ACSC/STAT 3740, Predictive Analytics

WINTER 2024

Toby Kenney

Homework Sheet 1

Model Solutions

[Note: all data in this homework are simulated.]

[Note: With many of these problems, there is no "correct" solution. These model solutions give a range of reasonable approaches, but there are many other good approaches that could be taken.]

Common `ggplot` settings

These settings are used for a number of plots in this homework. For clarity, we define them as variables at the start of our script.

```
library(dplyr)
### For various commands to tidy up the data mostly using the %>% operator.

library(ggplot2)
### For various plotting commands.

library(scales)
### For various commands that adjust scales, such as trans_new, which
### creates a new axis transformation.


### This defines a theme that can be added to any ggplot to make the text
### larger. It is used in many of the solutions for this homework.

largertextsize<-theme(plot.title=element_text(size=20,hjust=0.5),
                      axis.title=element_text(size=20,hjust=0.5),
                      axis.text=element_text(size=16),
                      legend.title=element_text(size=20,hjust=0.5),
                      legend.text=element_text(size=16),
                      strip.text=element_text(size=16))

### Not all of these are used for every plot.
### Most of the names are obvious. strip.text is for titles of
### subplots made with facet_wrap or facet_grid.
```

Basic Questions

1. *The file* `HW1Q1.txt` *contains data from a university about student research projects. The data are not formatted in a very convenient way. Read the data into* `R` *and reformat into a more convenient way, and use it to create a plot showing funding (y-axis) vs GPA (x-axis) with colour showing student age and size showing professor age, with a facet grid of professor subject versus student subject. Make a list of all corrections made to the data.*

   We first read and clean the profs table:

```
profs<-read.table("HW1Q1.txt",skip=2,nrow=157)
### could use stringsAsFactors, but will need to fix factors manually
### anyway to correct mistakes.

profs$ID<-as.integer(profs$ID)
str(profs)
### Try to set the levels of the gender variable.
prof.gender<-factor(profs$gender,levels=c("female","male"))
### Check for any unexpected values
profs[is.na(prof.gender),]
profs$gender[34]<-"female"
profs$gender[60]<-"male"
prof.gender<-factor(profs$gender,levels=c("female","male"))
which(is.na(prof.gender))

### Now check subject
table(profs$subject)
profs$subject[profs$subject=="Chem"]<-"Chemistry"
profs$subject[profs$subject=="Physcs"]<-"Physics"
profs$subject<-factor(profs$subject,levels=c("Biology","Chemistry","Mathematics",
                                             "Physics","Statistics"))
which(is.na(profs$subject))
```

   We identify and fix several misspellings and abbreviations. We then do the same for the students table.

```
students<-read.table("HW1Q1.txt",skip=163,nrow=609)
str(students)
### We note that GPA is character, when it should be numeric.
### This indicates there are probably some errors that need to be fixed.

table(students$gender)
### We spot two values that are incorrect.

students$gender[students$gender=="F"]<-"female"
students$gender[students$gender=="mael"]<-"male"
students$gender<-factor(students$gender,levels=c("female","male"))
which(is.na(students$gender))
### All convert correctly.

table(students$subject)
### several spelling erros and abbreviations
students$subject[students$subject=="Bilogy"]<-"Biology"
students$subject[students$subject=="Chemistr"]<-"Chemistry"
students$subject[students$subject=="Stats"]<-"Statistics"
students$subject<-factor(students$subject,levels=c("Biology","Chemistry","Mathematics","P


stud.GPA<-as.numeric(students$GPA)
### likely to be some problems, so create a new variable to preserve original.
students$GPA[is.na(stud.GPA)]
students$GPA[is.na(stud.GPA)]<-3.73
stud.GPA<-as.numeric(students$GPA)
which(is.na(stud.GPA))
### Now it works.
students$GPA<-stud.GPA

summary(students)
### Some GPA values above 4.3 - the maximum possible
students$GPA[students$GPA>4.3] # probably decimal point in wrong place.
students$GPA[students$GPA>4.3]<-3.93
```

We might not check the range of GPA values at this stage, preferring to fix them when we examine graphs of the data. Next we read and clean the funding table.

```
funding<-read.table("HW1Q1.txt",skip=776)
dim(funding)
typeof(as.matrix(funding)) # should be "double"

### try converting to numeric and see where NAs appear.

### We could write a loop for each column, but simpler to use the fact
### that matrices are just vectors with a dimension attribute.
funding.value<-as.numeric(as.matrix(funding))
dim(funding.value)<-dim(funding)

### Check which values created NA when coercing to numeric.
as.matrix(funding)[is.na(funding.value)&!is.na(as.matrix(funding))]

### "N/A" is clearly supposed to indicate NA, so we replace these
funding[funding=="N/A"]<-NA

### Other values are clearly meant to be numeric, but are formatted strangely.
funding[is.na(funding.value)&!is.na(as.matrix(funding))]<-c(11400,5900,14600,5800,8500,13

### Redo the conversion
funding.value<-as.numeric(as.matrix(funding))
which(is.na(funding.value)&!is.na(as.matrix(funding)))
### All numbers convert OK.

dim(funding.value)<-c(157,609)
funding<-data.frame(funding.value)
```

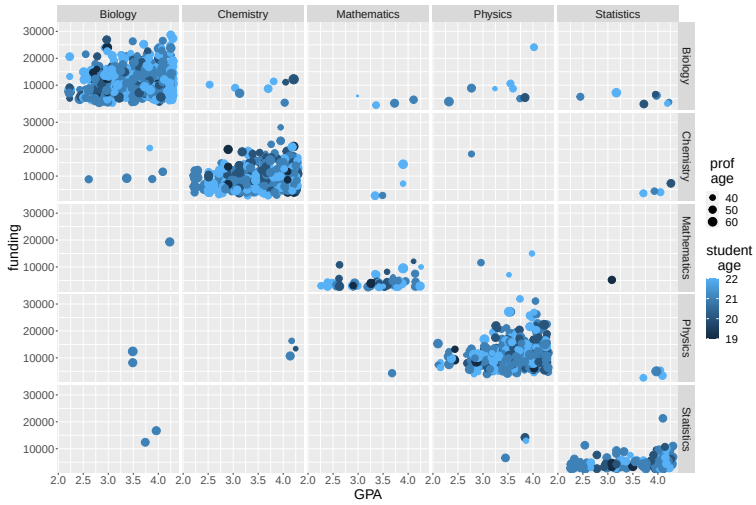
Now that each table is cleaned, we convert the funding table to long format and join the three tables.

```
### First need to include Prof_ID as a column in the funding data frame.
funding_pid<-data.frame(Prof_ID=row.names(funding),funding)

### Now we can pivot to long format.
funding_long<-tidyr::pivot_longer(funding_pid,cols=seq_len(609)+1,names_to="student",valu
funding_long
### Unfortunately, the original student IDs were integers, which were
### not valid column names, so when we processed the funding table,
### the IDs were prefixed by "X". We need to remove these, so that
### they can match with the student table.

library(dplyr)
funding_long$student<-as.integer(tidyr::separate(data.frame(funding_long$student),col=1,s

### Now we can use left_join to obtain the professor and student
### information in the funding table. We use left_join, so that we
### only include this information for funded projects.
funding_full<-funding_long%>%left_join(profs,by=c("Prof_ID"="ID"))%>%left_join(students,b
### ID is a primary key for both the profs and students data frames,
### so we don't need to worry about duplication.
```

Finally, we can make the plot requested.

```
library(ggplot2)
ggplot(data=funding_full,
       mapping=aes(x=GPA,
                   y=funding,
                   colour=age.y,
                   size=age.x))+
    geom_point()+
    facet_grid(subject.x~subject.y)+
    scale_colour_continuous(name="student\nage")+
    scale_size_continuous(name="prof\nage")+
    largertextsize
```

This gives the following plot.



During this process, we fixed the following errors in the data:

Professors

gender

- Professor 34 has gender "femle", which is almost certainly a misspelling of "female".
- Professor 60 has gender "M", which is clearly an abbreviation for "male".

subject

- Professor 91 has subject "Chem", which is clearly an abbreviation for "Chemistry".
- Professor 137 has subject "Physcs", which is clearly a misspelling of "Physics".

Students

gender

- Student 541 has gender "mael", which is almost certainly a misspelling of "male".
- Professor 507 has gender "F", which is clearly an abbreviation for "female".

subject

- Student 104 has subject "Stats", which is clearly an abbreviation for "Statistics".
- Student 351 has subject "Bilogy", which is clearly a misspelling for "Biology".
- Student 602 has subject "Chemistr", which is clearly a misspelling for "Chemistry".

GPA

- Student 321 has GPA "39.3", which should probably be "3.93".
- Student 323 has GPA "3,73", which should probably be "3.73".

Funding

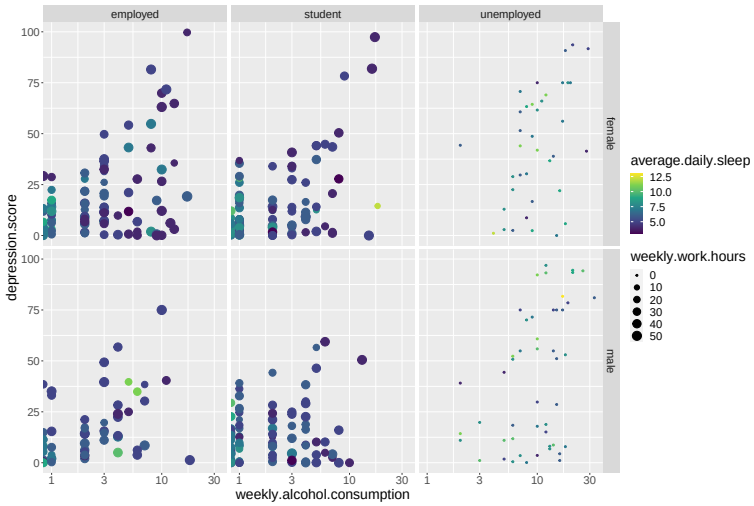
- For Professor 82, the funding values are prefixed by a $ sign.
- For Professor 149, one of the funding values is written as "10K", which should probably be "10000".
- Professor 156 has a number of values "N/A", which are clearly meant to be "NA".

2. *The file* `HW1Q1.txt` *is from an experiment about the effect of alcohol consumption on depression in young adults. It includes the following variables:*

Variable name	*Meaning*
Age	*The subject's age*
Gender	*The subject's gender*
Employment.Status	*The subject's employment status.*
Weekly.work.hours	*The average number of weekly hours spent working or studying.*
Average.daily.sleep	*The average number of hours spent asleep each day.*
Weekly.alcohol.consumption	*The subject's weekly alcohol consumption*
Depression.score	*An index indicating how many symptoms of depression the subject has.*

Display this data set in a plot.

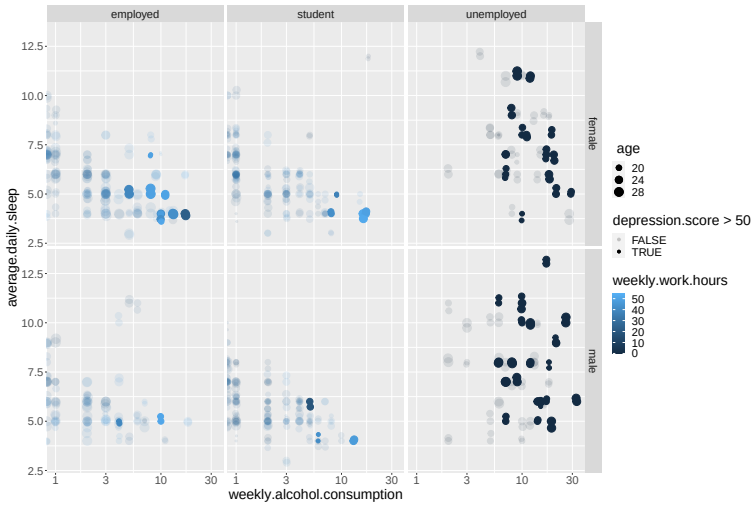
There are five continuous variables and two categorical variables. As the variable of interest, depression score should probably be on the y-axis, though using colour is an option. The variables most correlated with depression are weekly alcohol consumption and weekly work hours. These should probably use the x-axis and colour. If we subdivide by employment status, we see that average daily sleep is more correlated with depression, so we could use x-axis or colour for that instead of work hours. The factor variables do not have many levels, so might be effectively displayed using a facet grid. We find that weekly alcohol consumption is heavy-tailed, and might benefit from a log-transformation.



The code for this plot is

```
### scatterplot with depression score on the y-axis
ggplot(HW1Q2,
       mapping=aes(x=weekly.alcohol.consumption,
                   y=depression.score,
                   colour=average.daily.sleep,
                   size=weekly.work.hours))+
    geom_point()+
    facet_grid(gender~employment.status)+
    scale_x_log10()+
    scale_colour_viridis_c()+
    largertextsize
```

If we use alpha or colour as depression score, the plot can be difficult to read. One option to improve this is to set a cut-off, and only indicate whether the depression score is above the cut-off. This creates a very clear distinction between depressed and not-depressed individuals.

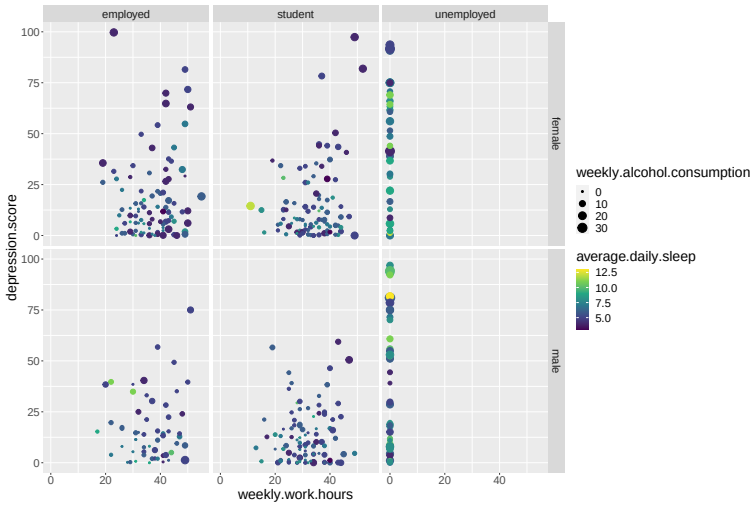


The code for this plot is

```
### use alpha to highlight points with depression>50
ggplot(HW1Q2, mapping=aes(x=weekly.alcohol.consumption,
                          alpha=depression.score>50,
                          y=average.daily.sleep,
                          size=age,
                          colour=weekly.work.hours))+
    geom_point()+
    facet_grid(gender~employment.status)+
    scale_x_log10()+
    geom_jitter()+ # a lot of overlapping values because of rounding.
    largertextsize
```

This plot uses size to show age, which is not very clear, but the relations with the other predictors are relatively clear.

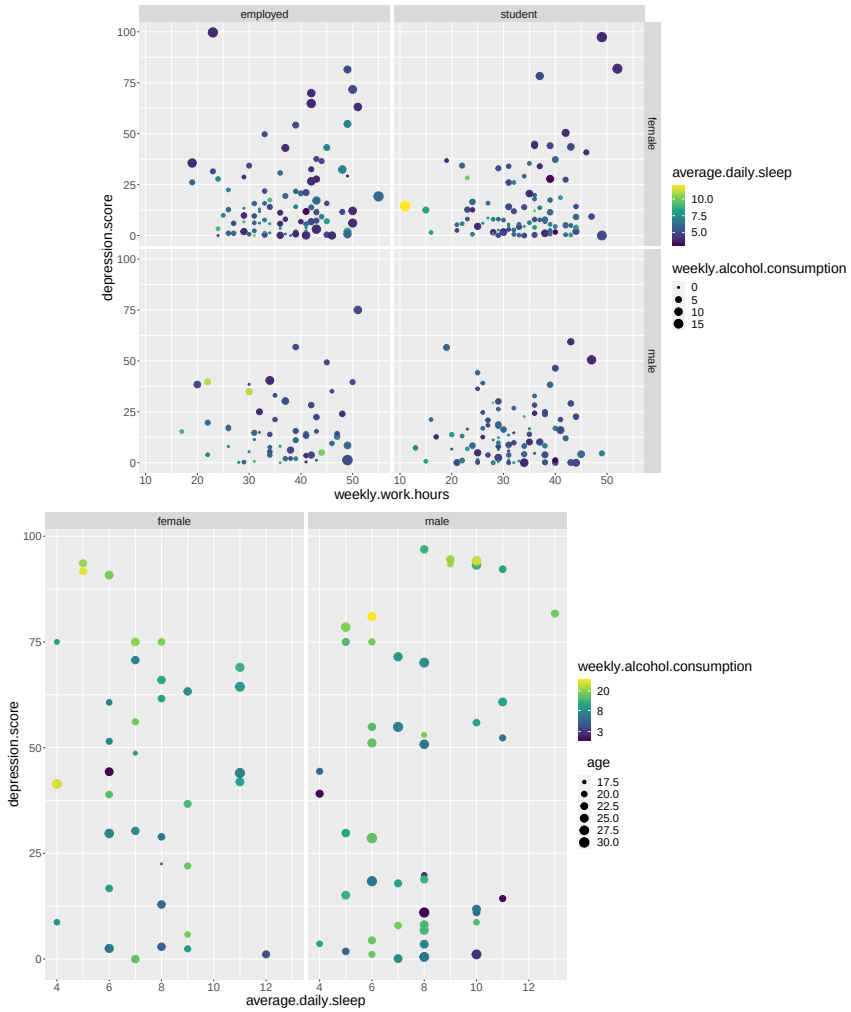
Plotting weekly work hours on the x-axis shows some interesting patterns. The relationship with alcohol consumption is actually still fairly clear if we use size for alcohol consumption.



The code for this plot is

```
### Try rearranging aesthetics
ggplot(HW1Q2, mapping=aes(x=weekly.work.hours,
                          y=depression.score,
                          colour=average.daily.sleep,
                          size=weekly.alcohol.consumption))+
    geom_point()+
    facet_grid(gender~employment.status)+
    scale_colour_viridis_c()+
    largertextsize
### could add scale_size_continuous(trans="log") but this highlights
### the high values more.
```

One problem with this plot is that weekly work hours do not contain useful information for unemployed individuals. One option here is to make separate plots for unemployed individuals, with different aesthetics.

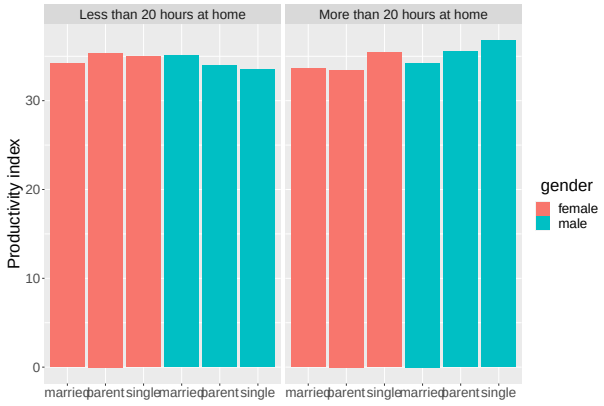


The code for these plots is

```r
### Restrict to employed and students.
ggplot(HW1Q2%>%filter(employment.status!="unemployed"),
       mapping=aes(x=weekly.work.hours,
                   y=depression.score,
                   colour=average.daily.sleep,
                   size=weekly.alcohol.consumption))+
    geom_point()+
    facet_grid(gender~employment.status)+
    scale_colour_viridis_c()+
    largertextsize

### Make a separate plot for unemployed
ggplot(HW1Q2%>%filter(employment.status=="unemployed"),
       mapping=aes(x=average.daily.sleep,
                   y=depression.score,
                   size=age,
                   colour=weekly.alcohol.consumption))+
    geom_point()+
    facet_wrap(gender~.)+
    scale_colour_viridis_c(trans="log",breaks=c(1,3,8,20))+
    largertextsize
```

The trouble with these separate plots for unemployed is that it makes comparison of depression scores for differnt employment categories more difficult.

3. *A company is studying the effect of working from home on productivity, and has produced the plot below. Identify at least three issues with the plot and produce a new plot that better displays the data.*



Provide `R` *code for the new plot. [The data used to produce the figure is in*

the file `HW1Q3.txt`*. You should include more information from that file in the plot as appropriate.]*

(i) The arbitrary cut-off of 20 hours from home does not account for hours worked — for some employees, 20 hours from home would mean working from home 80% of the time, while for others, it would mean working from home less than 50% of the time.

(ii) Observations are not divided by job type. Some job types are more productive and also have a higher proportion of employees working from home.

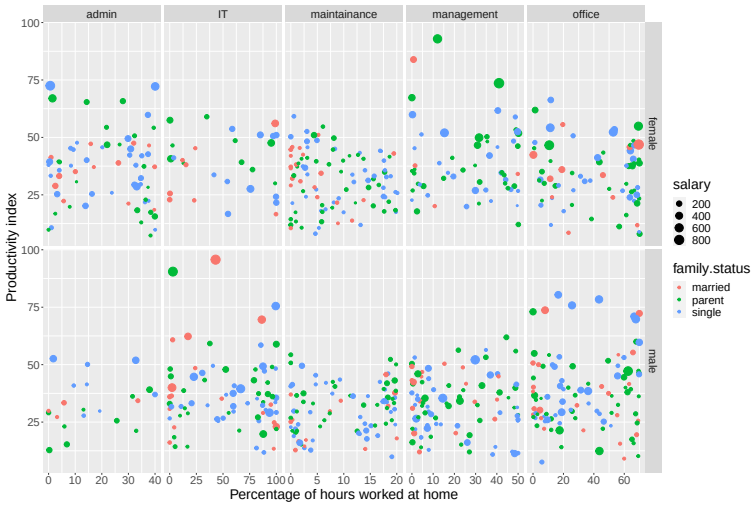
(iii) Dividing home hours using an arbitrary cut-off can miss some important details. They should at least consider using a continuous scale for the plots.

(iv) Since the focus is on the effect of working from home on productivity, we should be comparing bars for the same gender and family status, so these bars should be adjacent.

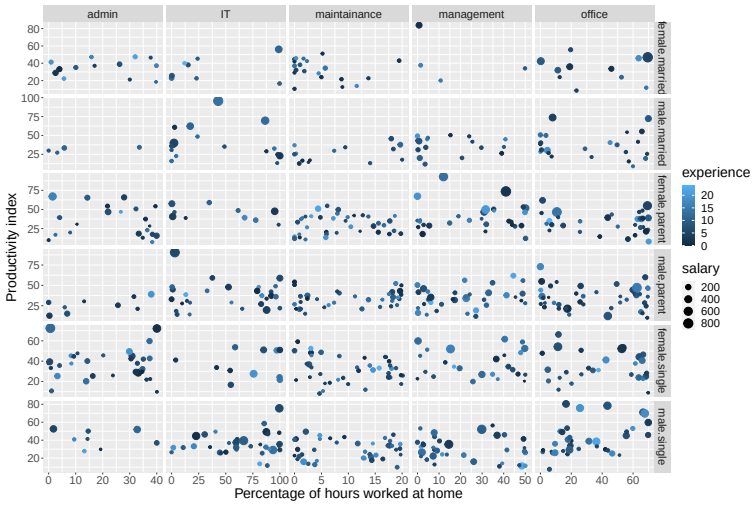
(v) The plot shows only averages, which may not tell the whole story. Showing the range of possibilities would be better. For example, a boxplot or scatterplot would give a better indication.

We should certainly divide by job type. It makes sense to consider proportion of work done from home, rather than total hours. We might consider using a scatterplot, rather than a bar chart, using the continuous proportion of hours worked from home, rather than the total number. Adding salary and experience to the plot could also be useful. However, adding both will probably make the plot too cluttered, so we might choose to add just salary.

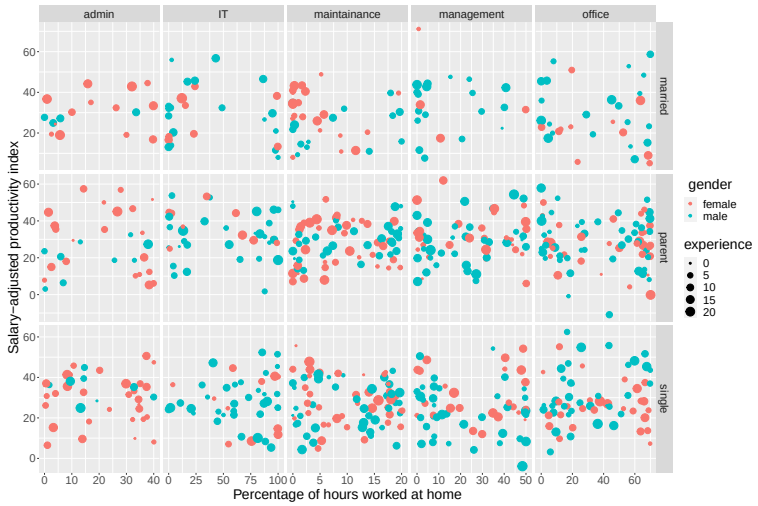
Since the proportion of hours worked from home varies a lot with type of work, it makes sense to plot a facet grid for type of work, and use different scales for each plot.



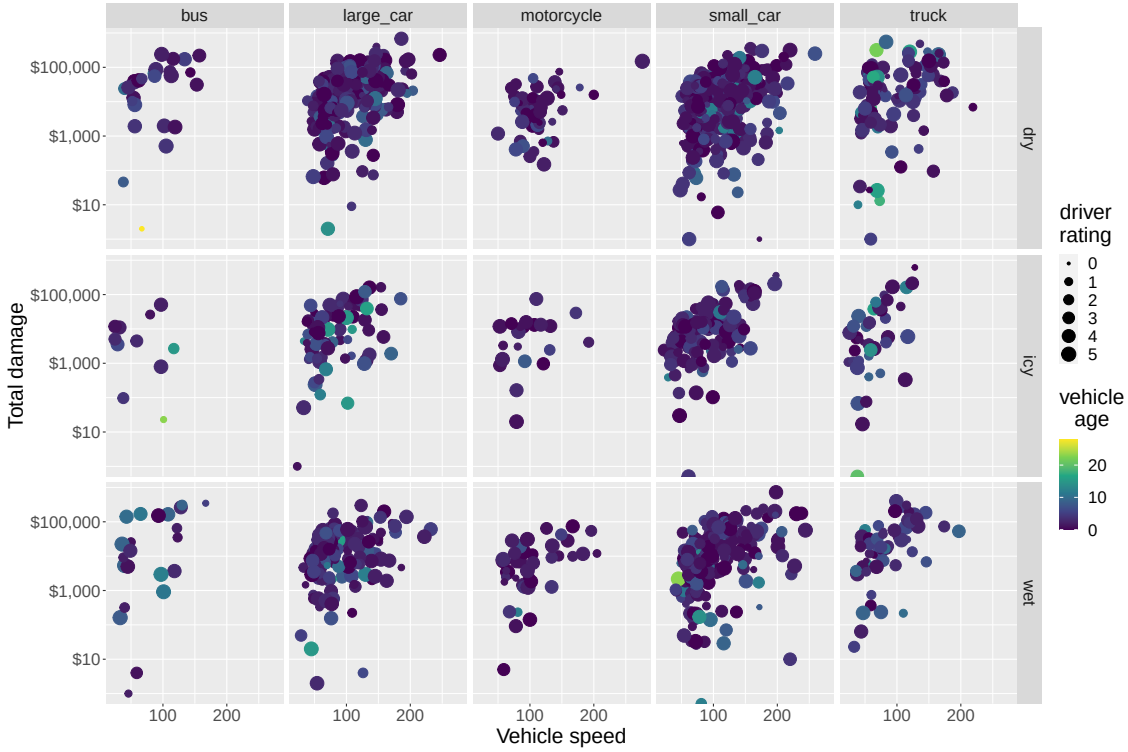
If we want to include more information, we might try to use an interaction term in the facet grid, to include more information in the plot, at the cost of making each subplot smaller.



Another approach we might try is to attempt to adjust for variables not included in the plot. For example, if we fit a linear model with response productivity, the coefficient of salary is approximately 0.05, so we could use `productivity.index` $-$ 0.05`salary` as the response to more clearly see the effect of other variables.



4. *Use* `ggplot` *to produce the following plot from the data in file* `HW1Q4.txt`*. [Make sure to reproduce all aspects of the plot — axis scales, labels, etc.]*



The code that originally produced the figure is

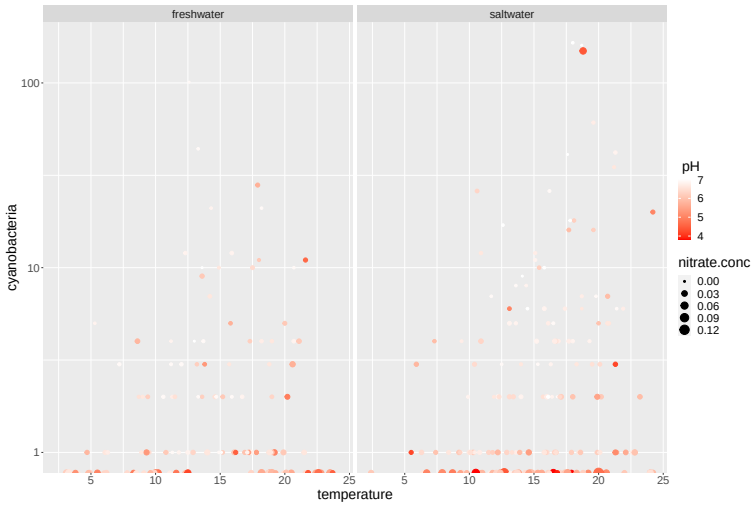
```
ggplot(HW1Q4,
       mapping=aes(y=total.damage,
                   x=vehicle.speed,
                   colour=vehicle.age,
                   size=driver.rating))+
    geom_point()+
    scale_y_log10(name="Total damage",labels=scales::dollar)+
    facet_grid(road.condition~vehicle.type)+
    scale_colour_viridis_c(name="vehicle\nage")+
    largertextsize+
    scale_size(name="driver\nrating")+
    scale_x_continuous(name="Vehicle speed")
```

5. *The file* `HW1Q5.txt` *contains the following data from an experiment into the effect of pollution on bacterial growth in lakes.*

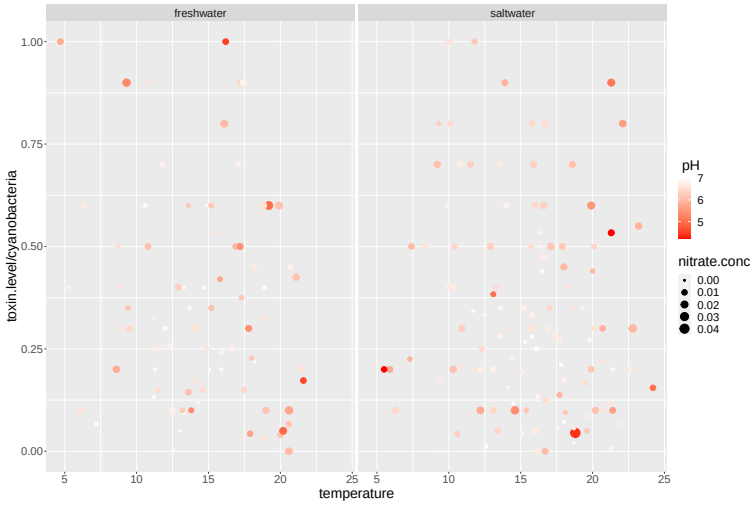
Variable	*Meaning*
nitrate.conc	*Concentration of nitrates in the lake*
phosphate.conc	*Concentration of phosphates in the lake*
pH	*pH of the lake (0=strong acid, 14=strong alkali, 7=neutral)*
salt	*Concentration of salt in the lake*
temperature	*Water temperature of the lake.*
weekly.rainfall	*Total rainfall in the past week (mm)*
cyanobacteria	*Abundance of cyanobacteria in the lake*
toxin.level	*Concentration of toxins in the lake*

Construct a plot or plots to show these data for the purpose of data exploration.

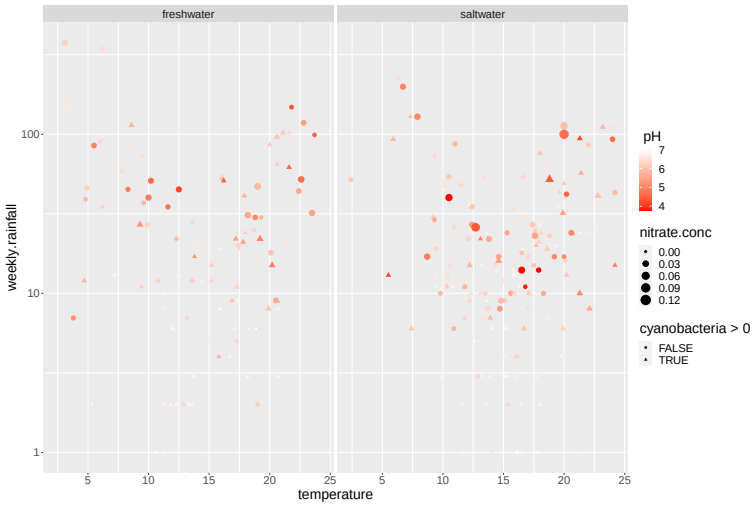
There are a lot of numeric variables in this data. For salt, we see that most lakes have salt close to zero, as they are freshwater. It may therefore make sense to replace this variable by a categorical variable with levels "fresh" and "salt", which can then be represented by shape. Given the high correlation between cyanobacteria and toxin, we might plot only one of them, or we might plot the ratio. One approach would be to plot cyanobacteria as the response in one plot, and the ratio as the response in another plot. We use a log transformation for cyanobacteria.



For the ratio, we need to remove data points with zero cyanobacteria.



Another approach to dealing with the heavy-tailed response is to break it into a logical variable, with a chosen cut-off. For example, we could plot the points which have non-zero cyanobacteria counts, using the shape aesthetic.



This plot allows us to see some patterns, but it is slightly tricky to judge which regions of the plot are more likely to have cyanobacteria, and which are less likely.

These plots were produced using the following code:

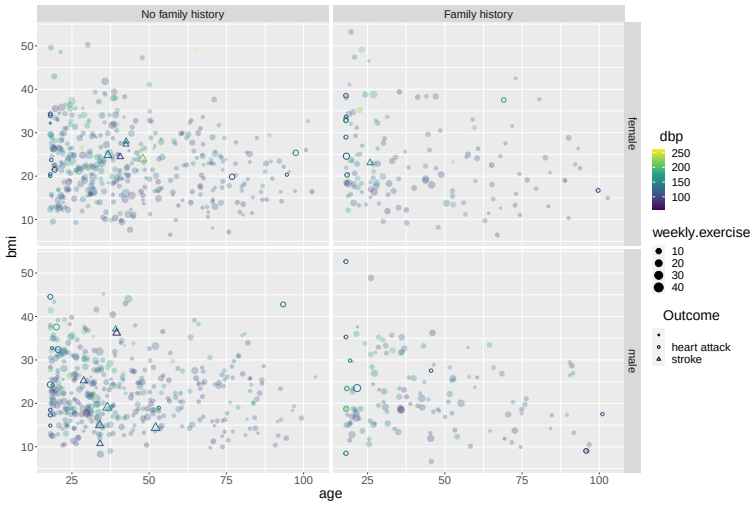
```
ggplot(HW1Q5, mapping=aes(x=temperature,
                          y=cyanobacteria,
                          colour=pH,
                          size=nitrate.conc))+
    geom_point()+scale_y_log10()+largertextsize+
    facet_wrap(salt>0~.,labeller=as_labeller(c("FALSE"="freshwater",
                                               "TRUE"="saltwater")))+
    scale_colour_gradient2(low="red",mid="white",high="blue",midpoint=7)
### Traditionally for pH red=acid, blue=alkali.


### Separate plot of toxin.level/cyanobacteria
ggplot(HW1Q5%>%filter(cyanobacteria>0),
       mapping=aes(x=temperature,
                   y=toxin.level/cyanobacteria,
                   colour=pH,
                   size=nitrate.conc))+
    geom_point()+largertextsize+
    facet_wrap(salt>0~.,labeller=as_labeller(c("FALSE"="freshwater","TRUE"="saltwater")))+
    scale_colour_gradient2(low="red",mid="white",high="blue",midpoint=7)

### Using shape to indicate presence of cyanobacteria.
ggplot(HW1Q5, mapping=aes(x=temperature,
                          y=weekly.rainfall,
                          size=nitrate.conc,
                          colour=pH,
                          shape=cyanobacteria>0))+
    geom_point()+largertextsize+
    facet_wrap(salt>0~.,labeller=as_labeller(c("FALSE"="freshwater","TRUE"="saltwater")))+
    scale_colour_gradient2(low="red",mid="white",high="blue",midpoint=7)+scale_y_log10()
```

6. *A doctor collects the following data on patients. The data are contained in the file* `HW1Q6.txt` *and include the following variables:*

Variable	*Meaning*
age	*The patient's age*
sex	*The patient's sex*
weekly.exercise	*The number of hours per week spent exercising*
daily.calorie.intake	*The patient's average estimated daily number of calories*
family.history	*Whether the patient has a family history of heart disease*
bmi	*The patient's BMI*
dbp	*The patient's diastolic blood pressure*
year.heart.attack	*Whether the patient suffers a heart attack in the year from the appointment.*
year.stoke	*Whether the patient suffers a stroke in the year from the appointment.*

Make a plot to show these data.

Here we have a two discrete response variables. Furthermore, the classes are very unbalanced. We can represent the response using colour or shape. Shape can be difficult to identify, but with a good choice of shape, and using alpha to also indicate these points, it can actually be fairly clear, allowing us to show a lot of information on the plot.

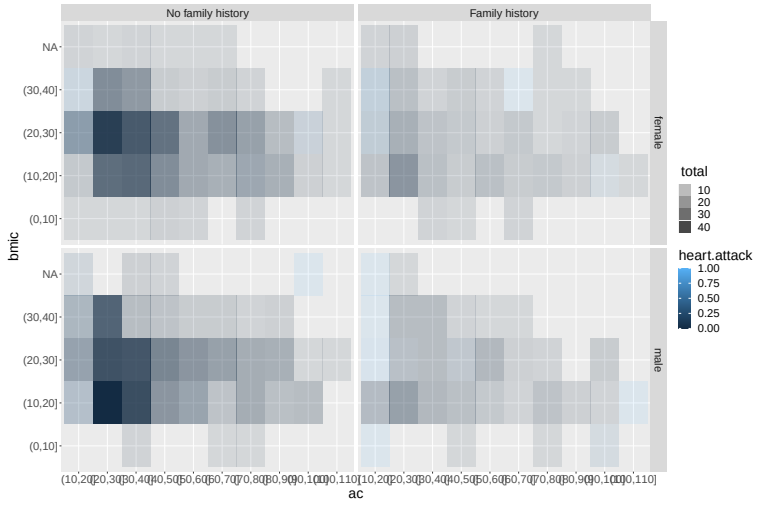


If we prefer to use colour for the outcome, we do not have so many channels for continuous predictors, but the heart attack and stroke outcomes are much more obvious.

As the outcome is categorical, using the x-axis or y-axis is not ideal. A facet wrap is also not ideal for the response variable, as it does not allow easy comparison of different outcomes.

An alternative approach is to plot the relative density of heart attacks for

each value using `dplyr` to summarise each rectangle in a grid.

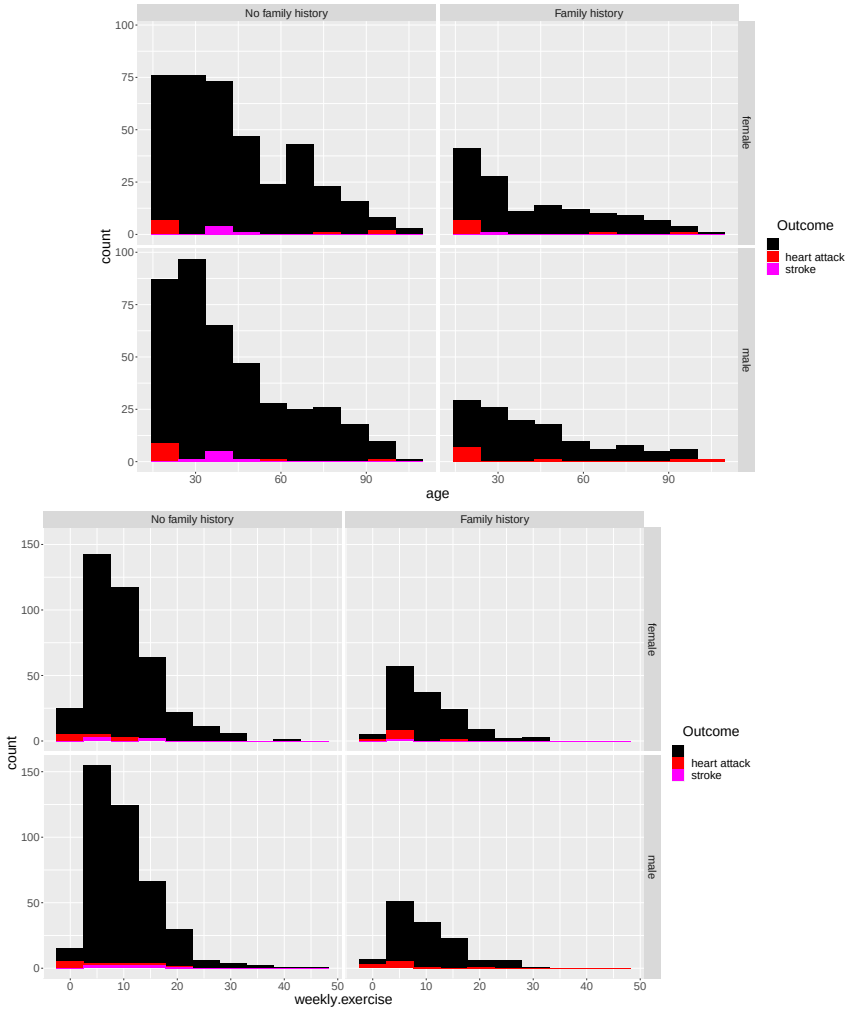


This data set is slightly sparse for this to be an effective plot. Low-abundance grid cells have too much variance. For a much larger data set, this approach might work well.

These plots were produced using the following code:

```
### Using shape to indicate the outcome.
### Need to choose shapes carefully, and use alpha to make the outcome clear.
ggplot(HW1Q6, mapping=aes(shape=interaction(year.heart.attack, year.stroke),
                          x=age,
                          y=bmi,
                          size=weekly.exercise,
                          colour=dbp,
                          alpha=year.heart.attack+year.stroke))+
    geom_point()+
    scale_shape_manual(name="Outcome", values=c(16,1,2),
                       labels=c("","heart attack","stroke"))+
    scale_colour_viridis_c()+
    facet_grid(sex~family.history,
               labeller=as_labeller(c("FALSE"="No family history",
                                      "TRUE"="Family history",
                                      "male"="male",
                                      "female"="female")))+
    scale_alpha_continuous(range=c(0.3,1))+largertextsize+guides(alpha="none")

### Using colour to indicate outcome is much clearer, but loses a channel
### that could be used for a predictor.
ggplot(HW1Q6, mapping=aes(colour=interaction(year.heart.attack, year.stroke),
                          x=age,
                          y=bmi,
                          size=weekly.exercise,
                          alpha=year.heart.attack+year.stroke))+
    geom_point()+
    scale_colour_manual(name="Outcome",
                        values=c("black","red","magenta"),
                        labels=c("","heart attack","stroke"))+
    facet_grid(sex~family.history,
               labeller=as_labeller(c("FALSE"="No family history",
                                      "TRUE"="Family history",
                                      "male"="male",
                                      "female"="female")))+
    scale_alpha_continuous(range=c(0.3,1))+largertextsize+guides(alpha="none")

### Density plots can be good for this type of data, but can be too
### variable if the density is low.
ggplot(HW1Q6%>%mutate(ac=cut(age, breaks=10*seq_len(125)),
                      bmic=cut(bmi, breaks=c(0,10,20,30,40)))%>%
       group_by(ac,bmic,sex,family.history)%>%
       summarise(total=n(),
                 heart.attack=mean(year.heart.attack),
                 stroke=mean(year.stroke)),
       mapping=aes(fill=heart.attack, alpha=total, x=ac, y=bmic))+
    geom_tile()+largertextsize+
    facet_grid(sex~family.history,
               labeller=as_labeller(c("FALSE"="No family history",
                                      "TRUE"="Family history",
                                      "male"="male",
                                      "female"="female")))
```

For this data, stacked histograms might work, perhaps one for each predictor, for example, the following two plots show age and weekly exercise.



These plots were produced using the following code:

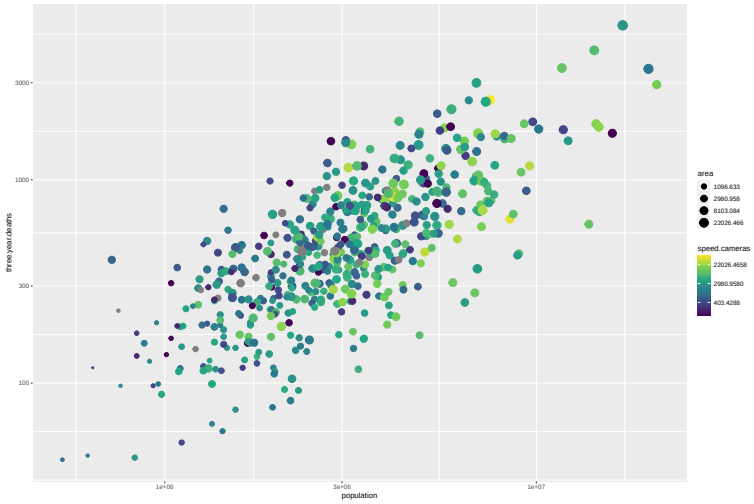
```
### Stacked histograms don't have so many channels but do give a clear
### picture for the channels shown.
ggplot(HW1Q6,mapping=aes(fill=as.factor(year.heart.attack+2*year.stroke),x=age))+
    geom_histogram(bins=10)+largertextsize+
    facet_grid(sex~family.history,
               labeller=as_labeller(c("FALSE"="No family history",
                                      "TRUE"="Family history",
                                      "male"="male",
                                      "female"="female")))+
    scale_fill_manual(name="Outcome",values=c("black","red","magenta"),
                      labels=c("","heart attack","stroke"))

ggplot(HW1Q6,mapping=aes(fill=as.factor(year.heart.attack+2*year.stroke),
                         x=weekly.exercise))+
    geom_histogram(bins=10)+largertextsize+
    facet_grid(sex~family.history,
               labeller=as_labeller(c("FALSE"="No family history",
                                      "TRUE"="Family history",
                                      "male"="male",
                                      "female"="female")))+
    scale_fill_manual(name="Outcome",values=c("black","red","magenta"),
                      labels=c("","heart attack","stroke"))
```

7. *The file* `HW1Q7.txt` *contains data on the effect of traffic cameras on road safety.*

Variable name	*Meaning*
population	*The population of the town or city*
area	*The area of the town or city.*
cars	*The number of cars in the town or city*
speed.cameras	*The total number of speed cameras in the city.*
bicycle.lanes	*The proportion of roads with bicycle lanes.*
three.year.deaths	*The number of deaths in traffic accidents during the past 3 years.*

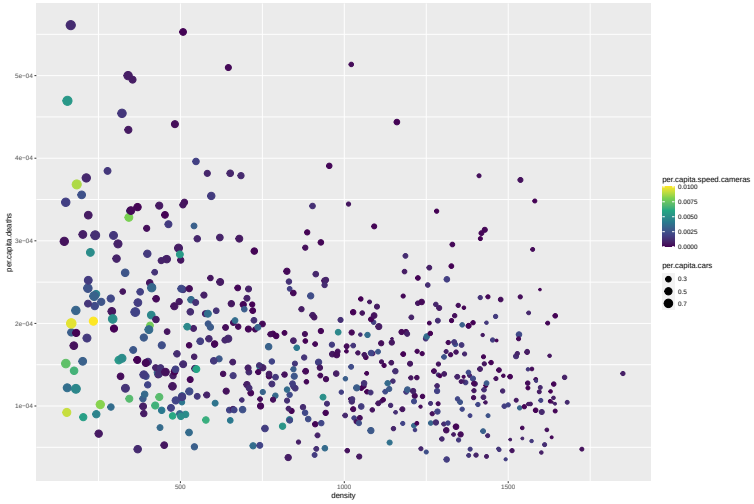
(a) Produce a figure to show these data for the purpose of data exploration.

There are 6 continuous variables in this dataset. We could use (x, y) coordinates, colour and size to show 4 of them — for example in the following plot:



Since this is for data exploration, I have not spent so much time tidying up axis labels — for data exploration, you know what the variables are. Using size for area is a natural choice, as it is intuitive. Because the distributions are heavy-tailed, log transformations are appropriate for all variables.

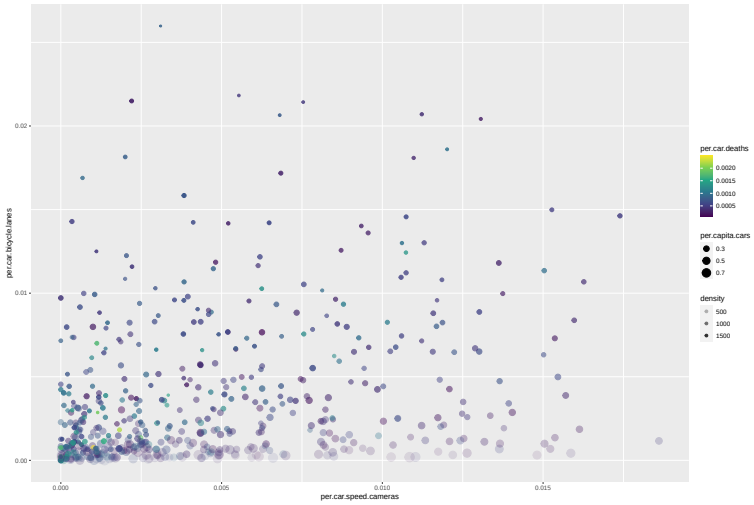
From this first plot, the difficulty is immediately clear — the variables are all highly correlated, which makes sense. Instead of plotting the original variables, it may be more informative to convert to per-capita varibles.



Here, I have divided most variables by population. For area, I have taken the inverse `population/area` as this is a more commonly-used measure. The per-capita variables are not so heavy-tailed, so log-transformation is not crucial, but might still be a good idea.

These per-capita variables are still highly correlated. An alternative is to

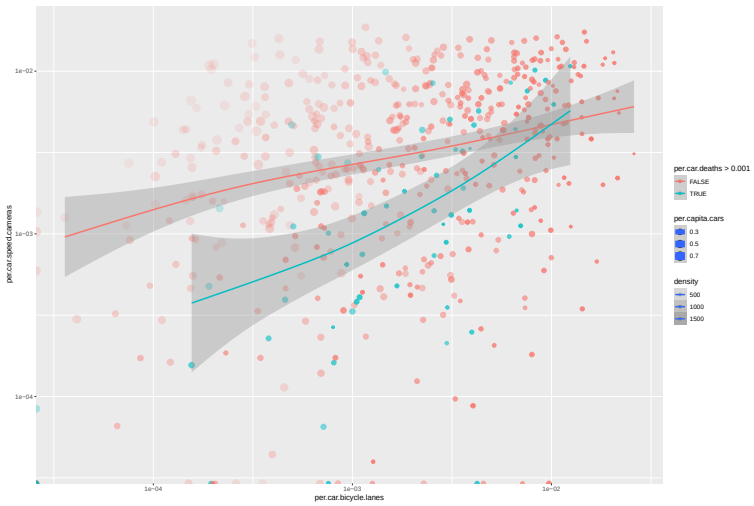
measure deaths per car. The predictors are still very correlated. We might try using colour for deaths to highlight which values of other variables lead to high deaths. This allows us to put predictors of interest — speed cameras and bicycle lanes on the axes.



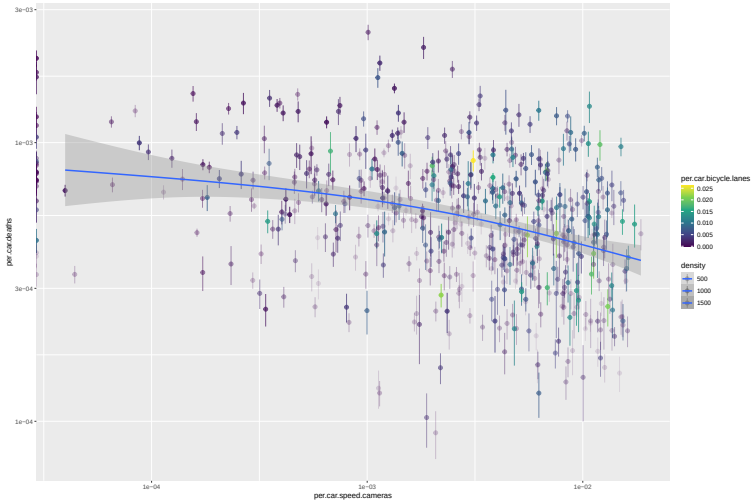
These plots were produced using the following code:

```
### Direct plot without creating better features
ggplot(HW1Q7, mapping=aes(x=population,
                          y=three.year.deaths,
                          size=area,
                          colour=speed.cameras))+
    geom_point()+scale_x_log10()+scale_y_log10()+
    scale_size_continuous(trans="log")+scale_colour_viridis_c(trans="log")
## Use log transformations, as predictors are heavy-tailed.


### Use per-capita variables
ggplot(HW1Q7%>%mutate(density=population/area,
                      per.capita.cars=cars/population,
                      per.capita.speed.cameras=speed.cameras/population,
                      per.capita.bicycle.lanes=bicycle.lanes/population,
                      per.capita.deaths=three.year.deaths/population),
       mapping=aes(x=density,
                   y=per.capita.deaths,
                   size=per.capita.cars,
                   colour=per.capita.speed.cameras))+
    geom_point()+scale_size_continuous()+scale_colour_viridis_c()

### Use per-car variables
ggplot(HW1Q7%>%mutate(density=population/area,
                      per.capita.cars=cars/population,
                      per.car.speed.cameras=speed.cameras/cars,
                      per.car.bicycle.lanes=bicycle.lanes/cars,
                      per.car.deaths=three.year.deaths/cars),
       mapping=aes(x=per.car.speed.cameras,
                   y=per.car.bicycle.lanes,
                   colour=per.car.deaths,
                   size=per.capita.cars,
                   alpha=density))+
    geom_point()+scale_colour_viridis_c()
```

Some of the above plots are difficult to assess. It might be easier to choose a threshold, and highlight points above this threshold. This also has the advantage of allowing us to fit a smooth curve for low deaths and high deaths to more easily summarise the results.



One issue with the transformation to per-capita variables is that the number of deaths is probably a Poisson sample, so the variance depends on the sample size — small populations have much larger relative errors. Using alpha to show population can indicate this by making less reliable points faded. A different approach is to add error bars to the response.

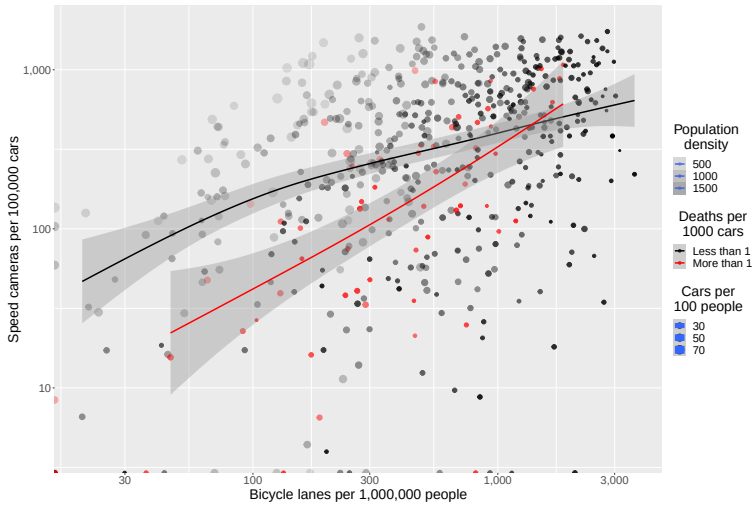


These plots were produced using the following code:

```
### Use colour to highlight high-risk cities
ggplot(HW1Q7%>%mutate(density=population/area,
                      per.capita.cars=cars/population,
                      per.car.speed.cameras=speed.cameras/cars,
                      per.car.bicycle.lanes=bicycle.lanes/cars,
                      per.car.deaths=three.year.deaths/cars),
       mapping=aes(y=per.car.speed.cameras,
                   x=per.car.bicycle.lanes,
                   colour=per.car.deaths>0.001,
                   size=per.capita.cars,alpha=density))+
    geom_point()+scale_y_log10()+geom_smooth(method="gam")+scale_x_log10()

### plot with error bars.
ggplot(HW1Q7%>%mutate(density=population/area,per.capita.cars=cars/population,
                      per.car.speed.cameras=speed.cameras/cars,
                      per.car.bicycle.lanes=bicycle.lanes/cars,
                      per.car.deaths=three.year.deaths/cars,
                      per.car.deaths.sd=sqrt(three.year.deaths)/cars),
       mapping=aes(y=per.car.deaths,
                   x=per.car.speed.cameras,
                   colour=per.car.bicycle.lanes,
                   alpha=density,
                   ymin=per.car.deaths-2*per.car.deaths.sd,
                   ymax=per.car.deaths+2*per.car.deaths.sd))+
    geom_pointrange()+scale_y_log10()+geom_smooth(method="gam")+
    scale_x_log10()+scale_colour_viridis_c()
```

(b) After analysing the data, you conclude that for fixed values of the other parameters, `speed.cameras` *is negatively associated with* `three.year.deaths` *when* `bicycle.lanes` *is relatively low for the population; but when* `bicycle.lanes` *is high for the population* `speed.cameras` *may be positively associated with* `three.year.deaths`*. Make a plot that emphasises these conclusions.*

The fourth plot in part (a) shows this to some extent. We could tidy that plot up a little, and change to per-capita bicycle lanes, rather than per-car, as indicated in the conclusion.



Alternatively, we can use a facet wrap with a cut-off to distinguish low and high numbers of bicycle lanes.

Adding error bars to this plot as in the fifth plot in part (a) is also possible, but does not make the conclusion much easier to see.

These plots were produced using the following code:

```
### Using colour to indicate dangerous areas
ggplot(HW1Q7%>%mutate(density=population/area,
                      per.capita.cars=cars/population,
                      per.car.speed.cameras=speed.cameras/cars,
                      per.capita.bicycle.lanes=bicycle.lanes/population,
                      per.car.deaths=three.year.deaths/cars),
       mapping=aes(y=100000*per.car.speed.cameras,
                   x=1000000*per.capita.bicycle.lanes,
                   colour=per.car.deaths>0.001,
                   size=100*per.capita.cars,
                   alpha=density))+
    geom_point()+geom_smooth(method="gam")+largertextsize+
    scale_y_log10(name="Speed cameras per 100,000 cars",labels=scales::comma)+
    scale_x_log10(name="Bicycle lanes per 1,000,000 people",labels=scales::comma)+
    scale_colour_manual(name="Deaths per\n1000 cars",values=c("black","red"),labels=c("Le
    scale_alpha_continuous(name="Population\ndensity")+
    scale_size_continuous(name="Cars per\n100 people")

### Using y-axis for deaths, and a facet_wrap on bicycle lanes.
ggplot(HW1Q7%>%mutate(density=population/area,
                      per.capita.cars=cars/population,
                      per.car.speed.cameras=speed.cameras/cars,
                      per.capita.bicycle.lanes=bicycle.lanes/population,
                      per.car.deaths=three.year.deaths/cars),
       mapping=aes(x=100000*per.car.speed.cameras,
                   colour=1000000*per.capita.bicycle.lanes,
                   y=1000*per.car.deaths,
                   size=100*per.capita.cars,alpha=density))+
    geom_point()+geom_smooth(method="gam")+largertextsize+
    scale_y_log10(name="deaths per 1,000 cars",labels=scales::comma)+
    scale_x_log10(name="Speed cameras per 100,000 cars",labels=scales::comma)+
    scale_colour_viridis_c(name="Bicycle lanes\nper 1,000,000 people")+
    scale_alpha_continuous(name="Population\ndensity")+
    scale_size_continuous(name="Cars per\n100 people")+
    facet_wrap(per.capita.bicycle.lanes>0.001~.,
               labeller=as_labeller(c(
                   "FALSE"="Less than 1,000 bicycle lanes per 1,000,000 people",
                   "TRUE"="Over 1,000 bicycle lanes per 1,000,000 people")),
               scales="free_x")
```