Lab 7

(cont…)

Goodness-of-fit for linear regression

"All models are wrong, but some are useful."

George Box (1919-2013), pioneering statistician

How well does a model fit our data? In many cases, answers are based on analysis of the residuals:

-normality of residuals;

-homoscedasticity: constant residual variance throughout the range of the predicted values.

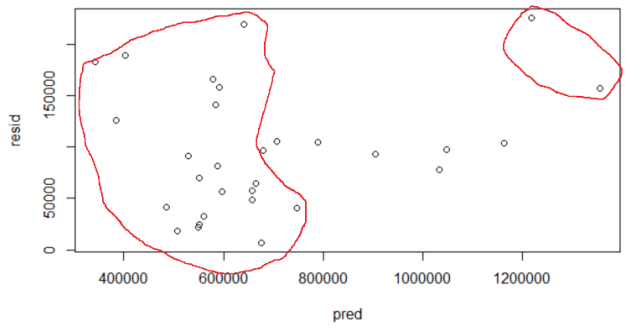
Another measure is that of "variance explained" R^2 (see C11 slide 14).

```
# example adapted from [1] p. 157
# house_sales.csv taken from
#https://drive.google.com/drive/folders/0B98qpkK5EJemYnJ1ajA1ZVJwMzg

house <- read.csv(file.path('C:\\Users\\Marina\\Desktop\\curs Statistics for Data
Science\\house_sales.csv'), sep='\t')

house_98105 <- house[house$ZipCode == 98105, ]   #select the houses in an
                                                  #area

sample_house_98105<- house_98105[sample(nrow(house_98105),30), ]

model <- lm(AdjSalePrice ~ SqFtTotLiving + SqFtLot + Bathrooms +
            Bedrooms + BldgGrade, data=sample_house_98105)

summary(model)


#Multiple R-squared:  0.8218
#that is R2  -- we say that 82.18% of variability can be explained through the
#linear model – by this measure, the model fits well


shapiro.test(residuals(model))
```

Shapiro-Wilk normality test

[1] Peter Bruce, Andrew Bruce. Practical Statistics for Data Scientists, O'Reilly Media, 2017

```
        data:  residuals(model)
        W = 0.96923, p-value = 0.5184
#normality assumption is met for our model

#Obs. Models are robust to non-normality to a certain extent


# homoscedasticity

resid <- abs(residuals(model))  #interested in the magnitude of the residuals
#names(resid)<-NULL              #to remove the names of the resid vector

pred<-predict(model)

plot(pred,resid)
```



We see a greater variation of residuals for lower-valued homes (left) and for the higher-valued homes (right) and much less variation for the mid-valued homes -- homoscedasticity assumption not met. It may indicate a “cause” missing from the linear equation (causes ->effect) -- worth investigating other (linear) models.

Repeat the same steps for the goodness-of-fit for the example below (taken from Lab6):

```
longley
X <- longley[, "Employed"]     # number of people employed
Y <- longley[,"Population"]    # population ≥ 14 years of age

model1<-lm(X~Y)
```

What is your conclusion after this analysis?

Nonlinear regression

!!Please read C12

```
# we use house_sales.csv to generate the data for the nonlinear regression
#examples

house <- read.csv(file.path('C:\\Users\\Marina\\Desktop\\curs Statistics for Data S
cience\\house_sales.csv'), sep='\t')


house_98105 <- house[house$ZipCode == 98105, ]

sample_house_98105<- house_98105[sample(nrow(house_98105),30), ]
#random selection of 30 rows from the dataset for a clearer visualization of the
#points


lm_30 <- lm(AdjSalePrice ~ SqFtTotLiving + SqFtLot + Bathrooms +
              Bedrooms + BldgGrade, data=sample_house_98105)
resid <- abs(residuals(lm_30))
pred<-predict(lm_30)

#-----------------------------------
#we consider (pred,resid) as (cause,effect) data for nonlinear regression model

plot(pred,resid, ylim = c(-100000,300000))

library(splines)

#interpolation using splines
func = splinefun(x=pred, y=resid, method="natural",ties=mean)

#method= the type of the spline used
#ties = the name of a function specifying how to handle duplicate x values. The y
#values corresponding to the same x value are passed to the function, which
#return a single number (mean in our case)


xmin=min(pred)
xmax=max(pred)  # [xmin,xmax] is the interval for regression

new_points=seq(xmin,xmax,length=50)

allpoints=c(new_points, pred)   #include the knots among the points for prediction
allpoints_ordered = allpoints [order(allpoints)]  #necessary to visualize well the
                                                  #regression curve
```

```
#lines(pred,func(pred), type="p", col=2)

lines(allpoints_ordered,func(allpoints_ordered), type="l", col=2)



#smoothing using splines

fit <- smooth.spline(pred, resid, all.knots=T, spar=0.7)
#try for different values of spar in [0,1]
#spar is alpha parameter in SS(g) – see C12 slide 3
#if alpha=0 the regression curve goes through all the knots – like interpolation
#if alpha=1 the regression curbe becomes smoother and it doesn't go through
#(all) the knots


res <- stats:::predict.smooth.spline(fit, allpoints_ordered)$y
lines(allpoints_ordered,res, type="l", col=4) # plot the fitted spline
```