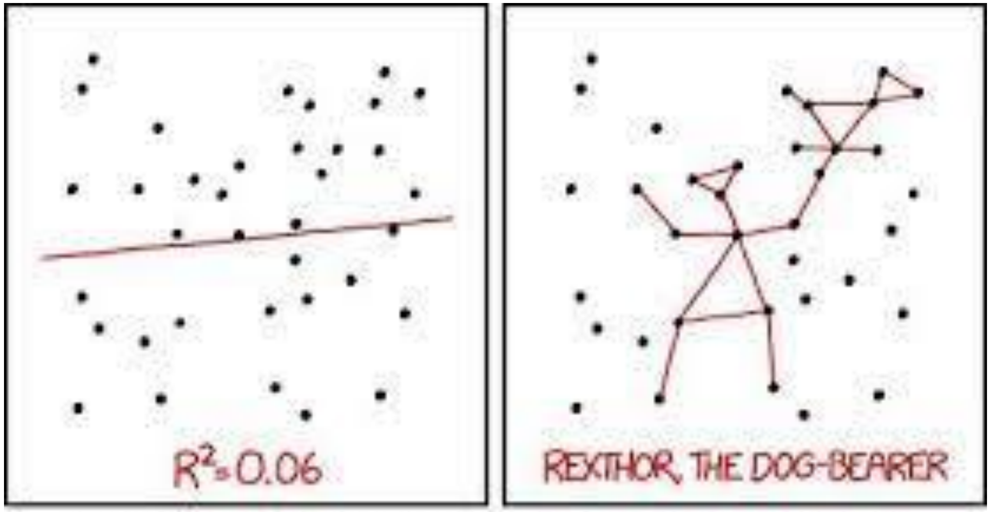
2. Check the hypothesis of a normal distribution for residuals (normal errors)
   i. Points should fall on a diagonal line.

3. Another check of linearity and homoscedasticity.
   ii. Residuals are standardized.

# Diagnostic plots (4/4)

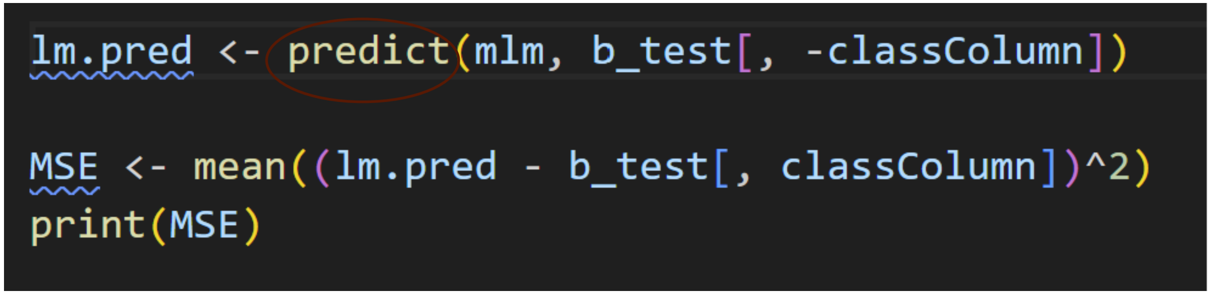


4. The Cook distance is a measure of the influence of each data on the regression coefficients.
   1. It measures the amount of change in the model if the record is omitted.
   2. Any data point for which the Cook distance is >=1 or larger than for the other points is influential for the model.

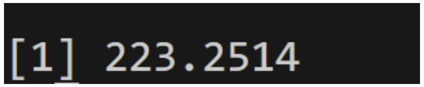


I DON'T TRUST LINEAR REGRESSIONS WHEN IT'S HARDER TO GUESS THE DIRECTION OF THE CORRELATION FROM THE SCATTER PLOT THAN TO FIND NEW CONSTELLATIONS ON IT.

# Prediction



```
lm.pred <- predict(mlm, b_test[, -classColumn])

MSE <- mean((lm.pred - b_test[, classColumn])^2)
print(MSE)
```

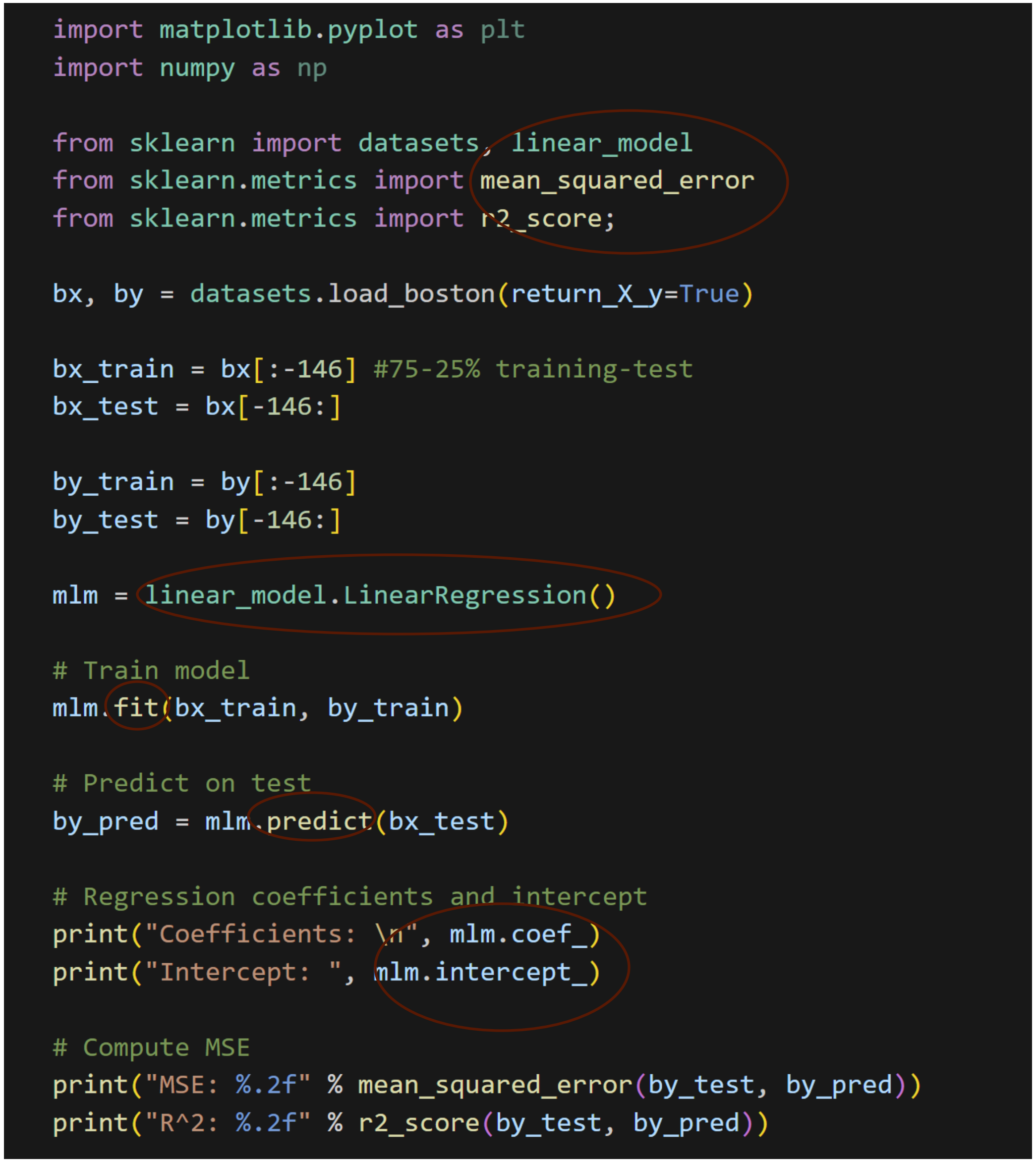


```
[1] 223.2514
```

- *Mean squared error*
- **y**$_i$ - actual values, **ŷ**$_i$ - predicted values
- n - number of records

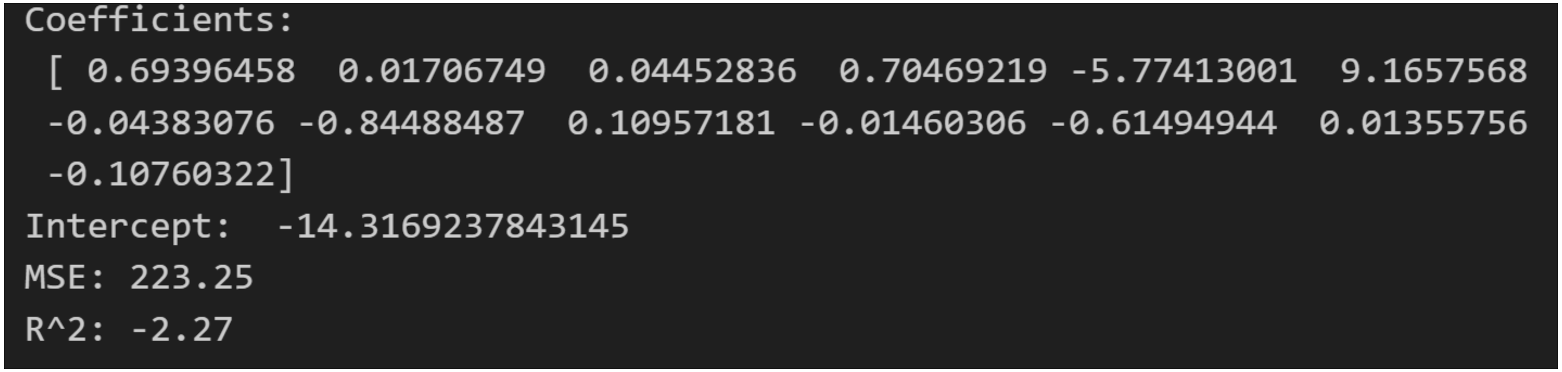
$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

# Multiple linear regression - Python



```python
import matplotlib.pyplot as plt
import numpy as np

from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score;

bx, by = datasets.load_boston(return_X_y=True)

bx_train = bx[:-146] #75-25% training-test
bx_test = bx[-146:]

by_train = by[:-146]
by_test = by[-146:]

mlm = linear_model.LinearRegression()

# Train model
mlm.fit(bx_train, by_train)

# Predict on test
by_pred = mlm.predict(bx_test)

# Regression coefficients and intercept
print("Coefficients: \n", mlm.coef_)
print("Intercept: ", mlm.intercept_)

# Compute MSE
print("MSE: %.2f" % mean_squared_error(by_test, by_pred))
print("R^2: %.2f" % r2_score(by_test, by_pred))
```

# Results



```
Coefficients:
 [ 0.69396458  0.01706749  0.04452836  0.70469219 -5.77413001  9.1657568
 -0.04383076 -0.84488487  0.10957181 -0.01460306 -0.61494944  0.01355756
 -0.10760322]
Intercept:  -14.3169237843145
MSE: 223.25
R^2: -2.27
```

- *Coefficient of determination $R^2$* $$R^2 = 1 - \frac{RSS}{TSS}$$
- $\mathbf{y}_{\mathrm{i}}$ - actual values, $\hat{\mathbf{y}}_{\mathrm{i}}$ or $\mathbf{f}\,(\mathbf{x}_{\mathrm{i}})$- predicted values, $\bar{\mathbf{y}}$ - mean of $\mathbf{y}$
- n - number of records
- RSS – residual sum of squares $$\mathrm{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$
- TSS – total sum of squares $$\mathrm{TSS} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

# Logistic regression

- *m* training data of the type *(X,Y)*:
  - $X = (X^1, X^2, \ldots, X^n)$ input vector
  - *Y* – outcome
    - Y binary variable (2 classes)
  - Generalization to multi-class -> softmax regression
- Classification problem
- Logistic regression
  - Similar form to regression
  - Different sense
  - Prediction upon a transformation of Y

# Logistic regression

- The logit transformation
- p – proportion of records with a certain characteristic
  - Ex.: proportion of pacients with positive diagnosis for a disease
- $\text{logit}(Y) = \ln\left(\frac{p}{1-p}\right)$
  - p – proportion of data of class positive
  - 1-p – proportion of data of the opposite class

- The logit model presumes that the new outcome is a linear combination of the predictive variables
  - $\text{logit}(Y) = \alpha = b_0 + b_1 X^1 + b_2 X^2 + \ldots + b_n X^n$

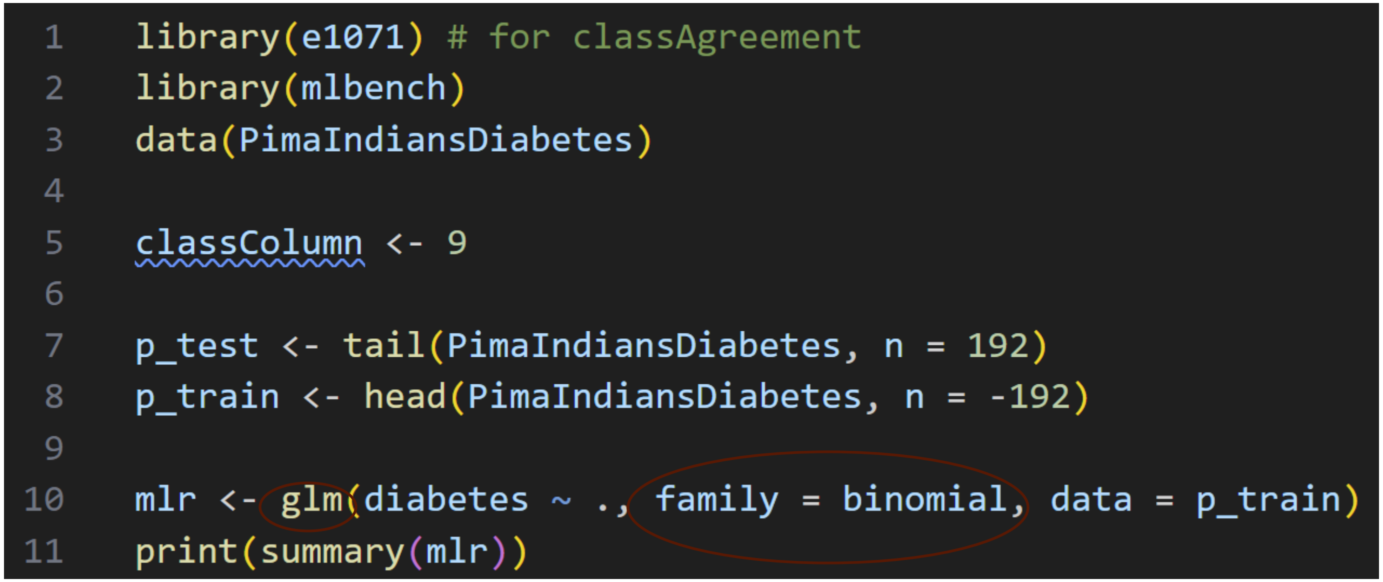
# Logistic regression

- Once the problem is solved
  - Compute $\alpha$ for a new example
  - p= $\frac{e^{\alpha}}{1+e^{\alpha}}$ (the sigmoid (or logistic) function)
  - If p>= 0.5 then class = 1; else class = -1
  - p – the probability to belong to class 1
  - Ex.: probability that a pacient has a positive diagnosis

- Parameter estimation
  - Maximum likelihood method

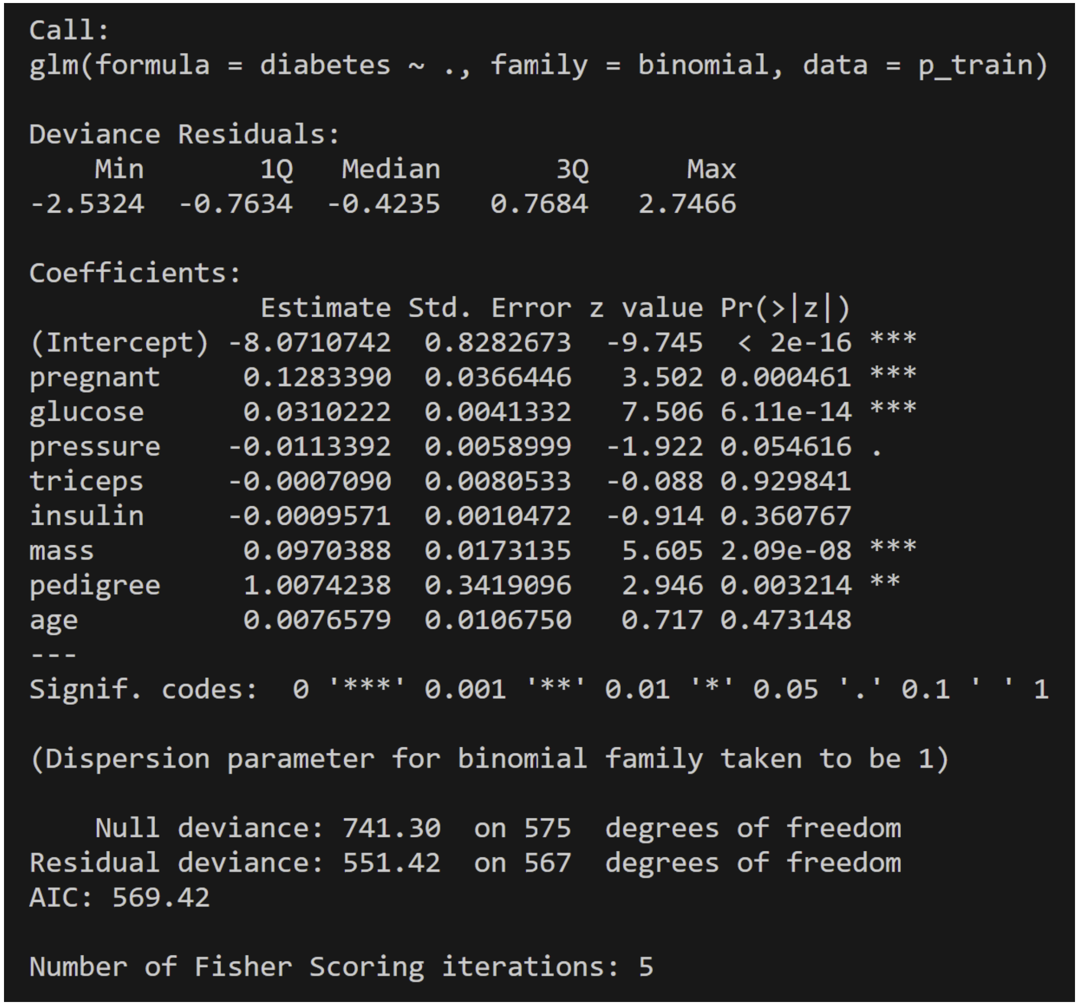
# Function glm() in R

- glm - generalized linear model –non-normal errors, non-constant variance
- Set family = binomial such that R calls logistic regression.



```r
library(e1071) # for classAgreement
library(mlbench)
data(PimaIndiansDiabetes)

classColumn <- 9

p_test <- tail(PimaIndiansDiabetes, n = 192)
p_train <- head(PimaIndiansDiabetes, n = -192)

mlr <- glm(diabetes ~ ., family = binomial, data = p_train)
print(summary(mlr))
```

# Results and interpretation (1/5)

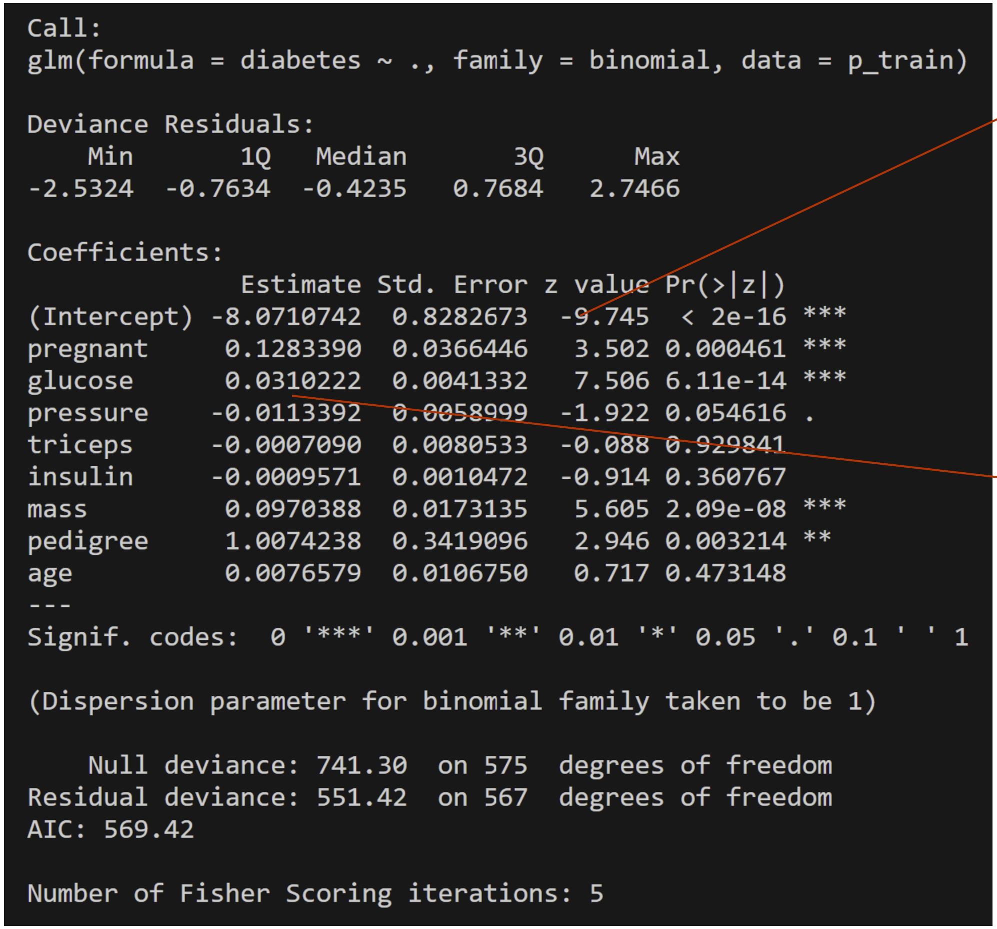


```
Call:
glm(formula = diabetes ~ ., family = binomial, data = p_train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.5324  -0.7634  -0.4235   0.7684   2.7466  

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -8.0710742  0.8282673  -9.745  < 2e-16 ***
pregnant     0.1283390  0.0366446   3.502 0.000461 ***
glucose      0.0310222  0.0041332   7.506 6.11e-14 ***
pressure    -0.0113392  0.0058999  -1.922 0.054616 .  
triceps     -0.0007090  0.0080533  -0.088 0.929841    
insulin     -0.0009571  0.0010472  -0.914 0.360767    
mass         0.0970388  0.0173135   5.605 2.09e-08 ***
pedigree     1.0074238  0.3419096   2.946 0.003214 ** 
age          0.0076579  0.0106750   0.717 0.473148    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 741.30  on 575  degrees of freedom
Residual deviance: 551.42  on 567  degrees of freedom
AIC: 569.42

Number of Fisher Scoring iterations: 5
```

- Model is:

logit(Y)=-8.07 + 0.12*pregnant + 0.03*glucose - 0.01*pressure + 0.09*mass + pedigree + 0.007 * age

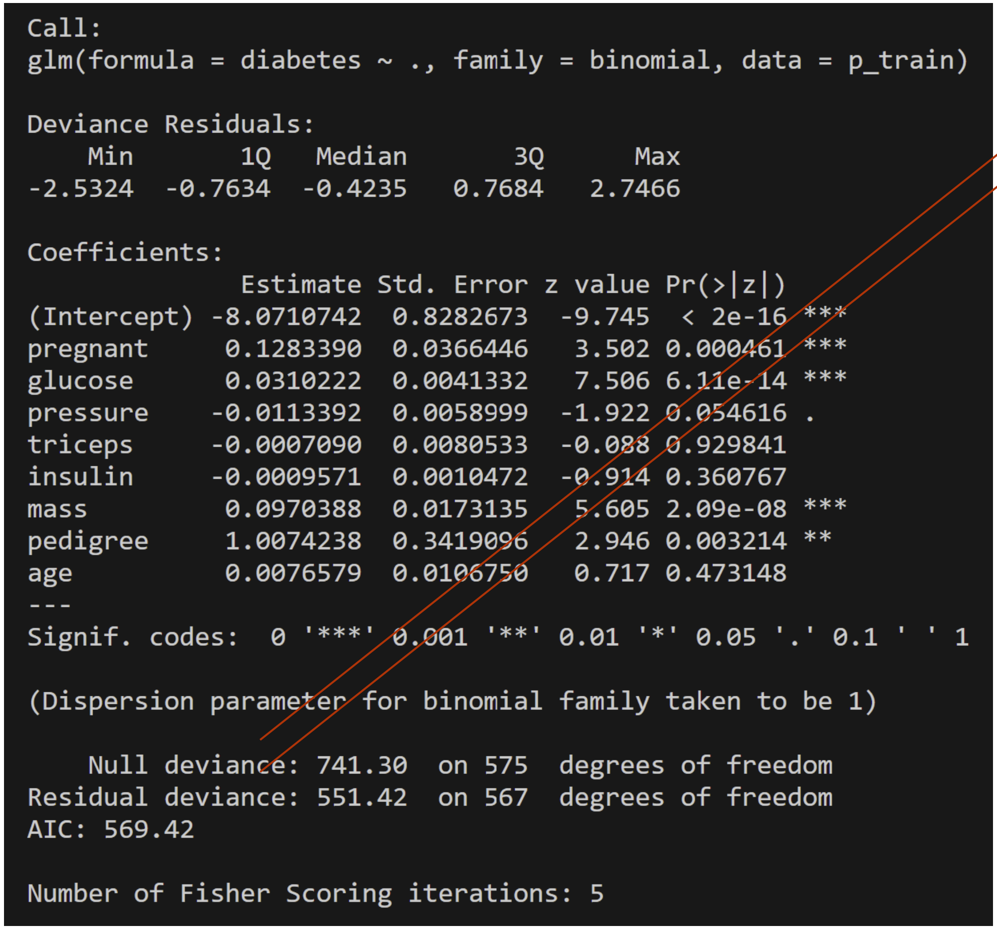
# Results and interpretation (2/5)



```
Call:
glm(formula = diabetes ~ ., family = binomial, data = p_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5324  -0.7634  -0.4235   0.7684   2.7466

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.0710742  0.8282673  -9.745  < 2e-16 ***
pregnant     0.1283390  0.0366446   3.502 0.000461 ***
glucose      0.0310222  0.0041332   7.506 6.11e-14 ***
pressure    -0.0113392  0.0058999  -1.922 0.054616 .
triceps     -0.0007090  0.0080533  -0.088 0.929841
insulin     -0.0009571  0.0010472  -0.914 0.360767
mass         0.0970388  0.0173135   5.605 2.09e-08 ***
pedigree     1.0074238  0.3419096   2.946 0.003214 **
age          0.0076579  0.0106750   0.717 0.473148
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 741.30  on 575  degrees of freedom
Residual deviance: 551.42  on 567  degrees of freedom
AIC: 569.42

Number of Fisher Scoring iterations: 5
```

- Measure of variable importance
  - A z in absolute value ≥2 is significant at the level p = 0.05.
- The coefficient for a variable (as argument of the exponential function) gives the proportional change in the response for a one unit change in the variable value.

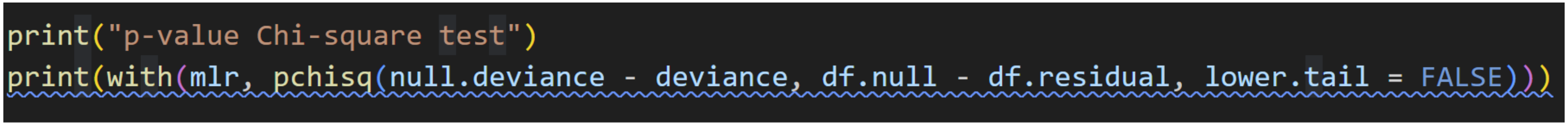
# Results and interpretation (3/5)



```
Call:
glm(formula = diabetes ~ ., family = binomial, data = p_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5324  -0.7634  -0.4235   0.7684   2.7466

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.0710742  0.8282673  -9.745  < 2e-16 ***
pregnant     0.1283390  0.0366446   3.502 0.000461 ***
glucose      0.0310222  0.0041332   7.506 6.11e-14 ***
pressure    -0.0113392  0.0058999  -1.922 0.054616 .
triceps     -0.0007090  0.0080533  -0.088 0.929841
insulin     -0.0009571  0.0010472  -0.914 0.360767
mass         0.0970388  0.0173135   5.605 2.09e-08 ***
pedigree     1.0074238  0.3419096   2.946 0.003214 **
age          0.0076579  0.0106750   0.717 0.473148
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 741.30  on 575  degrees of freedom
Residual deviance: 551.42  on 567  degrees of freedom
AIC: 569.42

Number of Fisher Scoring iterations: 5
```

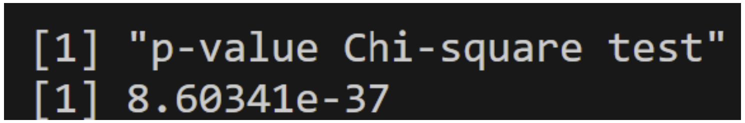
- The fit of the model can be determined through the difference regarding the residual deviance between a model with predictors vs the null model (only with intercept).
- The chi-square test is applied, with the degrees of freedom given by the difference between those of the current model and those of a null one (i.e. number of predictive variables).

# Results and interpretation (4/5)



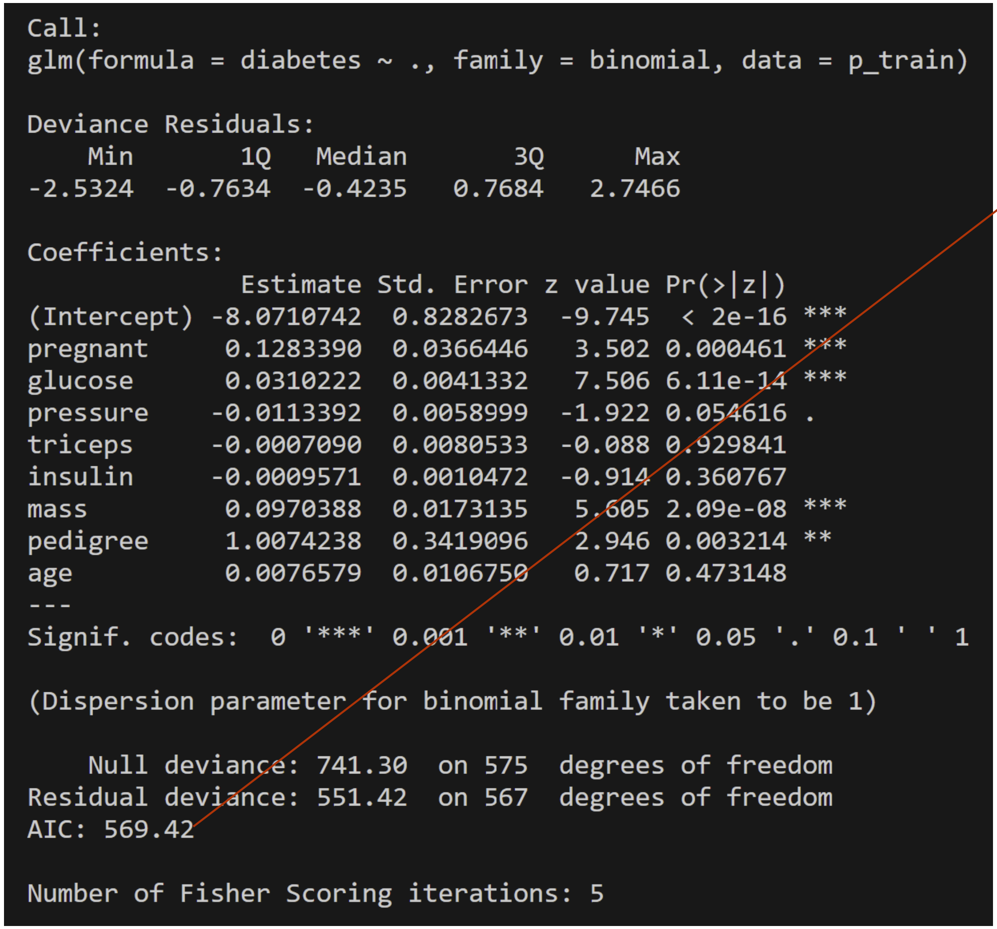
```
print("p-value Chi-square test")
print(with(mlr, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE)))
```

- The obtained *p* confirms that the model with predictors is significantly better than a null one.



```
[1] "p-value Chi-square test"
[1] 8.60341e-37
```

# Results and interpretation (5/5)

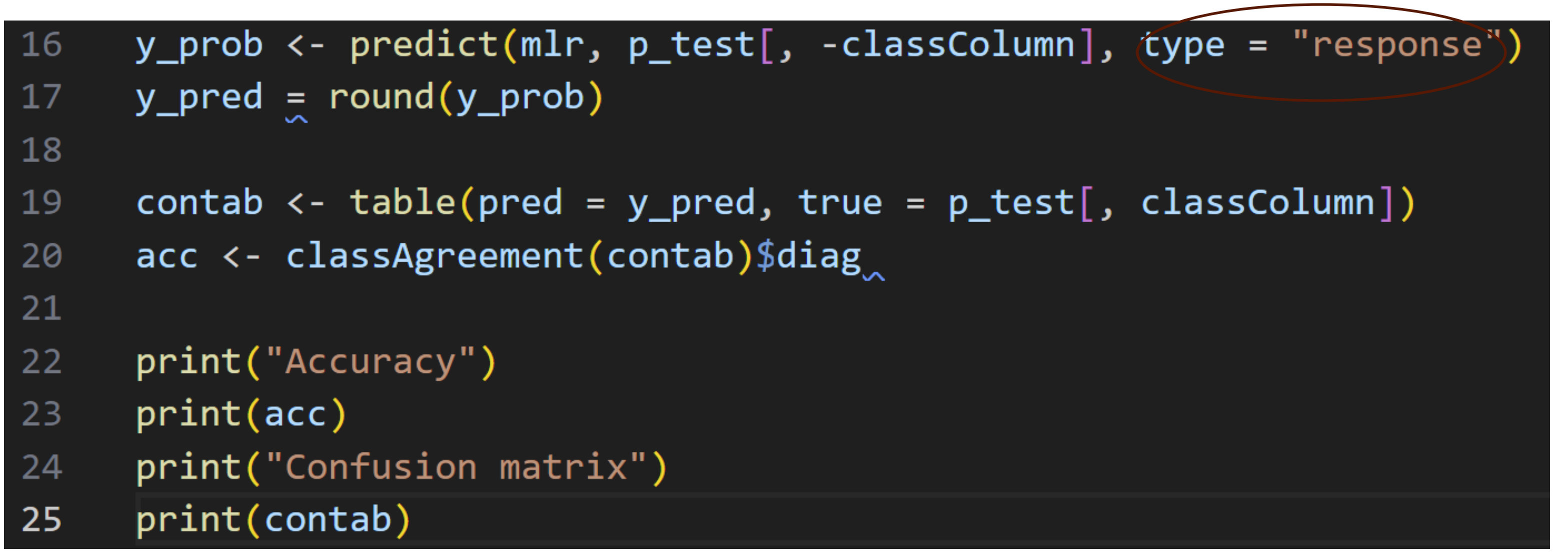


```
Call:
glm(formula = diabetes ~ ., family = binomial, data = p_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5324  -0.7634  -0.4235   0.7684   2.7466

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.0710742  0.8282673  -9.745  < 2e-16 ***
pregnant     0.1283390  0.0366446   3.502 0.000461 ***
glucose      0.0310222  0.0041332   7.506 6.11e-14 ***
pressure    -0.0113392  0.0058999  -1.922 0.054616 .
triceps     -0.0007090  0.0080533  -0.088 0.929841
insulin     -0.0009571  0.0010472  -0.914 0.360767
mass         0.0970388  0.0173135   5.605 2.09e-08 ***
pedigree     1.0074238  0.3419096   2.946 0.003214 **
age          0.0076579  0.0106750   0.717 0.473148
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 741.30  on 575  degrees of freedom
Residual deviance: 551.42  on 567  degrees of freedom
AIC: 569.42

Number of Fisher Scoring iterations: 5
```

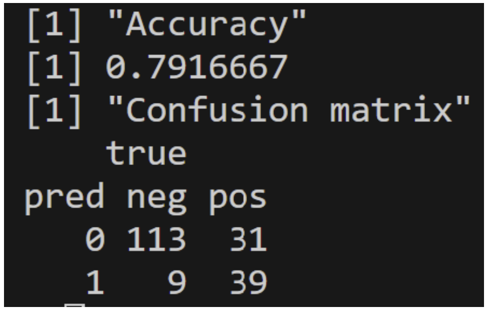
$$AIC = 2k - 2\ln(\hat{L})$$

- AIC (Akaike information criterion) computes the fit of the model to the data:
  - k – number of estimated parameters
  - $L_{hat}$ – maximum value for likelihood function
- The AIC can be compared to that of other models, the best being the one with the minimum value:
  - explaining the largest amount of variation with the fewest possible predictors

# Prediction on test data: p>=0.5, class = 1 (poz); else = 0 (neg)

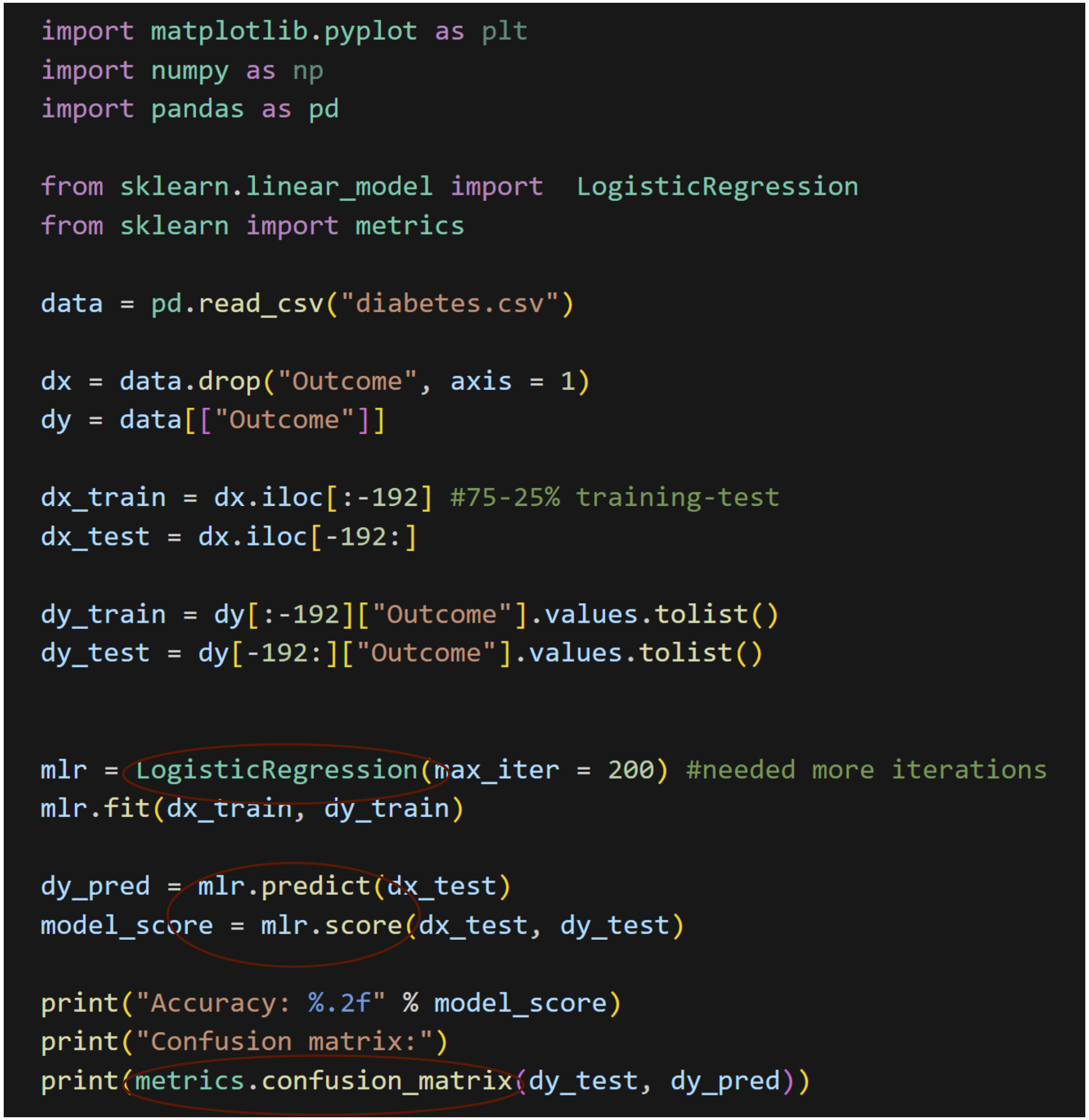


```r
y_prob <- predict(mlr, p_test[, -classColumn], type = "response")
y_pred = round(y_prob)

contab <- table(pred = y_pred, true = p_test[, classColumn])
acc <- classAgreement(contab)$diag

print("Accuracy")
print(acc)
print("Confusion matrix")
print(contab)
```



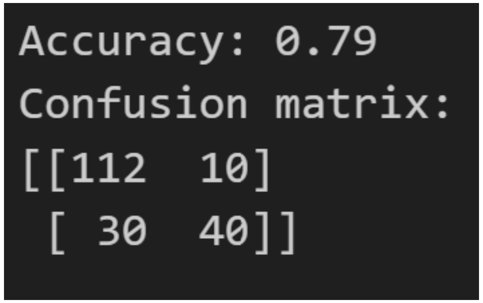
```
[1] "Accuracy"
[1] 0.7916667
[1] "Confusion matrix"
    true
pred neg pos
   0 113  31
   1   9  39
```

# Python



```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

from sklearn.linear_model import  LogisticRegression
from sklearn import metrics

data = pd.read_csv("diabetes.csv")

dx = data.drop("Outcome", axis = 1)
dy = data[["Outcome"]]

dx_train = dx.iloc[:-192] #75-25% training-test
dx_test = dx.iloc[-192:]

dy_train = dy[:-192]["Outcome"].values.tolist()
dy_test = dy[-192:]["Outcome"].values.tolist()


mlr = LogisticRegression(max_iter = 200) #needed more iterations
mlr.fit(dx_train, dy_train)

dy_pred = mlr.predict(dx_test)
model_score = mlr.score(dx_test, dy_test)

print("Accuracy: %.2f" % model_score)
print("Confusion matrix:")
print(metrics.confusion_matrix(dy_test, dy_pred))
```

# Results



```
Accuracy: 0.79
Confusion matrix:
[[112  10]
 [ 30  40]]
```

# How to handle categorical variables in linear models?

*non linear relationship exists*

Linear Regression:

# Homework

WHEN YOU ADVERTISE, IT'S ARTIFICIAL INTELLIGENCE. WHEN YOU HIRE, IT'S MACHINE LEARNING.
WHEN YOU IMPLEMENT, IT'S LINEAR REGRESSION.
makeameme.org

- Implement the appropriate linear model for either:
  - predicting the amount of the tip (tip) for a restaurant meal according to problem 7.1 Tips, pp 153 (Cook, Swayne, 2007).
  - discriminating between a rock and a classical song (type) according to problem 7.12 Music, pp. 171 (Cook, Swayne, 2007).
- (Optional) Choose a data set of interest and apply either a linear regression or a logistic regression model.

Dianne Cook, Deborah F. Swayne, Graphics for Data Analysis. Interactive and Dynamic With R and Ggobi, Springer, 2007