Regression and Correlation

Daniel Lawson University of Bristol

Lecture 02.1

Signposting

- Last time we looked at **Exploratory Data Analysis**.
- Correlation is a description of such data, whilst regression is the first tool to reach for when trying to make sense of such an analysis.
- Regression is a **linear** method and as such, it is usually best considered a form of EDA.

Intended Learning Outcomes

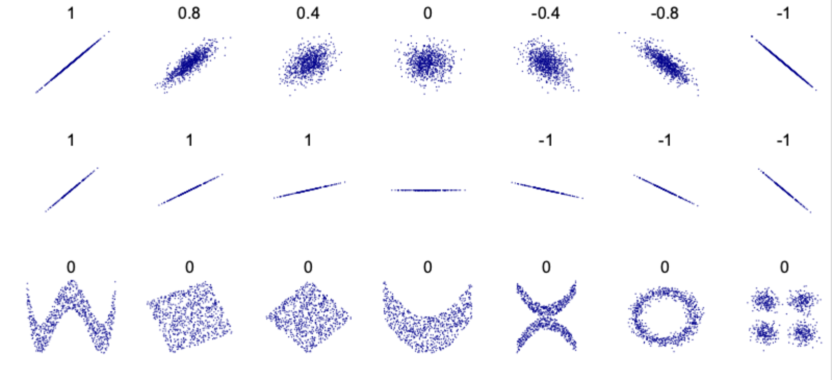
- ILOs used:
  - ILO1 Be able to **access and process cyber security data** into a format suitable for mathematical reasoning
  - ILO2 Be able to **use and apply basic machine learning** tools
  - ILO3 Be able to **make and report appropriate inferences** from the results of applying basic tools to data

Correlation

- The basic relationship between x and y
- A first summary of linear relationships found in the data used to construct a scatterplot.
- How does **variation** in x associate with variation in y?
- **Correlation** describes the observed association between A and B.
  - Correlation examines this relationship in a symmetric manner.
  - Consequently, correlation does not attempt to establish any cause and effect.

Examples

Wikipedia[1]



[1] https://en.wikipedia.org/wiki/Correlation_and_dependence#/media/File:Correlation_exampl

Regression

- **Regression**, considers the relationship of a response variable as determined by one or more explanatory variables.
  - Regression is designed to help **make predictions**.
  - Regression is a often used as a tool to establish causality.
  - A and B share a causal relationship if a regression for B given A, conditional on C (C=**everything else**), has an association
- Since we don't measure **everything else**, regression rarely establishes causality!
- Assumptions are needed to make a causal connection.

Linear algebra view of covariance

- The **covariance matrix** of a random variable X
- Where X is an $n \times 1$ matrix, i.e. a column vector,
- has entries:

$$\mathrm{Cov}(\mathrm{X})_{ij} = \mathrm{Cov}(\mathrm{X}_i, \mathrm{X}_j) = \mathbb{E}[(\mathrm{X}_i - \mu_i)(\mathrm{X}_j - \mu_j)].$$

- It is most naturally defined in matrix form:

$$\Sigma = \mathbb{E}[(\mathrm{X} - \mathbb{E}[\mathrm{X}])(\mathrm{X} - \mathbb{E}[\mathrm{X}])^T],$$

- which is the most straightforward analogy from the scalar version.

Linear algebra view of correlation

- Division by standard deviations is required to correctly generalise the **scalar correlation**:

$$\mathrm{Corr}(X, Y) = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}.$$

- The **matrix form** for correlation is:

$$\mathrm{Corr}(\mathrm{X}) = (\mathrm{diag}(\Sigma))^{-1/2} \Sigma (\mathrm{diag}(\Sigma))^{-1/2}$$

- The matrix inversion is not computationally challenging because it is for a **diagonal matrix**.

Example of correlation

- R code:

```
conncor1=c(linear=cor(conndata2[,'orig_pkts'],
                      conndata2[,'orig_ip_bytes']),
    log=cor(log(1+conndata2[,'orig_pkts']),
                log(1+conndata2[,'orig_ip_bytes'])))
kable(conncor1)
```

- Which gives:

Example of correlation

- R code:

```
conncor1=c(linear=cor(conndata2[,'orig_pkts'],
                      conndata2[,'orig_ip_bytes']),
    log=cor(log(1+conndata2[,'orig_pkts']),
                log(1+conndata2[,'orig_ip_bytes'])))
kable(conncor1)
```

- Which gives:

	Correlation
linear	0.9911887
log	0.9452585

- Linear-scale correlation is dominated by the large values, which makes it look better than it really is.

Example of data frame correlation

```r
## Extracting valid data
conndatasize=conndata2[,c('orig_pkts','resp_pkts',
                         'orig_bytes','resp_bytes')]
conndatasize=conndatasize[
    !apply(conndatasize,1,function(x)any(x=="-")),]
for(i in 1:dim(conndatasize)[2])
conndatasize[,i]=as.integer(conndatasize[,i])
```

Example of data frame correlation

```
## Extracting valid data
conndatasize=conndata2[,c('orig_pkts','resp_pkts',
                          'orig_bytes','resp_bytes')]
conndatasize=conndatasize[
    !apply(conndatasize,1,function(x)any(x=="-")),]
for(i in 1:dim(conndatasize)[2])
conndatasize[,i]=as.integer(conndatasize[,i])

## Computing the correlation matrix
cordatasize=cor(conndatasize)
```

Example of data frame correlation

	orig_pkts	resp_pkts	orig_bytes	resp_bytes
orig_pkts	1	0.999705713975153	0.00108611529297986	0.00264433365396342
resp_pkts	0.999705713975153	1	0.000945267947070292	0.00262111604209595
orig_bytes	0.00108611529297986	0.000945267947070292	1	0.0735429914375197
resp_bytes	0.00264433365396342	0.00262111604209595	0.0735429914375197	1

R Code for previous slide

```r
## Extracting valid data
library(DT)
library(RColorBrewer)
cuts=seq(0,1,length.out=101)[-1]
colors=colorRampPalette(brewer.pal(9,'Blues'))(101)
datatable(cordatasize) %>%
  formatStyle(columns = rownames(cordatasize),
  background = styleInterval(cuts,colors))
```

Regression is analogous to linear algebra with noise

- Most problems in Linear Algebra can be seen as **solving a system of linear equations:**

$$Ax + b = 0.$$

- However, data are not usually generated from a linear model.
- We therefore typically seek the least-bad fit that we can:

$$\min ||Ax + b||_2^2$$

  - i.e. we find A and b such that they minimise the distance (in the squared L_2 norm)
- Linear Algebra is therefore a very powerful way to view regression.

Matrix form of least squares

- Consider data X' with p' features (columns) and n observations.
- Given the regression problem:

$$\mathbf{y} = \mathrm{X}'\beta' + \mathbf{b} + \mathbf{e}$$

  - to find β' (a matrix dimension $p' \times 1$))
  - and b to minimise e
  - in $e^2 = \sum_{i=1}^{n} \epsilon_i^2$:

Matrix form of least squares

- We construct a simpler representation by adding a constant feature:

$$X = \begin{bmatrix} 1 & X_{11} & \cdots & X_{1p'} \\ & & \cdots & \\ 1 & X_{n1} & \cdots & X_{np'} \end{bmatrix}$$

  - which has $p = p' + 1$ features.
- We now solve the analogous equation:

$$\mathbf{y} = X\beta + \mathbf{e}$$

  - which has the same solution but is in a more convenient form.

Mean Squared Error (MSE)

- The prediction error is:

$$\mathbf{e}(\beta) = \mathbf{y} - \mathrm{X}\beta$$

- It can be shown that:

$$\mathrm{MSE}(\beta) = \frac{1}{n}\mathbf{e}^T\mathbf{e}$$

Minimising MSE

Taking (vector) derivatives with respect to β:

$$\nabla \mathrm{MSE}(\beta) = \frac{1}{n}(\nabla \mathbf{y}^T\mathbf{y} - 2\nabla\beta^T x^T\mathbf{y} + \nabla\beta^T \mathrm{X}^T\mathrm{X}\beta) \quad (1)$$

$$= \frac{1}{n}(0 - 2x^T\mathbf{y} + 2\mathrm{X}^T\mathrm{X}\beta) \quad (2)$$

which is zero at the optimum $\hat{\beta}$:

$$\mathrm{X}^T\mathrm{X}\hat{\beta} - \mathrm{X}^T\mathbf{y} = 0$$

with the solution:

$$\hat{\beta} = (\mathrm{X}^T\mathrm{X})^{-1}\mathrm{X}^T\mathbf{y}.$$

Exercise: For the case $p' = 1$, check that this solution is the same as you can find in regular linear algebra textbooks.

The Hat Matrix

There is an important and **data independent** quantity hidden in the prediction:

$$\mathrm{H} = \mathrm{X}(\mathrm{X}^T\mathrm{X})^{-1}\mathrm{X}^T$$

The fitted values are:

$$\hat{y} = \mathrm{X}\hat{\beta} = \mathrm{X}(\mathrm{X}^T\mathrm{X})^{-1}\mathrm{X}^T\mathbf{y} = \mathrm{H}\mathbf{y}$$

- H is dimension $N \times N$
- H "projects" y into the fitted value space $\hat{y}$

Properties of the Hat Matrix

- **Influence**: $\frac{\partial \hat{y}_i}{\partial y_j} = \mathrm{H}_{ij}$. So H controls how much a change in one observation changes the estimates of each other point.
- **symmetry**: $\mathrm{H}^T = \mathrm{H}$. So influence is symmetric.
- **Idempotency**: $\mathrm{H}^2 = \mathrm{H}$. So the predicted value for any projected point is the predicted value itself.

You should read up on these and other vector algebra properties.

Residuals and the Hat Matrix

The residuals can be written:

$$e = y - \mathrm{H}y = (\mathrm{I} - \mathrm{H})\mathbf{y}$$

$\mathrm{I} - \mathrm{H}$ is also symmetric and idempotent, and can also be interpreted in terms of Influence. Because of this,

$$\mathrm{MSE}(\hat{\beta}) = \frac{1}{n}\mathbf{y}^T(1 - \mathrm{H})^T(1 - \mathrm{H})\mathbf{y} = \frac{1}{n}\mathbf{y}^T(1 - \mathrm{H})\mathbf{y}$$

Expectations

If the data were generated by our model(**!**) then they are described by a random variable $\mathbf{Y}$:

$$\mathbf{Y} = \mathbf{x}\beta + \epsilon$$

where ϵ is an $n \times 1$ matrix of RVs with mean $\mathbf{0}$ and covariance $\sigma^s \mathrm{I}$. From this it is straightforward to show that the **fitted values are unbiased**:

$$\mathbb{E}[\hat{y}] = \mathbb{E}[\mathrm{H}Y] = \mathbf{x}\beta$$

using the properties of Expectations with the symmetry and idempotency of H.

Covariance

Similarly, it is straightforward to show that

$$\mathrm{Var}[\hat{y}] = \sigma^2 \mathrm{H}$$

using the properties of Variances with the symmetry and idempotency of H.

Discrete predictors

If you include categorical/factor predictors, each **level** or unique value is used as a binary predictor.
Nothing clever is done by default!

A new dataset for average packet size

```
conndatasize2=conndata2[,c('orig_pkts','resp_pkts','orig_bytes',
                           'resp_bytes','service')]
conndatasize2=conndatasize2[!apply(conndatasize2,1,
                                   function(x)any(x=="-")),]
for(i in 1:4)
    conndatasize2[,i]=as.integer(conndatasize2[,i])
conndatasize2$orig_avg_size=
        conndatasize2$orig_bytes/conndatasize2$orig_pkts
conndatasize2$resp_avg_size=
        conndatasize2$resp_bytes/conndatasize2$resp_pkts
conndatasize2[,'service']=as.factor(conndatasize2[,'service'])
for(i in 1:4) # log-transform raw data
    conndatasize2[,i]=log(1+conndatasize2[,i])
```

Linear Modelling in R: average packet size

Try to predict packet size

```
## Correlate numerical variables
cor(conndatasize2[,c(1:4,6:7)])
# Can't use variables that are too correlated!
summary(lm(orig_avg_size~resp_avg_size+orig_bytes,
           data=conndatasize2))
summary(lm(orig_avg_size~resp_avg_size+orig_pkts+
               orig_bytes,data=conndatasize2))
summary(lm(orig_avg_size~resp_avg_size+orig_pkts+
               orig_bytes+service,data=conndatasize2))
```

Linear Modelling in R: average packet size

```
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)    -1.119e+02  2.371e+01  -4.719 2.44e-06 ***
resp_avg_size  -4.301e-08  2.233e-07  -0.193   0.8472
orig_pkts      -6.259e+01  1.843e+00 -33.956  < 2e-16 ***
orig_bytes      6.985e+01  1.193e+00  58.544  < 2e-16 ***
serviceftp      1.059e+01  2.504e+01   0.423   0.6723
serviceftp-data 2.120e+02  2.748e+01   7.715 1.49e-14 ***
servicehttp    -1.111e+02  2.379e+01  -4.669 3.12e-06 ***
servicesmtp     8.929e+01  3.735e+01   2.390   0.0169 *
servicessh     -5.968e+01  2.438e+01  -2.448   0.0144 *
servicessl     -1.189e+02  2.387e+01  -4.982 6.52e-07 ***
```

Linear Modelling in R: average packet size

Conclusions:

- Can't predict received packet size from these data

For sent packets:

- Packet size is larger if you send fewer packets, or more data
- HTTP, SSH and SSL all send smaller packets than DNS, FTP, SMTP

Important caveats:

- **this is all excluding any record containing missing data**
- Compare to raw (untransformed) results!

Reflection

By the end of the course, you should: - Be able to define **correlation** and **regression** in multivariate context - Be able to perform basic calculations using these concepts - Be able to extend intuition about their application.

Signposting

Now we know about a class of important models, we have something to look into with:

- **Statistical Testing**,
- **Resampling** methods, and
- **Model Selection**

References

There is a lot more technical detail in Cosma Shalizi's course on **Modern Regression**:
Modern Regression, by Coasma Shalizi
http://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/