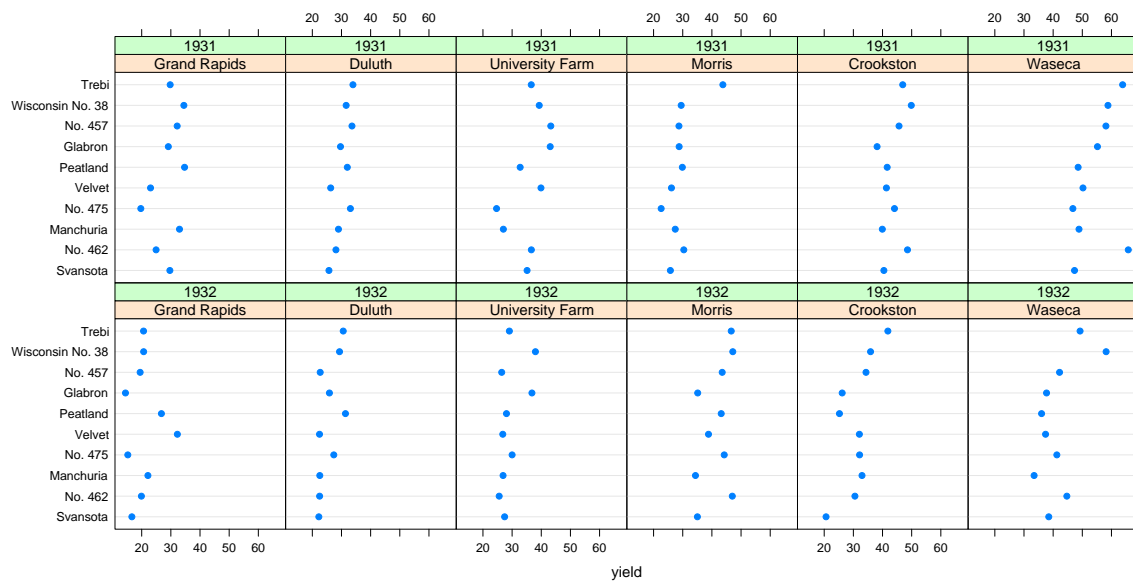


# Leftovers

Lada Adamic

SI 544

## 1 Trellis graphics



The trellis (lattice) package allows one to do some visual exploratory data analysis fairly easily, especially when one has several explanatory variables. Install and load the package :

```
library(lattice)
```

See a quick demo of the many different kinds of plots you can produce

```
example(xyplot)
```

The commands for these are listed one by one in your R script. Let's just list one here, it is the "Total yield in bushels per acre for 10 varieties at 6 sites in each of two years."

```
dotplot(variety ~ yield | site*year, data=barley)
```

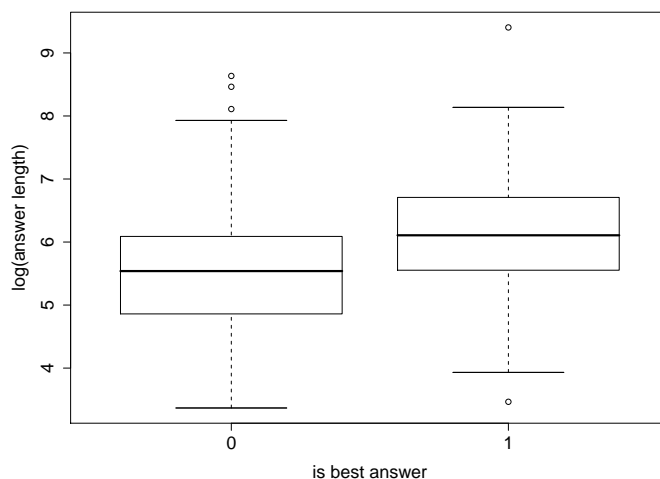
## 2 Logistic regression

Sometimes the outcome you want to predict is not continuous, but rather binary - whether someone will purchase a product, whether someone will win a game, etc. What we are modeling is the log of the odds of a given event occurring

$$\log(\text{odds}) = \log(p/(1-p)) = \log(p) - \log(1-p) = \alpha + \beta_1 X_1 + \beta_2 X_2 + \dots$$

Here we will be trying to figure out whether an answer will be selected as the best answer based on the following variables:

- the length of the answer
- number of other answers supplied by the user
- number of best answers supplied by the user
- number of other people who have answered the same question



```
> ya = data.frame(ya)
> pd.glm = glm(is_best_ans ~ ans_length+num_replies+num_user_ans + num_user_best,
+ family=binomial,data=ya)
> summary(pd.glm)
```

Call:

```
glm(formula = is_best_ans ~ ans_length + num_replies + num_user_ans +
    num_user_best, family = binomial, data = ya)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-3.1835	-0.9936	-0.1726	1.0530	2.2002

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	0.4981808	0.1939247	2.569	0.0102 *
ans_length	0.0010072	0.0001825	5.519	3.40e-08 ***
num_replies	-0.2700685	0.0399614	-6.758	1.40e-11 ***

```
num_user_ans -0.0231300 0.0049542 -4.669 3.03e-06 ***
num_user_best 0.0582897 0.0085216 6.840 7.91e-12 ***
```

---

```
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 1340.5 on 966 degrees of freedom
Residual deviance: 1116.5 on 962 degrees of freedom
AIC: 1126.5
```

```
Number of Fisher Scoring iterations: 5
```

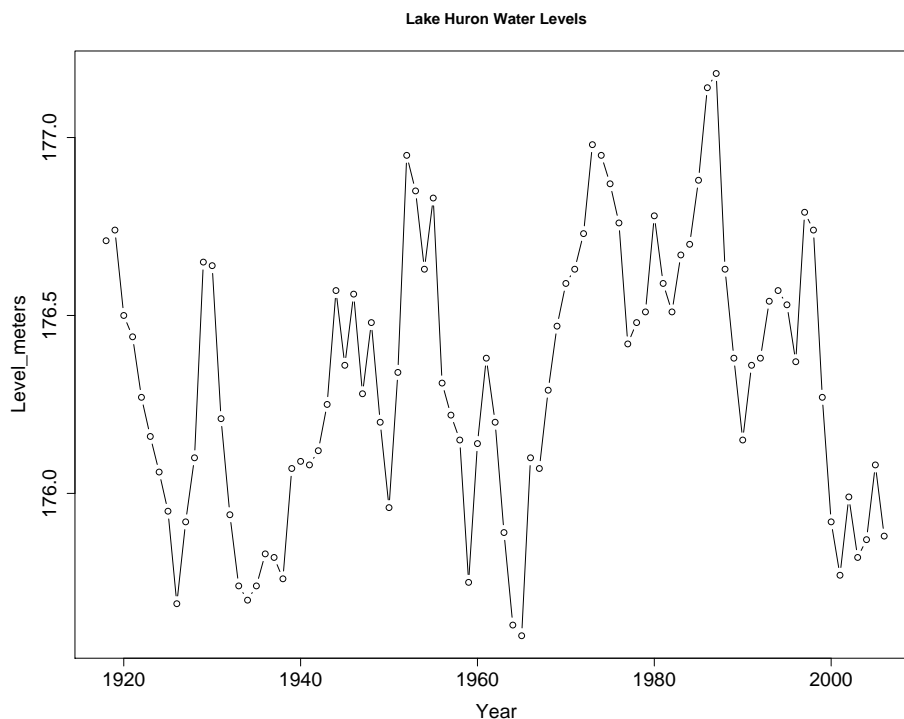
```
>
> cv.binary(pd.glm)
```

```
Fold: 6 2 9 1 7 3 4 10 8 5
Internal estimate of accuracy = 0.733
Cross-validation estimate of accuracy = 0.736
```

So we can get roughly 73% accuracy, where our odds of just guessing randomly whether an answer will be selected as a best answer or not was only 50%.

### 3 Time series analysis

Let's return to the water levels in lake Huron. First, we would like to know if water levels have in fact been declining since 1970, as the recent newspaper article claimed. We already have a tool for this, which is just simple linear regression.



```

> huron = read.table(file=file.choose(),head=T)
> attach(huron)

> summary(lm(Level_meters ~ Year, data = huron[Year >= 1970,]))

Call:
lm(formula = Level_meters ~ Year, data = huron[Year >= 1970, ])

Residuals:
    Min       1Q   Median       3Q      Max
-0.42784 -0.20256 -0.02534  0.13243  0.65938

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 222.332798   8.355645  26.609 < 2e-16 ***
Year        -0.023056   0.004203  -5.486 3.68e-06 ***
Residual standard error: 0.273 on 35 degrees of freedom
Multiple R-Squared:  0.4623, Adjusted R-squared:  0.4469
F-statistic: 30.09 on 1 and 35 DF,  p-value: 3.682e-06

```

Indeed, it appears that water levels have been dropping at a rate of 2.3cm (approx 1") a year, since 1970. But how much is this attributable to a carefully selected starting year? What if we consider the entire period (1918-2006)?

```

> summary(lm(Level_meters ~ Year, data = huron))

Call:
lm(formula = Level_meters ~ Year, data = huron)

Residuals:
    Min       1Q   Median       3Q      Max
-0.73285 -0.25993 -0.01501  0.28574  0.77932

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.703e+02  3.028e+00  56.228 <2e-16 ***
Year         3.083e-03  1.543e-03   1.998  0.0489 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 0.374 on 87 degrees of freedom
Multiple R-Squared:  0.04387, Adjusted R-squared:  0.03288
F-statistic: 3.992 on 1 and 87 DF,  p-value: 0.04885

```

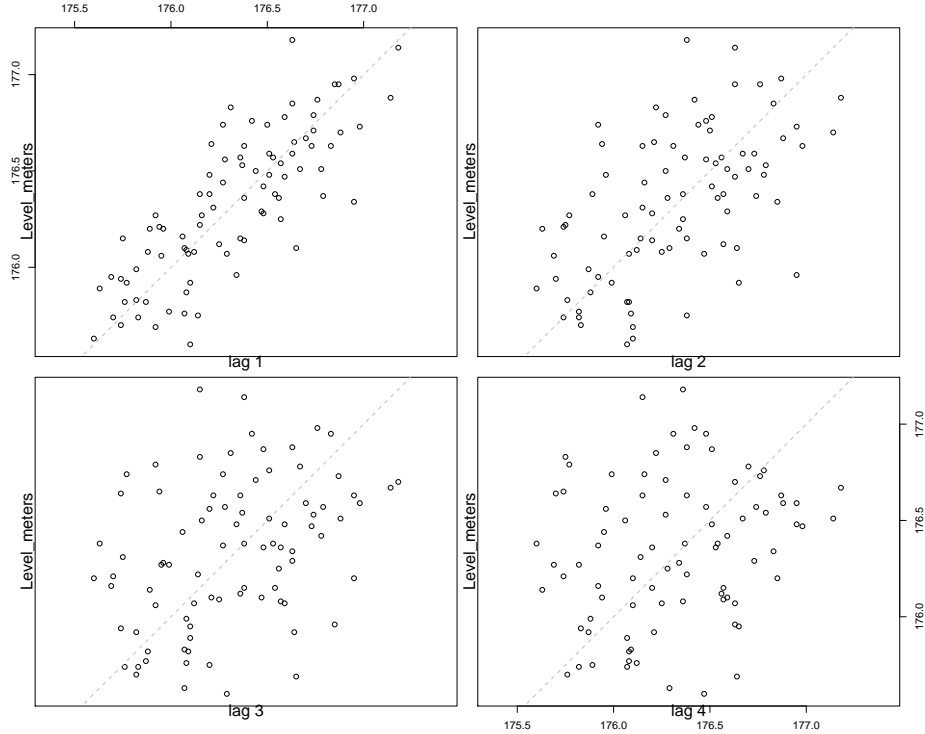
Clearly, if we take the entire period into account, there isn't a significant trend. So there is no support for the hypothesis that water levels have been dropping throughout.

But what we can see is that the water level in any given year appears to be correlated with the water level in the previous year. This is what is called autocorrelation, and we can run an autoregression to get at whether this is in fact the case, or whether year to year levels are uncorrelated.

First let's look at some lag plots. When the lag is 1, we are plotting the level in one year against the level in the year directly preceding it. When the lag is 2, we are looking at water levels two years apart, etc.

```
lag.plot(Level_meters,lags=4,do.lines=FALSE)
```

We can see that year-to-year the water levels do appear to be correlated, and even levels 2 years apart. Beyond 2 years of lag, the correlation weakens further.



We can use the **ACF** (autocorrelation function), to compute the autocorrelation coefficient for all possible lags.

And finally, we can perform the autoregression which models the observation at time  $t$  against observations at times  $t - 1$ ,  $t - 2$ , ...

$$X_t = \mu + \alpha_1(X_{t-1} - \mu) - \alpha_2(X_{t-2} - \mu) + \epsilon_t \quad (1)$$

Where  $\epsilon_t$  is a random variable (the noise term) with zero mean.

```
> ar(Level_meters,method="mle")
```

Call:

```
ar(x = Level_meters, method = "mle")
```

Coefficients:

```
      1      2
0.9867 -0.2264
```

```
Order selected 2  sigma^2 estimated as  0.04812
```

What R is telling us is that the best estimate is

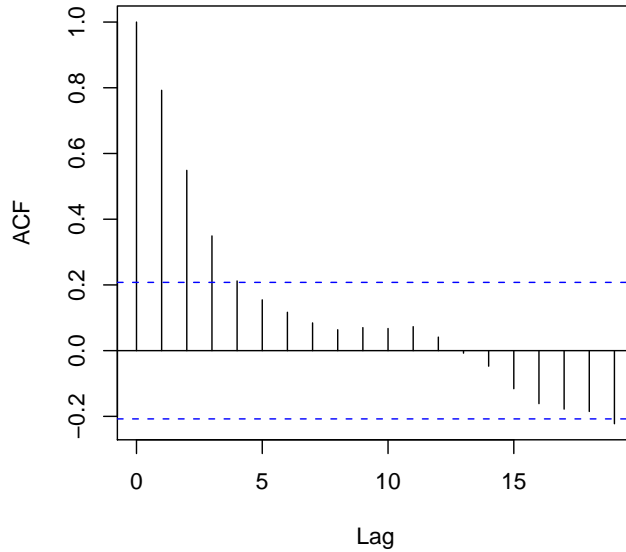
$$X_t = \mu + 0.9867(X_{t-1} - \mu) - 0.2264(X_{t-2} - \mu) + \epsilon_t \quad (2)$$

Sometimes, people who model (microeconomic) time series want to know whether the series has a “unit root”, that is, whether our  $\alpha_1 = 0.987$  is indistinguishable from 1. If that is the case, the “first-order” model we would have would be:

$$X_{t+1} = \mu + X_t + \epsilon_t \quad (3)$$

Such a process is called a random walk with drift, where

### Series Level\_meters



$$X_t = t * \mu + \sum_1^t \epsilon_t + X_0 \quad (4)$$

On the other hand if  $\alpha_1 < 1$  the model will tend to revert to the mean, and any shock to the system in the form of  $\epsilon_t$  will dissipate over time.

We can use either the Augmented Dickey-Fuller test or the Phillips-Perron test. Both tell us that we cannot reject the null hypothesis that there is a “unit root”, and that the lake levels do in fact vary randomly year to year. For these tests, we’ll need the **tseries** R package. If you’d like to do these tests, you’ll need to install the package and the other packages it depends on.

```
> adf.test(Level_meters)
```

Augmented Dickey-Fuller Test

```
data: Level_meters
Dickey-Fuller = -2.5368, Lag order = 4, p-value = 0.3553
alternative hypothesis: stationary
```

```
> pp.test(Level_meters)
```

Phillips-Perron Unit Root Test

```
data: Level_meters
Dickey-Fuller Z(alpha) = -21.087, Truncation lag parameter =
3, p-value = 0.04467
alternative hypothesis: stationary
```