Lab05: Advanced linear regression - Classification

Author: davide.eynard@gmail.com
Notebook: Didattica
Created: December 1, 2014 9:11:03 AM
Updated: December 15, 2014 12:38:24 AM

---

0) Load the dataset from the previous class and look into it

Credit = read.csv("~/wrk/didattica/2014-2015/201501 - Matteucci PAMI/materiale/Credit.csv")
attach(Credit)
pairs(Credit)

% see what's in credit and take away what we are not interested in (i.e. X)
Credit
C = Credit[,2:7]
fit = lm(Balance ~ ., C)
summary(fit)

% comment on the results: what is interesting and what is not?

1) Extensions of linear regression

 # discrete inputs - that's already built in the model

$$\texttt{balance}_i \approx \beta_0 + \beta_1 \times \texttt{income}_i + \begin{cases} \beta_2 & \text{if } i\text{th person is a student} \\ 0 & \text{if } i\text{th person is not a student} \end{cases}$$

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is Asian} \\ \beta_0 + \beta_2 + \epsilon_i & \text{if } i\text{th person is Caucasian} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is African American.} \end{cases}$$

Credit
C = Credit[,2:11]
fit = lm(Balance ~ ., C)
summary(fit)

NOTE HOW THE VARIABLES HAVE BEEN AUTOMATICALLY SPLIT!
(There will always be one fewer dummy variable than the number of levels. The levels you don't see --e.g. Gender:Male, Student:No, Ethnicity:African American-- are called the *baselines)*

```
GenderFemale          -10.65325    9.91400  -1.075   0.2832
StudentYes            425.74736   16.72258  25.459  < 2e-16 ***
MarriedYes             -8.53390   10.36287  -0.824   0.4107
EthnicityAsian         16.80418   14.11906   1.190   0.2347
EthnicityCaucasian     10.10703   12.20992   0.828   0.4083
```

 # accounting for non-linear relationships

```
fit2 = lm(Balance ~ . + I(Rating^2), C)
summary(fit2)
anova(fit,fit2)
```

```
# you can also try with log:
fit2 = lm(Balance ~ . +log(Rating), C)
summary(fit2)
anova(fit,fit2)
```

```
# also try with poly
fit2 = lm(Balance ~ . + poly(Rating,5), C)
summary(fit2)
anova(fit,fit2)
```

discuss - if we had training and testing sets what would happen?

 # accounting for interactions

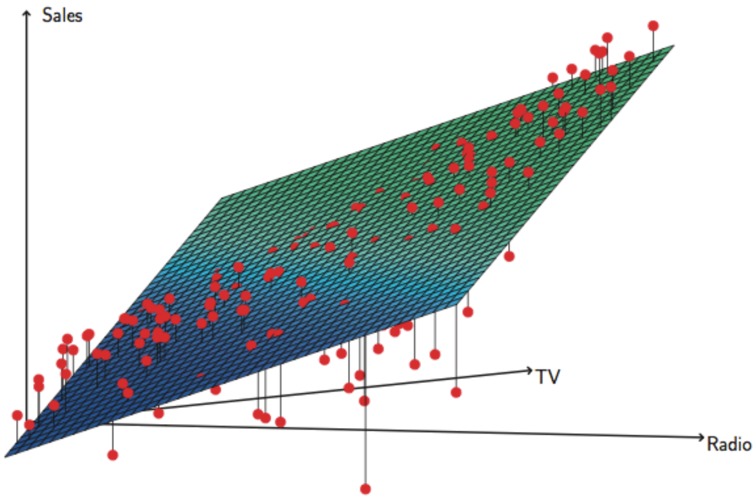

FIGURE 3.5. *For the* Advertising *data, a linear regression fit to* sales *using* TV *and* radio *as predictors. From the pattern of the residuals, we can see that there is a pronounced non-linear relationship in the data. The positive residuals (those visible above the surface), tend to lie along the 45-degree line, where TV and Radio budgets are split evenly. The negative residuals (most not visible), tend to lie away from this line, where budgets are more lopsided.*

```
Ads = read.csv("~/wrk/didattica/2014-2015/201501 - Matteucci PAMI/materiale/Advertising.csv")
Ads = Ads[,2:5]
attach(Ads)
fit = lm(Sales ~ ., Ads)
summary(fit)

# actually we can just include TV and Radio, as Newspapers contribution is negligible
fit = lm(Sales ~ TV + Radio, Ads)
summary(fit)

fit2 = lm(Sales ~ TV + Radio + Tv*Radio)
summary(fit2)
anova(fit, fit2)
```

2) Logistic Regression

Recap: Why not linear regression?

$$Y = \begin{cases} 1 & \text{if } \texttt{stroke;} \\ 2 & \text{if } \texttt{drug overdose;} \\ 3 & \text{if } \texttt{epileptic seizure.} \end{cases}$$

- Logistic regression solves the negative probability (ad other issues as well) by regressing the logistic function

Logistic Regression

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}.$$

from this we derive

This is called *odds*

$$\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

and taking logarithms

This is called *log-odds or logit*

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

Go to http://statweb.stanford.edu/~tibs/ElemStatLearn/ for datasets

```
heart = read.table("http://statweb.stanford.edu/~tibs/ElemStatLearn/datasets/SAheart.data", sep=",",head=T,row.names=1)
```

```
names(heart)
cor(heart[,c(1:4,6:10)])

glm.fit = glm(chd ~ tobacco + age + adiposity + alcohol, family = binomial)
summary(glm.fit)

glm.fit = glm(chd ~ ., data=heart, family = binomial)
summary(glm.fit)

glm.probs = predict(glm.fit,type="response")
max(glm.probs)
which(glm.probs==max(glm.probs))
glm.probs[407]
heart[407,]

# try to evaluate the accuracy of our model
glm.pred = rep(0,462)
glm.pred[glm.probs>.5]=1
table(glm.pred,chd)
mean(glm.pred==chd)

# now do a more realistic test, dividing the dataset into training and testing
datasets
train = rep (FALSE,dim(heart)[1])
train[1:360]=TRUE
heart.test = heart[!train,]

glm.fit = glm(chd ~ ., data=heart, family = binomial, subset = train)
glm.probs = predict(glm.fit, heart.test, type="response")

glm.pred = rep(0,dim(heart.test)[1])
glm.pred[glm.probs>.5]=1
chd.test = chd[!train]
table(glm.pred,chd.test)
mean(glm.pred==chd.test)
```