Relationships Cheat Sheet

Nick Huntington-Klein

April 5, 2019

Definitions

- **Dependent**: `X` and `Y` are *dependent* if knowing something about `X` gives you information about what `Y` is likely to be, or vice versa
- **Correlated**: `X` and `Y` are *correlated* if knowing that `X` is unusually high tells you whether `Y` is likely to be unusually high or unusually low
- **Explaining**: *Explaining* `Y` using `X` means that we are predicting what `Y` is likely to be, given a value of `X`

Tables

`table(x)` will show us the full *distribution* of `x`. `table(x,y)` will show us the full *distribution* of `x` and `y` together (the "joint distribution"). Typically not used for continuous variables.

```
library(Ecdat)
data(Benefits)

table(Benefits$joblost)
```

```
##
##              other position_abolished seasonal_job_ended
##               1976                402                177
##         slack_work
##               2322
```

```
table(Benefits$joblost,Benefits$married)
```

```
##
##                        no  yes
##   other               709 1267
##   position_abolished  119  283
##   seasonal_job_ended   72  105
##   slack_work          891 1431
```

You can label the variable names using the confusingly-named *dimnames names* option, `dnn`

```
table(Benefits$joblost,Benefits$married,dnn=c('Job Loss Reason','Married'))
```

```
##                     Married
## Job Loss Reason        no  yes
##   other               709 1267
##   position_abolished  119  283
##   seasonal_job_ended   72  105
##   slack_work          891 1431
```

Wrap `table()` in `prop.table()` to get proportions instead of counts. The `margin` option of `prop.table()` will give the proportion within each row (`margin=1`) or within each column (`margin=2`) instead of overall.

```r
prop.table(table(Benefits$joblost,Benefits$married))
```

```
##
##                              no        yes
##   other              0.14537626 0.25979086
##   position_abolished 0.02440025 0.05802748
##   seasonal_job_ended 0.01476317 0.02152963
##   slack_work         0.18269428 0.29341808
```

```r
prop.table(table(Benefits$joblost,Benefits$married),margin=1)
```

```
##
##                             no       yes
##   other              0.3588057 0.6411943
##   position_abolished 0.2960199 0.7039801
##   seasonal_job_ended 0.4067797 0.5932203
##   slack_work         0.3837209 0.6162791
```

```r
prop.table(table(Benefits$joblost,Benefits$married),margin=2)
```

```
##
##                              no        yes
##   other              0.39586823 0.41056384
##   position_abolished 0.06644333 0.09170447
##   seasonal_job_ended 0.04020101 0.03402463
##   slack_work         0.49748744 0.46370706
```

Correlation

We can calculate the correlation between two (numeric) variables using `cor(x,y)`

```r
cor(Benefits$age,Benefits$tenure)
```

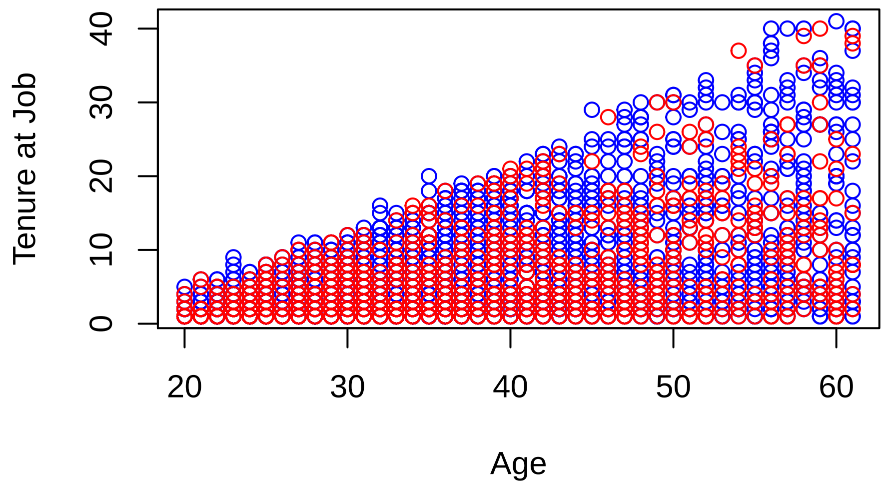
```
## [1] 0.4864526
```

Scatterplots

You can plot one variable against another with `plot(xvar,yvar)`. Add `xlab`, `ylab`, and `main` options to title the axes and entire plot, respectively. Use `col` to assign a color.

Use `points()` to add more points to a graph after you've made it, likely with a different `color`.

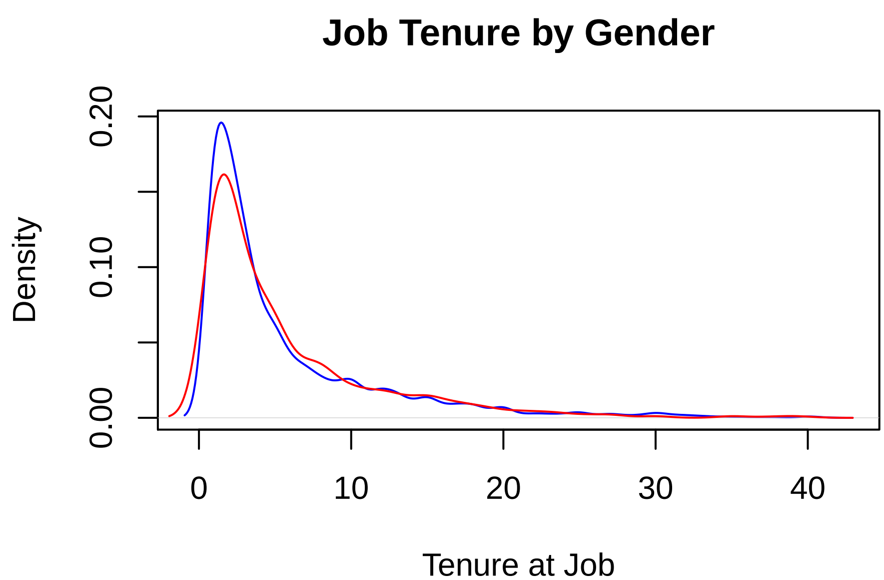
```r
library(tidyverse)
BenefitsM <- Benefits %>% filter(sex=='male')
BenefitsF <- Benefits %>% filter(sex=='female')
plot(BenefitsM$age,BenefitsM$tenure,xlab='Age',ylab='Tenure at Job',col='blue')
points(BenefitsF$age,BenefitsF$tenure,xlab='Age',ylab='Tenure at Job',col='red')
```



Overlaid Densities

You can show how the *distribution* of `Y` changes for different values of `X` by plotting the density separately for different values of `X`. Use `lines` to add the second density plot after you've done the first one.

```
plot(density(BenefitsM$tenure),
     xlab='Tenure at Job',col='blue',main="Job Tenure by Gender")
lines(density(BenefitsF$tenure),xlab='Tenure at Job',col='red')
```



Means Within Groups and Explaining

Part of looking at both *correlation* and *explanation* will require getting the mean of `Y` within values of `X`, which we can do with `group_by()` in `dplyr/tidyverse`.

Using `summarize()` after `group_by()` will give us a table of means within each group. Using `mutate()` will add a new variable assigning that mean. Use `mutate()` with `mean(y)` to get the part of `y` explained by `x`, or with `y - mean(y)` to get the part not explained by `x` (the residual). Don't forget to `ungroup()`!

```
Benefits %>% group_by(joblost) %>%
  summarize(tenure = mean(tenure), age = mean(age))
```

```
## # A tibble: 4 x 3
##   joblost            tenure   age
##   <fct>               <dbl> <dbl>
## 1 other                7.12  37.3
## 2 position_abolished   6.28  38.8
## 3 seasonal_job_ended   3.53  32.8
## 4 slack_work           4.48  34.9
```

```
Benefits <- Benefits %>% group_by(joblost) %>%
  mutate(tenure.exp = mean(tenure),
         tenure.resid = tenure - mean(tenure)) %>% ungroup()
head(Benefits %>% select(joblost,tenure,tenure.exp,tenure.resid))
```

```
## # A tibble: 6 x 4
##   joblost    tenure tenure.exp tenure.resid
##   <fct>       <int>      <dbl>        <dbl>
## 1 other          21       7.12        13.9
## 2 slack_work      2       4.48        -2.48
## 3 other          19       7.12        11.9
## 4 slack_work     17       4.48        12.5
## 5 slack_work      1       4.48        -3.48
## 6 other           3       7.12        -4.12
```

Explaining With a Continuous Variable

If we want to explain `Y` using `X` but `X` is continuous, we need to break it up into bins first. We will do this with `cut()`, which has the `breaks` option for how many bins to split it up into.

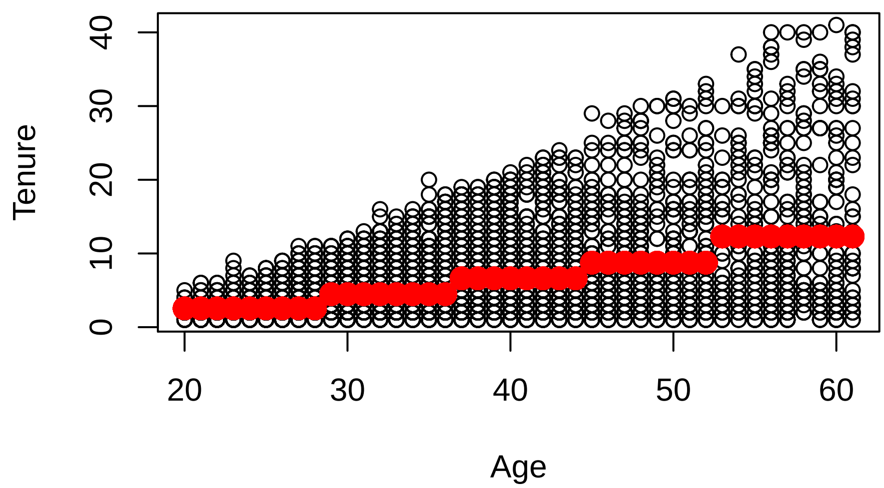
In this class, we will be choosing the number of breaks arbitrarily. I'll tell you what values to use.

```
Benefits <- Benefits %>% mutate(agebins = cut(age,breaks=5)) %>%
  group_by(agebins) %>%
  mutate(tenure.ageexp = mean(tenure),
         tenure.ageresid = tenure - mean(tenure)) %>% ungroup()
head(Benefits %>% select(agebins,tenure,tenure.ageexp,tenure.ageresid))
```

```
## # A tibble: 6 x 4
##   agebins     tenure tenure.ageexp tenure.ageresid
##   <fct>        <int>         <dbl>           <dbl>
## 1 (44.6,52.8]     21          8.75          12.2
## 2 (20,28.2]        2          2.55          -0.551
## 3 (36.4,44.6]     19          6.62          12.4
## 4 (44.6,52.8]     17          8.75           8.25
## 5 (28.2,36.4]      1          4.48          -3.48
```

```
## 6 (44.6,52.8]      3          8.75          -5.75
```

```r
plot(Benefits$age,Benefits$tenure,xlab="Age",ylab="Tenure",col='black')
points(Benefits$age,Benefits$tenure.ageexp,col='red',cex=1.5,bg='red',pch=21)
```



Proportion of Variance Explained

When `Y` is numeric, we can calculate its variance, and see how much of that variance is explained by `X`, and also how much is not. We do this by calculating the variance of the residuals, as this is the amount of variance in `Y` left over after taking out what `X` explains.

```r
#Proportion of tenure NOT explained by age
var(Benefits$tenure.ageresid)/var(Benefits$tenure)
```

```
## [1] 0.7725183
```

```r
#Proportion of tenure explained by age
1 - var(Benefits$tenure.ageresid)/var(Benefits$tenure)
```

```
## [1] 0.2274817
```

```r
var(Benefits$tenure.ageexp)/var(Benefits$tenure)
```

```
## [1] 0.2274817
```