Linear Models R exercise

Bolker (2006) provides a comprehensive summary of *general* linear models as opposed to the more modern techniques. (see Handout). Essentially general linear models (NOT general*ized* linear models) are the traditional models of normal residual distributions, independent observations, homoscedasticity, and (assumed) lack of any observation error. In R this includes everything that the **lm** function does (simple and multiple least-squares regression), ANOVA, and ANCOVA. The only difference between these is whether the model includes only continuous variables (regression), only factor variables (ANOVA), or both (ANCOVA).

1. Simple linear regression

All modeling approaches in R use the same basic structure of "response ~ explanatory variable(s)".
Bank customers' loan defaulting patterns are often associated with their age
Level of education, job hoping, number of years living at current address, Household income etc. Here's an example of a model of bank loan defaulting.

First read in the dataset `bankloan.sav` `from the spss Samples folder` *`C:\PROGRAM FILES\IBM\SPSS\STATISTICS\24\SAMPLES.`*

```
>bankloan<-read.spss(file.choose(),to.data.frame=T,use.value.labels=T)
```

This will create a dataframe named "bankloan" which 12 fields. Some fields are numerical and others are categorical. Check.

```
> bankloan
```

```
    age                         ed employ address income debtinc  creddebt
1    41               Some college     17      12    176     9.3 11.359392
2    27 Did not complete high school   10       6     31    17.3  1.362202
3    40 Did not complete high school   15      14     55     5.5  0.856075
4    41 Did not complete high school   15      14    120     2.9  2.658720
5    24           High school degree    2       0     28    17.3  1.787436
6    41           High school degree    5       5     25    10.2  0.392700
7    39 Did not complete high school   20       9     67    30.6  3.833874
8    43 Did not complete high school   12      11     38     3.6  0.128592
9    24 Did not complete high school    3       4     19    24.4  1.358348
10   36 Did not complete high school    0      13     25    19.7  2.777700
11   27 Did not complete high school    0       1     16     1.7  0.182512
………etc.
```

This file contains information on bank customers which includes:

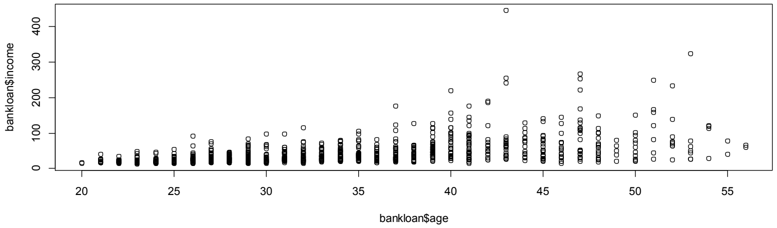
age "Age in years"
Ed "Level of education" (coded 1. Did not complete high school 2. High School Degree 3. Some College 4. College Degree 5. Post –Undergraduate degree)
employ "Years with current employer"
address "Years at current address"
income "Household income in thousands"
debtinc "Debt to income ratio (x100)"
creddebt "Credit card debt in thousands"
othdebt "Other debt in thousands"
default "Previously defaulted" (coded 0. No 1. Yes)

Get some descriptive statistics. Notice the frequency distributions for the categorical variables and the five point summary for the numerical variables.

```
> summary(bankloan)
```

Does income vary with age? Always plot the data first to look at the relationship between the variables

```
>plot(bankloan$age,bankloan$income)
```



There seems to be a relationship.
Now implement a linear model: Income = β0 + β1*age + ε.
Note the syntax that R uses for this type of model:

```
>Income.fit = lm(income~age,data=bankloan)
```

The *lm* function (and many similar functions that we will be using) creates a "model fit" object which holds all sorts of results pertaining to this linear model (obviously you do not need to name the variable **xxx.fit**, but we follow that convention for consistency here). The trick to using *lm* effectively is to know what other functions you have to use to extract the information you want from the fit object.
Use the ANOVA function to obtain an overall test

```
> anova(Income.fit)

Analysis of Variance Table

Response: income
          Df Sum Sq Mean Sq F value    Pr(>F)
age        1 286030  286030  248.72 < 2.2e-16 ***
Residuals 848 975216    1150
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

A summary of the fitted model object is available with the **summary** function:

```
> summary(Income.fit)


Call:
lm(formula = income ~ age, data = bankloan)

Residuals:
   Min     1Q Median     3Q    Max
-61.98 -16.48  -4.88   7.70 381.13

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -33.2808     5.2016  -6.398  2.6e-10 ***
age           2.2825     0.1447  15.771  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1
‘ ’ 1

Residual standard error: 33.91 on 848 degrees of freedom
Multiple R-squared:  0.2268,    Adjusted R-squared:
0.2259
F-statistic: 248.7 on 1 and 848 DF,  p-value: < 2.2e-16
```

All fitted model objects can be explored with **summary**. You get a restatement of the model, the range and quantile estimates of the model residuals, a table of fitted coefficients along with their precision (SE) and significance (t tests), the residual SE (square root of the error variance), the R^2 or coefficient of determination (fraction of variance explained), the adjusted-R^2 (applies a penalty for more fitted parameters), and the associated F statistic evaluating the significance of the full model. There are many attributes of lm, most of which are not shown. For example, you can retrieve a vector of fitted (predicted) values, a vector of residuals, and return the coefficients with **coef**.

Notice the similarity in the results of those two operations. summary() gives you the coefficients and their standard errors. Another way to get the same information is this:

```
> Income.coefs = coef(Income.fit) # the coefficients
> Income.coefs
(Intercept)         age
 -33.280782    2.282541

> Income.vcov = vcov(Income.fit) # the var-covar matrix for the coefs
> Income.vcov
            (Intercept)         age
(Intercept)   27.056706 -0.73377601
age           -0.733776  0.02094743

> Income.coef_se = sqrt(diag(Income.vcov)) # the diagonals of vcov are the variances
> Income.coef_se
(Intercept)         age
  5.2016061   0.1447323
```

Alternatively, note that this summary is itself an object, with components that can be accessed with indexing. For example:

```
>summary(Income.fit)[[4]]

Estimate Std. Error   t value     Pr(>|t|)
```

```
(Intercept) -33.280782  5.2016061 -6.398174 2.597806e-10
age          2.282541  0.1447323 15.770785 2.498712e-49
```

returns the 4th element of summary(lm1), which is stored as a list. Here this element is the coefficient table. We could also show these results as an ANOVA-type table:

```
>summary.aov(Income.fit)
```

```
Df Sum Sq Mean Sq F value Pr(>F)
age          1 286030  286030   248.7 <2e-16 ***
Residuals   848 975216    1150
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

In fact, all lm fitted objects can be returned in both summary forms (summary.lm and summary.aov). Other helpful related items are a report of the confidence intervals of each parameter:

```
>confint(Income.fit)
```

```
                    2.5 %     97.5 %
(Intercept) -43.490315 -23.071250
age          1.998466   2.566617
```

and associated AIC:

```
>AIC(Income.fit)
```

```
[1] 8406.597
```

Note that AIC values are particular to a given response vector: a change in any of the values of the response (e.g., transformation, removing an outlier, etc.) results in a non-comparable vector.

Is our model a good model? One easy way to examine model results is with the plot function:
Let’s check that the regression complied with all of our assumptions.

```
>plot(Income.fit)
```

Here you get four graphs (click to go from one panel to the next): 1) residuals vs. fitted values (a good model will show no pattern); 2) the qqnorm plot we saw above (values should be on the dashed line); 3) scale-location graph, indicating heteroscedasticity ; and 4) standardized residuals vs. leverage and Cook's distance, which is handy for identifying outliers. The first thing we notice in our data is that the y data are not continuous and that the residuals are probably not normally distributed (try **shapiro.test**(lm1$resid) to confirm). Would square root transformation help? We could compare the fit of a new model (lm2) with a transformed cover variable, by either calling lm again with new arguments.
To do better we'll have to move to a generalized linear model (**glm**).

```
> shapiro.test(Income.fit$resid)

        Shapiro-Wilk normality test

data:  Income.fit$resid
W = 0.7397, p-value < 2.2e-16
```

In this case our residuals are not normally distributed.

Two or more models can be compared with

```
>AIC (model, model2, model3, …, modelk)
```

The model with the least AIC is recommended.

2. ANOVA / ANCOVA
Now we will work a problem with both continuous regressors and categorical factors. If the predictors are categorical, we use ANOVA. If they are numeric, we use regression. These distinction turns out to be superficial. Both type of analyses can be carried out a generalization of the linear regression, general linear models. In this example, we have 5 categories of Level of education. Income might also depend on educational level.

As usual we want to plot the data. In order to tell which points correspond to which educational level, you can have the data markers be the characters "D", "H", “S”, “C”and "P" (corresponding to the 5 education levels (first letters of the category labes)).

```
> plot(bankloan$income,bankloan$employ,pch=as.character(bankloan$ed))
```

Note that this only works with single characters ('A', 'B', '4', etc., but not '11' or 'BC').
So looking at the data it does seem that there could be an effect of both years of employment and Level of education. Now, in fitting the model, there are a number of different options. Some of them are what you might call 'inappropriate' but we will explore all of them so you will know how to avoid potential pitfalls.

First, fit it like the linear regression in first exercise.

```
>Income.fit= lm(income~employ+ed,data=bankloan)
```

This fit treats Level of Education (*ed*) like a continuous variable (i.e., it takes the values 1, 2, 3, 4 and 5). Sometimes this works out OK but usually it is better to force R to treat Educational level like a categorical factor:

```
>Income.fit2 = lm(income~employ+as.factor(ed),data=bankloan)
```

This fit has an intercept, and there is no term for the effect of Level of Education 1 (or rather, the intercept represents the mean income for Level of Education 1). The mean effects of Level of Education 2, 3, 4 and 5 are expressed as 'contrasts' to the Level of Education 1 effect - just the arithmetic sum of each term plus the intercept.

It is also possible to make a fit without the intercept, so that the 'Education Level' terms give the mean effect for each Level of education category. **Note** the -1 in the model this time:

```
>Income.fit3 = lm(income~employ+as.factor(ed)-1,data=bankloan)
```

The same syntax will also force a pure regression model (i.e., no categorical factors) to run through the origin (i.e., no intercept).

```
>Income.fit4= lm(income~employ+ed-1,data=bankloan)
```

The logic behind this syntax is this: written out, the model is $y = \beta0 + \beta1x1 + \beta2x2$, where the β's are regression parameters and the x's are the predictor variables in the data. This can also be written as the product of two vectors: $X\beta$, where $\beta = [\beta0\ \beta1\ \beta2]$ and $X = [1\ x1\ x2]'$. Notice the 1 as the first term of X, which is necessary to get $1^*\ \beta0 = \beta0$ in the regression model. So putting in -1 in the model specification in R essentially subtracts that leading 1, giving you $0^*\ \beta0 = 0$, or no intercept.

Now use summary() and anova() to perform tests on each of those fit objects. Note which models produce identical results and which produce different results.

Income.fit2 is the standard way of doing things, so focusing on that one, plot the residuals vs. predictors and do a normal q-q plot to ensure the assumptions of the tests are met.

```
>plot (Income.fit2)
```

Finally, it is worth knowing how to do a Tukey test to determine whether the various groups are different from each other in an ANOVA. This test is not really meaningful in an ANCOVA or GLM context, because it only operates on categorical factors. So let's make one more fit using categorical factors only:

```
>Income.fit5= lm(income~as.factor(ed),data=bankloan)
```

The script for the Tukey test is in the stats package, so load that up:

```
>library(stats)
```

Now do the Tukey test (called TukeyHSD because the full name of the test is the Tukey Honest Significant Difference test.

```
>TukeyHSD(aov(Income.fit5))
```

TukeyHSD only works on fit objects created by aov(). aov is basically the same thing as lm but it is specifically written to do ANOVAs with categorical effects only. In fact, aov actually calls lm to do its thing. To convert an lm fit object to an aov fit object, you just have to run aov on the lm fit object. Alternatively, you could do this:

```
>Income.fit6= aov(income~as.factor(ed),data=bankloan)
```

```
>TukeyHSD(Income.fit6)
```

The output for the aov looks more readable. You can choose.
Notice that aov implements ANOVA models and anova produces ANOVA tables from a fit object. anova also does comparisons on other types of fit objects (like likelihood ratio tests on mle fit objects).
The same ideas hold for as many predictors as you can have.

Finally, note that for relatively simple nonlinear relationships it is often more productive to simply transform the dependent variable rather than worry about nonlinearity. Crawley (p. 205) lists these common transformations:

- log y vs x for exponential relationship
- log y vs log x for power
- exp(y) vs. x for logarithmic
- 1/y vs 1/x for asymptotic
- log(p/(1-p)) vs x for proportion data
- sqrt(y) for count data for variance homogenization
- arcsin(y) for percentage data for variance homogenization

References

General linear models (least squares) in R. http://plantecology.syr.edu/fridley/bio793/lm.html

Bolker, B. (2007).Ecological Models and Data in R, PRINCETON UNIVERSITY PRESS, PRINCETON AND OXFORD https://ms.mcmaster.ca

R Manual for Linear models