

# 1 Regression Diagnostics

An excellent review of regression diagnostics is provided in *Overview of Regression Diagnostics*.

Dr. John Fox's **car** package provides advanced utilities for regression modeling. The prestige data set comes with the car package

## avPlots

- Graphs outcome vs predictor variables holding the rest constant (also called partial-regression plots)
- Help identify the effect(or influence) of an observation on the regression coefficient of the predictor variable

```
library(car)
reg1 <-lm(prestige ~ education + income + type, data = Prestige)
avPlots(reg1, id.n=2, id.cex=0.7)
# id.nid most influential observation
# id.cex font size for id.
```

## influenceIndexPlot

- Cook's distance measures how much an observation influences the overall model or predicted values
- Studentized residuals are the residuals divided by their estimated standard deviation as a way to standardized
- Bonferronitest to identify outliers
- Hat-points identify influential observations (have a high impact on the predictor variables)

```
library(car)
reg1 <-lm(prestige ~ education + income + type, data = Prestige)
influenceIndexPlot(reg1, id.n=3)
```

## influencePlot

- **influencePlot** creates a bubble-plot combining the display of Studentized residuals, hat-values, and Cook's distance (represented in the circles).

```
library(car)
reg1 <-lm(prestige ~ education + income + type, data = Prestige)
influencePlot(reg1, id.n=3)
```

## 1.1 Alcohol and Tobacco Data

This example is for exposition only. We will ignore the fact that this may not be a great way of modeling the this particular set of data!

```
alctob <- data.frame( cbind(
Alcohol = c(6.47, 6.13, 6.19, 4.89, 5.63, 4.52,
5.89, 4.79, 5.27, 6.08, 4.02),
Tobacco = c(4.03, 3.76, 3.77, 3.34, 3.47, 2.92,
3.20, 2.71, 3.53, 4.51, 4.56)),
row.names = c("North", "Yorkshire", "Northeast",
"East Midlands", "West Midlands", "East Anglia",
"Southeast", "Southwest", "Wales",
"Scotland", "N. Ireland"))
```

```
alctobwo <- subset(alctob, rownames(alctob)!="N. Ireland")
#without North Ireland

plot(alctob$Tobacco, alctob$Alcohol,
main="Weekly Household Spending on Alcohol vs. Tobacco",
xlab="Tobacco Spending (GBP)",
ylab="Alcohol Spending (GBP)",
pch=16,col="red",cex=1.5,font.lab=2)
#note N. Ireland in the bottom-right
```

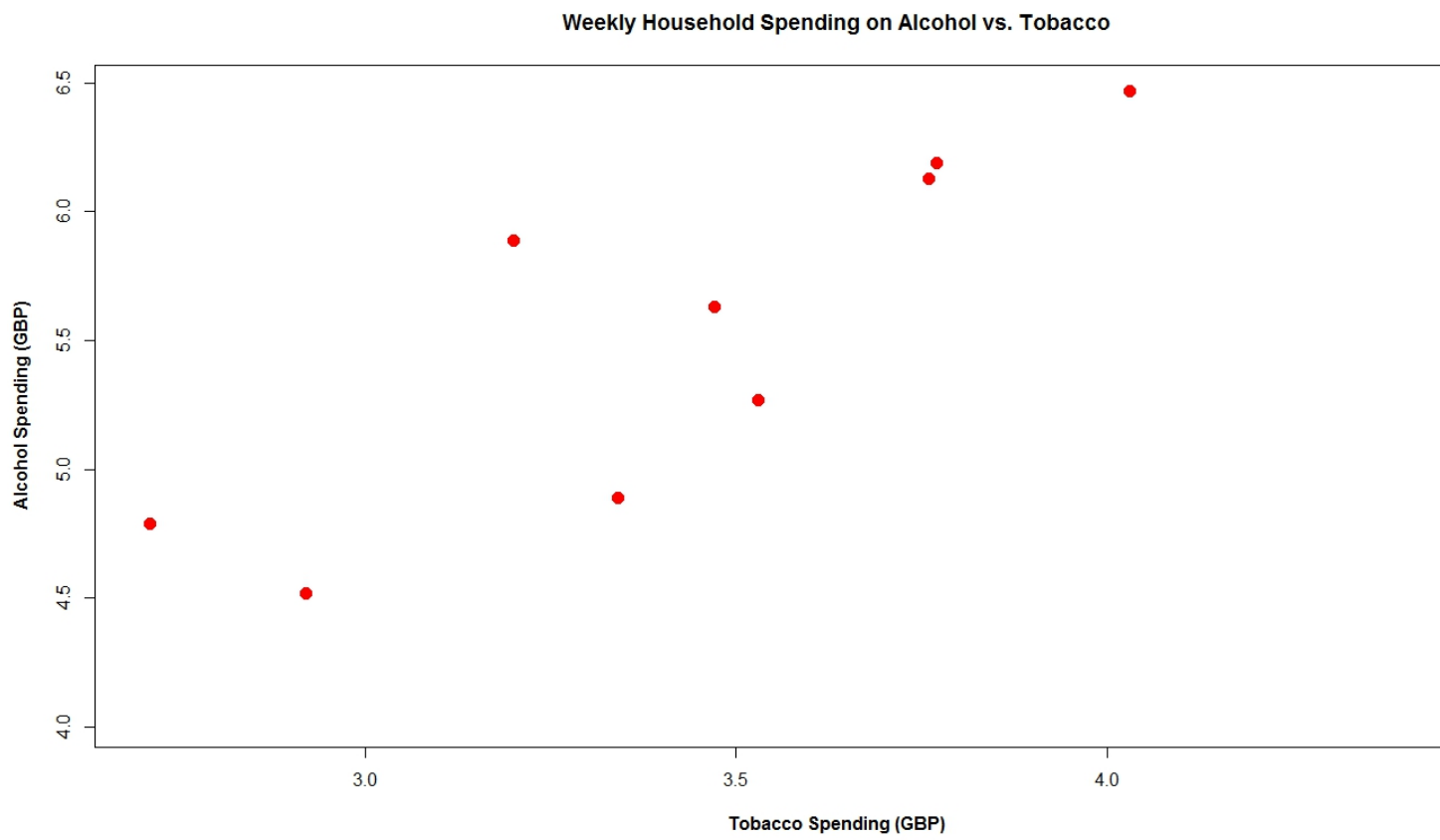


Figure 1:

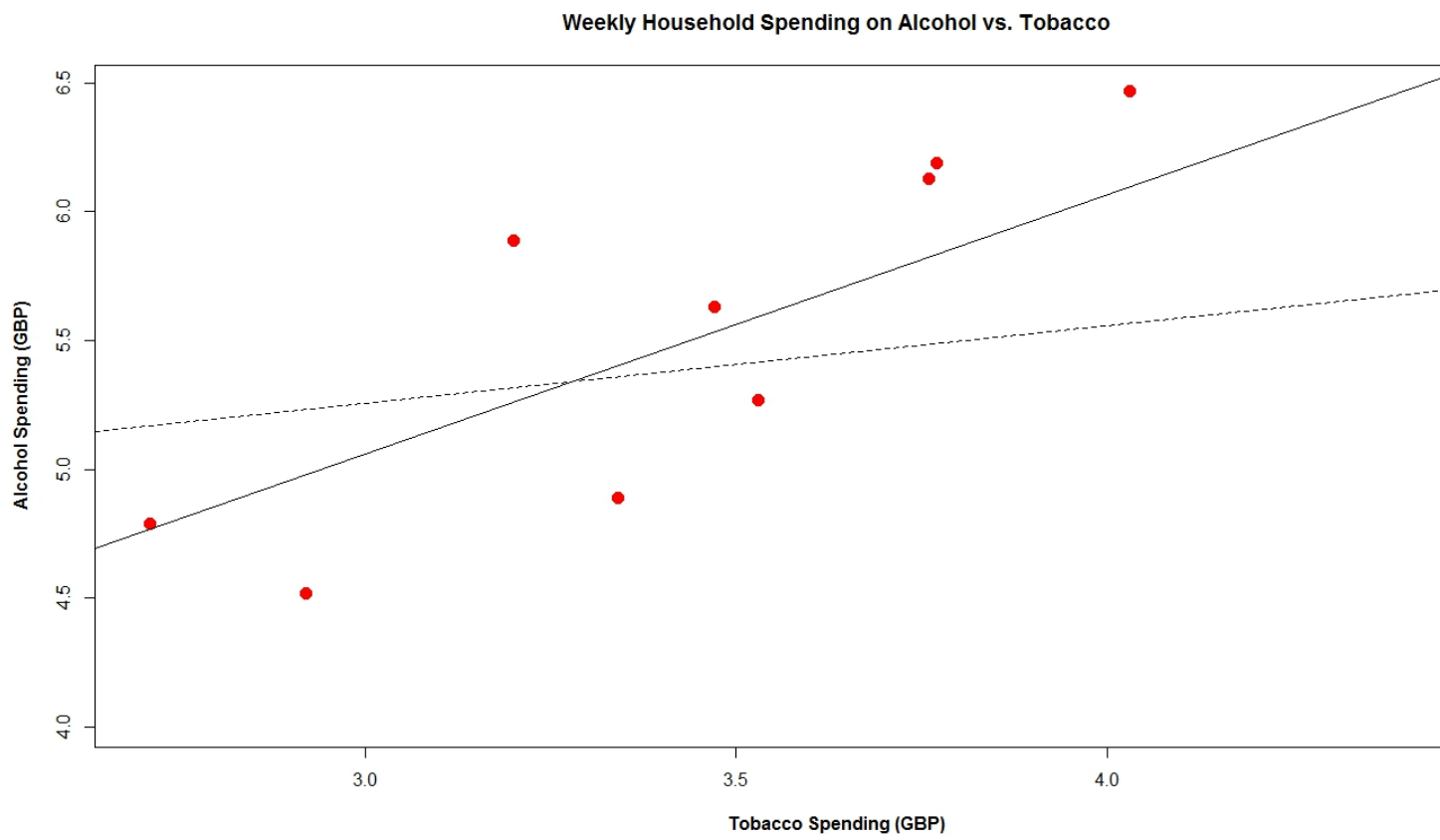


Figure 2:

### All Observations

```
> summary(fit1)

Call:
lm(formula = Alcohol ~ Tobacco, data = alctob)

Residuals:
    Min       1Q   Median       3Q      Max
-1.7080 -0.4245  0.2311  0.6081  0.9020

Coefficients:
(Intercept)  4.3512      1.6067      2.708      0.0241 *
Tobacco       0.3019      0.4388      0.688      0.5087
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 0.8196 on 9 degrees of freedom
Multiple R-squared:  0.04998,    Adjusted R-squared:  -0.05557
F-statistic: 0.4735 on 1 and 9 DF,  p-value: 0.5087
```

### Outlier Removed

```
> summary(fit2)

Call:
lm(formula = Alcohol ~ Tobacco, data = alctobwo)

Residuals:
    Min       1Q   Median       3Q      Max
-0.51092 -0.42434  0.06056  0.34406  0.62991

Coefficients:
(Intercept)  2.0412      1.0014      2.038      0.07586 .
Tobacco       1.0059      0.2813      3.576      0.00723 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 0.446 on 8 degrees of freedom
Multiple R-squared:  0.6151,    Adjusted R-squared:  0.567
```

```
F-statistic: 12.78 on 1 and 8 DF,  p-value: 0.007234
```

## Outliers

The conservative outlier test that we talked about in class uses the Bonferonni inequality to calculate the p-values we associate with the Student's-t test.

In R, we can use the `outlierTest` command to perform this test on our model. Remember that when we test for influence, we are testing the effect of an observation on model coefficients.

Therefore we need to give the `outlierTest` command a linear model as its input.

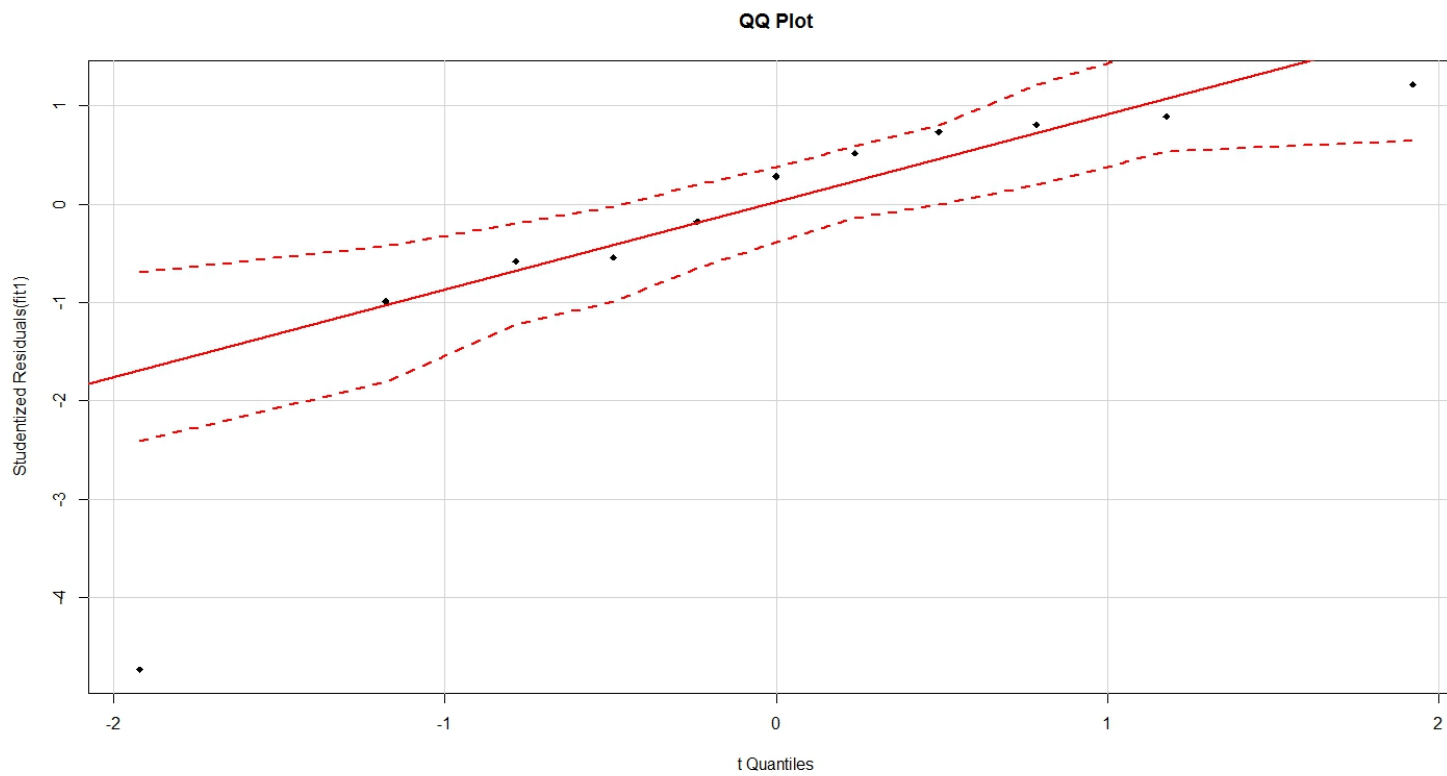
```
> # Assessing Outliers
> # syntax: outlierTest(fit)
> # Bonferonni p-value for most extreme obs
>
> outlierTest(fit1)
rstudent  unadjusted p-value Bonferonni p
N. Ireland -4.732091          0.0014789    0.016268
>
```

We can also use R to calculate Cook's distance.

```
> cooks.distance(fit1)
North      Yorkshire      Northeast      East Midlands
0.114101051 0.036517838      0.043728951 0.023600304
West Midlands East Anglia      Southeast      Southwest
0.004740759 0.147326647      0.046646563 0.077488350
Wales       Scotland      N. Ireland
0.001821694 0.068921892      1.747233521
```

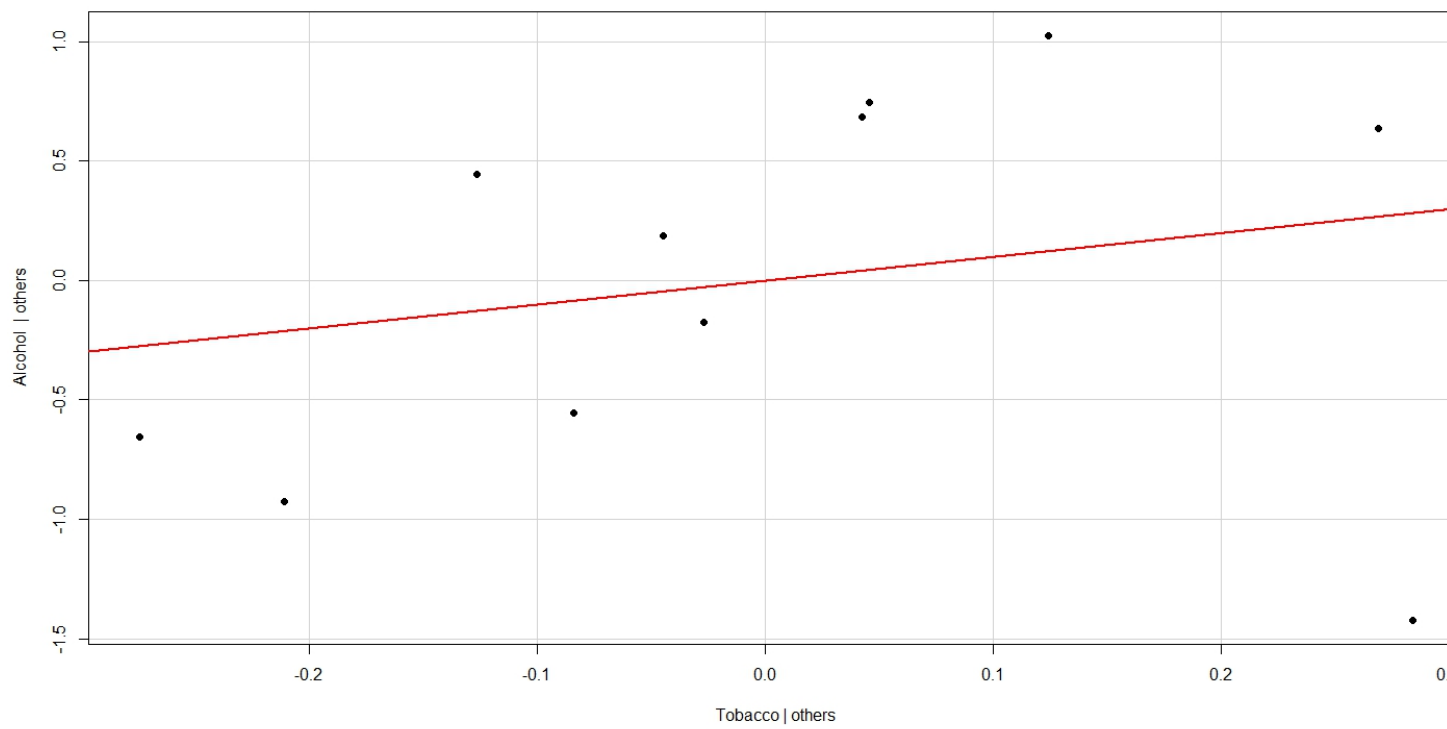
Finally, one of the easier ways to evaluate our residuals and look for influential points is through plots.

```
#qq plot for studentized resid
qqPlot(fit1, main="QQ Plot",pch=18, lim=c(-3,2))
```



```
# leverage plots  
leveragePlots(fit1)
```

```
Influential Observations  
# Influential Observations  
# added variable plots  
avPlots(fit1)  
  
# Cook's D plot  
# identify D values > 4/(n-k-1)  
cutoff <- 4/((nrow(mtcars)-length(fit1$coefficients)-2))  
plot(fit, which=4, cook.levels=cutoff)
```





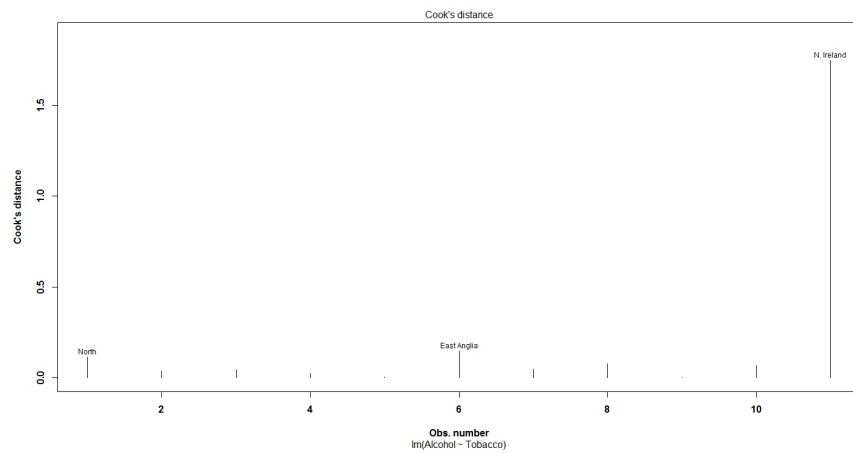


Figure 3:

```
# Influence Plot
influencePlot(fit1,id.method="identify",
main="Influence Plot",
col="red"
sub="Circle size is propoertial to Cook's Distance" )
```

## Non-normality

```
# Normality of Residuals
# qq plot for studentized resid
qqPlot(fit1, main="QQ Plot")
```

```
# distribution of studentized residuals
library(MASS)
sresid <- studres(fit)
hist(sresid, freq=FALSE,
main="Distribution of Studentized Residuals")
```

```
xfit<-seq(min(sresid),max(sresid),length=40)
yfit<-dnorm(xfit)
lines(xfit, yfit)
```

## Non-constant Error Variance

```
# Evaluate homoscedasticity
# non-constant error variance test
ncvTest(fit)
# plot studentized residuals vs. fitted values
spreadLevelPlot(fit)
```

## Multi-collinearity

```
# Evaluate Collinearity
vif(fit) # variance inflation factors
sqrt(vif(fit)) > 2 # problem?
```

## 1.2 Nonlinearity

```
# Evaluate Nonlinearity
# component + residual plot
crPlots(fit)
# Ceres plots
ceresPlots(fit)

# Non-independence of Errors
# Test for Autocorrelated Errors
durbinWatsonTest(fit)
```

## 1.3 Additional Diagnostic Help

The `gvlma( )` function in the `gvlma` package, performs a global validation of linear model assumptions as well separate evaluations of skewness, kurtosis, and heteroscedasticity.

```
# Global test of model assumptions
library(gvlma)
gvmodel <- gvlma(fit)
summary(gvmodel)
```