Introduction to Biostatistics (171:161)
Breheny

Lab #4

In lab # 3, we explored the Titanic data set and in the process, hopefully learned a lot about how to describe, graph, and explore categorical data. In this lab and the next, we will explore a data set that has both continuous and categorical variables, learning tools along the way for describing, graphing, and exploring the distribution of continuous variables as well as relationships between two continuous variables, and between continuous and categorical variables.

Our data set that we will use comes from the efforts of a waiter who recorded information about 244 tips he received over a period of a few months working in a restaurant (`tips.txt`). He recorded several variables:

- `TIP`, measured in dollars
- `TOTBILL`, the total cost of the meal, measured in dollars
- `SEX` of the bill payer
- `SMOKER`, whether the party included smokers (*i.e.*, was in the smoking section)
- `DAY` of the week
- `TIME`, whether the meal occurred during the day or night shift
- `SIZE` of the dining party

The important continuous variables in this data set are `TOTBILL` and `TIP`.

1 Summary statistics

In SAS, summary statistics for continuous data can be obtained using `PROC UNIVARIATE`. You can obtain numerical summaries for the variables you are interested in by using the `VAR` statement. In `R`, there are functions `mean`, `median`, `sd`, and so on. For example:

SAS:

```
PROC UNIVARIATE DATA = Tips;
    VAR TotBill tip;
RUN;
```

`R`:

```
attach(tips)
mean(Tip)
mean(TotBill)
median(TotBill)
sd(Tip)
quantile(Tip, c(0, .25, .5, .75, 1))
```

This provides us with all of the summary statistics we talked about in lecture: the mean, standard deviation, median, quantiles, minimum, and maximum, along with a bunch of other information.

These summary statistics are for the entire data set. To obtain group-specific summaries, we can include a `WHERE` statement in SAS or brackets in `R`, as in:

SAS:

```
PROC UNIVARIATE DATA = Tips;
  WHERE time = "Night";
  VAR TotBill;
RUN;
```

`R`:

```
mean(TotBill[Time == "Night"])
sd(TotBill[Time == "Night"])
mean(TotBill[Time == "Day"])
sd(TotBill[Time == "Day"])
```

In `R`, note the double equal sign (`A == 5`); this tests whether `A` is equal to 5, as opposed to the single equal sign (`A = 5`), which sets `A` equal to 5. We can do the same for day; note that the nighttime meals have a higher average bill as well as a larger standard deviation.

2 Histograms

What does this look like when we plot it? Let's make histograms first. Histograms are created with the logically-named `HISTOGRAM` statement/function:

SAS:

```
PROC SGPLOT DATA = Tips;
  HISTOGRAM TotBill;
RUN;
```

`R`:

```
require(lattice)
histogram(TotBill)
```

We can see that most bills were around $15, but that some were as high as $50. As we did last week, we can break this plot down by conditioning:

SAS:

```
PROC SGPANEL DATA = Tips;
  PANELBY time;
  HISTOGRAM TotBill;
RUN;
```

`R`:

```
histogram(~ TotBill | Time)
```

This is perhaps not the best layout for comparing which group had the higher mean. Let's put all the histograms in one column by specifying the `COLUMNS` option (`layout` in `R`):

SAS:

```
PROC SGPANEL DATA = Tips;
  PANELBY time / COLUMNS = 1;
  HISTOGRAM TotBill;
RUN;
```

`R`:

```
histogram(~ TotBill | Time, layout = c(1,2))
```

This makes it perhaps a little easier to see the difference between the two: a rather high percent of lunches tend to be between about $10 and $20, whereas dinners are more spread out through the $20-$30 range as well. Note that this observation agrees with the mean and SD that we got earlier.

3 Box plots

Box plots are pretty straightforward:

SAS:

```
PROC SGPLOT DATA = Tips;
  VBOX TotBill / CATEGORY = Time;
RUN;
```

R:

```
boxplot(TotBill ~ Time,
        ylab = "Total bill")
points(1, mean(TotBill[Time == "Day"]),
       pch = 5)
points(2, mean(TotBill[Time == "Night"]),
       pch = 5)
```

Once again, dinner bills are a little higher and more spread out than lunch bills. Note that SAS draws a little diamond on its box plots to represent the mean of the data. The same can be accomplished in R by using the points function.

4 Scatter plots

Scatter plots are made using a SCATTER statement. The relationship we are particularly interested in is the connection between total bill and tip:

SAS:

```
PROC SGPLOT DATA = Tips;
  SCATTER X = TotBill Y = Tip;
RUN;
```

R:

```
plot(TotBill, Tip) ## Or:
xyplot(Tip ~ TotBill)
```

The plot illustrates several trends:

- As we would expect, there is a positive association between bill and tip
- There is plenty of variation, however (big tips on small bills, small tips on big bills)
- There are more points in the lower right of the plot than the upper left – cheap tippers are more common than generous tippers?
- There seem to be some horizontal "stripes" in the plot – why?

We can create the same plot, but add the regression line by replacing SCATTER with REG in SAS, or using type in R to declare that we want both points ("p") and the regression line ("r"):

SAS:

```
PROC SGPLOT DATA = Tips;
  REG X = TotBill Y = Tip;
RUN;
```

R:

```
xyplot(Tip ~ TotBill, type = c("p", "r"))
```

This is the line that minimizes the residual sum of squares for predicting tip from total bill. Note that we get a different line if we change x and y:

SAS:

```
PROC SGPLOT DATA = Tips;
  REG Y = TotBill X = Tip;
RUN;
```

R:

```
xyplot(TotBill ~ Tip, type =c ("p", "r"))
```

Like it did before, the line goes up, but not through the same points. For example, the point $40 bill, $5 tip was on the first line, but not the second.

Finally, recall that conditioning helps us see how the relationship between bill and tip differs for different subcategories of dining parties. For example, let's compare smokers and nonsmokers:

SAS:

```
PROC SGPANEL DATA = Tips;
  PANELBY Smoker;
  SCATTER X = TotBill Y = Tip;
RUN;
```

R:

```
xyplot(Tip ~ TotBill | Smoker)
```

The correlation between tip and bill seems to be much stronger in the nonsmoking section than in the smoking section.

5 Correlation

Calculating correlations is pretty straightforward using `PROC CORR` in SAS, or the `cor` function in R:

SAS:

```
PROC CORR DATA = Tips;
RUN;
```

R:

```
cor(Tip, TotBill)
```

Which tells you that the overall correlation between total bill and tip is .68. Recall, however, that this correlation seemed to be different between smokers and nonsmokers... how can you calculate separate correlation coefficients? Note that `PROC CORR` also tells you the means and standard deviations of the two variables. Thus, you could calculate a regression line from this output; however, we'll see in the next section how to get SAS to do that directly.

NOTE: By default, `PROC CORR` will calculate the correlation between all of the numeric variables in a data set. So here, for example, SAS also tells you the correlation between party size and total bill; it's 0.6, which makes sense. However, if you have a lot of continuous variables in your data set, this output will get overwhelming. In these situations, you can use a `VAR` statement to restrict attention to a smaller group of variables. For example, try inserting `VAR TotBill Tip;` into the above code.

This concludes lab #4.