Week 4 - Descriptive Statistics in Mathematical Modelling

Basic Statistics

When it comes to calculating descriptive statistics, R can basically do it all. Let's start with functions that are included in the base installation. We will then look for extensions that are available through the use of user-contributed packages. For illustrative purposes, we will again use several of the variables from the Motor Trend Car Road Tests (mtcars) dataset provided in the base installation. We will focus on miles per gallon (mpg), horsepower (hp), and weight (wt):

```
myvars <- c("mpg", "hp", "wt")
head(mtcars[myvars])
```

```
                   mpg  hp    wt
Mazda RX4         21.0 110 2.620
Mazda RX4 Wag     21.0 110 2.875
Datsun 710        22.8  93 2.320
Hornet 4 Drive    21.4 110 3.215
Hornet Sportabout 18.7 175 3.440
Valiant           18.1 105 3.460
```

Let's first look at descriptive statistics for all 32 models. We will then examine descriptive statistics by transmission type (am) and number of cylinders (cyl). Transmission type is a dichotomous variable coded 0=automatic, 1=manual, while the number of cylinders can be 4, 5, or 6. In the base installation, you can use the summary() function to obtain descriptive statistics. An example is presented in the following listing.

```
myvars <- c("mpg", "hp", "wt")
summary(mtcars[myvars])
```

```
      mpg              hp              wt
 Min.   :10.40   Min.   : 52.0   Min.   :1.513
 1st Qu.:15.43   1st Qu.: 96.5   1st Qu.:2.581
 Median :19.20   Median :123.0   Median :3.325
 Mean   :20.09   Mean   :146.7   Mean   :3.217
 3rd Qu.:22.80   3rd Qu.:180.0   3rd Qu.:3.610
 Max.   :33.90   Max.   :335.0   Max.   :5.424
```

The summary() function provides the minimum, maximum, quartiles, and mean for numerical variables, and the respective frequencies for factors and logical vectors. The functions apply() or sapply() can be used to provide any descriptive statistics. The format in use is:

```
sapply(x, FUN, options)
```

where x is the data frame (or matrix) and FUN is an arbitrary function. If options are present, they're passed to FUN. Typical functions that can be plugged here are

```
mean()
sd()
var()
min()
max()
median()
length()
range()
quantile()
fivenum()
```

The example in the next listing provides several descriptive statistics using sapply(), including skew and kurtosis.

```
mystats <- function(x, na.omit=FALSE){
if (na.omit)
    x <- x[!is.na(x)]
    m <- mean(x)
    n <- length(x)
    s <- sd(x)
    skew <- sum((x-m)^3/s^3)/n
    kurt <- sum((x-m)^4/s^4)/n - 3
return(c(n=n, mean=m,
stdev=s, skew=skew,
```

```
kurtosis=kurt))
}

myvars <- c("mpg", "hp", "wt")
sapply(mtcars[myvars], mystats)
```

```
                mpg          hp          wt
n         32.000000  32.0000000 32.00000000
mean      20.090625 146.6875000  3.21725000
stdev      6.026948  68.5628685  0.97845744
skew       0.610655   0.7260237  0.42314646
kurtosis  -0.372766  -0.1355511 -0.02271075
```

For cars in this sample, the mean mpg is 20.1, with a standard deviation of 6.0. The distribution is skewed to the right (+0.61) and is somewhat flatter than a normal distri- bution (–0.37). This is most evident if you graph the data. Note that if you wanted to omit missing values, you could use

```
sapply(mtcars[myvars], mystats,
na.omit=TRUE)
```

```
                mpg          hp          wt
n         32.000000  32.0000000 32.00000000
mean      20.090625 146.6875000  3.21725000
stdev      6.026948  68.5628685  0.97845744
skew       0.610655   0.7260237  0.42314646
kurtosis  -0.372766  -0.1355511 -0.02271075
```

The Hmisc and pastecs packages

Several packages offer functions for descriptive statistics, including Hmisc and pastecs. Because these packages are not included in the base distribution, they need to be installed on first use. Hmisc's describe() function returns the number of variables and observations, the number of missing and unique values, the mean, quantiles, and the five highest and lowest values. An example is provided in Table 3. The pastecs package includes the function stat.desc() that provides a wide range of descriptive statistics. The format is

```
stat.desc(x, basic=TRUE, desc=TRUE,
norm=FALSE, p=0.95)
```

where x is a data frame or a time series. If basic=TRUE (the default), the number of values, null values, missing values, minimum, maximum, range, and sum are provided. If desc=TRUE (also the default), the median, mean, standard error of the mean, 95% confidence interval for the mean, variance, standard deviation, and coefficient of variation are also provided.

Finally, if norm=TRUE (not the default), normal distribution statistics are returned, including skewness and kurtosis (with statistical significance) and the Shapiro–Wilk test of normality. A p−value option is used to calculate the confidence interval for the mean (.95 by default). The next listing gives an example.

```
library(pastecs)
myvars <- c("mpg", "hp", "wt")
stat.desc(mtcars[myvars])
```

```
                    mpg           hp          wt
nbr.val      32.0000000   32.0000000  32.0000000
nbr.null      0.0000000    0.0000000   0.0000000
nbr.na        0.0000000    0.0000000   0.0000000
min          10.4000000   52.0000000   1.5130000
max          33.9000000  335.0000000   5.4240000
range        23.5000000  283.0000000   3.9110000
sum         642.9000000 4694.0000000 102.9520000
median       19.2000000  123.0000000   3.3250000
mean         20.0906250  146.6875000   3.2172500
SE.mean       1.0654240   12.1203173   0.1729685
CI.mean.0.95  2.1729465   24.7195501   0.3527715
var          36.3241028 4700.8669355   0.9573790
std.dev       6.0269481   68.5628685   0.9784574
coef.var      0.2999881    0.4674077   0.3041285
```

Correlations

Correlation coefficients are used to describe relationships among quantitative variables. The sign ± indicates the direction of the relationship (positive or inverse), and the magnitude indicates the strength of the relationship (ranging from 0 for no linear relationship to 1 for a perfectly predictable linear relationship).

In this section, we look at a variety of correlation coefficients, as well as tests of significance.

We will use the state.x77 dataset available in the base R installation. It provides data on the population, income, illiteracy rate, life expectancy, murder rate, and high school graduation rate for the 50 US states in 1977. There are also temper- ature and land-area measures, but we drop them to save space. In addition to the base installation, we'll be using the psych and ggm

packages. R can produce a variety of correlation coefficients, including Pearson, Spearman, Kendall, partial, polychoric, and polyserial. The Pearson product-moment correlation assesses the degree of linear relationship between two quantitative variables. Spearman's rank-order correlation coefficient assesses the degree of relationship between two rank-ordered variables. Kendall's tau is also a nonparametric measure of rank correlation.

The cor() function produces all three correlation coefficients, whereas the cov() function provides covariances. There are many options, but a simplified format for producing correlations is

```
cor(x, use= , method= )
```

Where x is a matrix or a data frame, and use specifies the handling of missing data. The options are all.obs (assumes no missing data), everything (any correlation involving a case with missing values will be set to missing), complete.obs (listwise deletion), and pairwise. complete.obs (pairwise deletion). The method spec- ifies the type of correlation. The options are pearson, spearman, and kendall. The default options are use ="everything" and method= "pearson". An example is provided in Table 4 The first call produces the variances and covariances. The second provides Pearson product-moment correlation coefficients, and the third produces Spearman rank-order correlation coefficients. You can see, for example, that a strong positive correlation exists between income and high school graduation rate and that a strong negative correlation exists between illiteracy rates and life expectancy. A partial correlation is a correlation between two quantitative variables, controlling for one or more other quantitative variables. You can use the pcor() function in the ggm package to provide partial correlation coefficients. Again, this package is not installed by default, so be sure to install it on first use. The format is

```
pcor(u, S)
```

where u is a vector of numbers, with the first two numbers being the indices of the variables to be correlated, and the remaining numbers being the indices of the conditioning variables (that is, the variables being partialed out), and S is the covariance matrix among the variables. An example will help clarify this:

```
library(ggm)
colnames(states)
pcor(c(1,5,2,3,6), cov(states))
```

In this case, 0.346 is the correlation between population (variable 1) and murder rate (variable 5), controlling for the influence of income, illiteracy rate, and high school graduation rate (variables 2, 3, and 6 respectively). The use of partial correlations is common in the social sciences.