STAC33 TUT-2

Introduction

We will be discussing problems from ch:3 and ch:4 from the PASIAS(Problems and Solutions in Applied Statistics). Ch:3 focuses on drawing graphs and ch:4 is about exploring data. So hopefully by the end of this session we are familiar with what graphs to draw given a problem and also perform introductory data analysis given a dataset.

The chapters can be found here: [https://ritsokiguess.site/pasias/drawing-graphs.html]

CH:3.6 Juice problem

First load the library tidyverse

Load data

1) Specify the URL (optional BUT recomended)

2) load data 2.1) `read_delim()` -> used to load data separated by a certain character which needs to be specefied

```
url <- "http://ritsokiguess.site/datafiles/ojuice.txt"
juice <- read_delim(url, " ")
```

```
## Rows: 24 Columns: 3

## -- Column specification --------------------------------------------------------
## Delimiter: " "
## dbl (3): run, sweetness, pectin

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Lets look at some of data `glimpse()` -> shows the values along with characteristics `head()` -> shows first few values

```
glimpse(juice)
```

```
## Rows: 24
## Columns: 3
## $ run       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 1~
## $ sweetness <dbl> 5.2, 5.5, 6.0, 5.9, 5.8, 6.0, 5.8, 5.6, 5.6, 5.9, 5.4, 5.6, ~
## $ pectin    <dbl> 220, 227, 259, 210, 224, 215, 231, 268, 239, 212, 410, 256, ~
```

```
head(juice)
```

```
## # A tibble: 6 x 3
##     run sweetness pectin
##   <dbl>     <dbl>  <dbl>
## 1     1       5.2    220
## 2     2       5.5    227
## 3     3       6      259
## 4     4       5.9    210
## 5     5       5.8    224
## 6     6       6      215
```

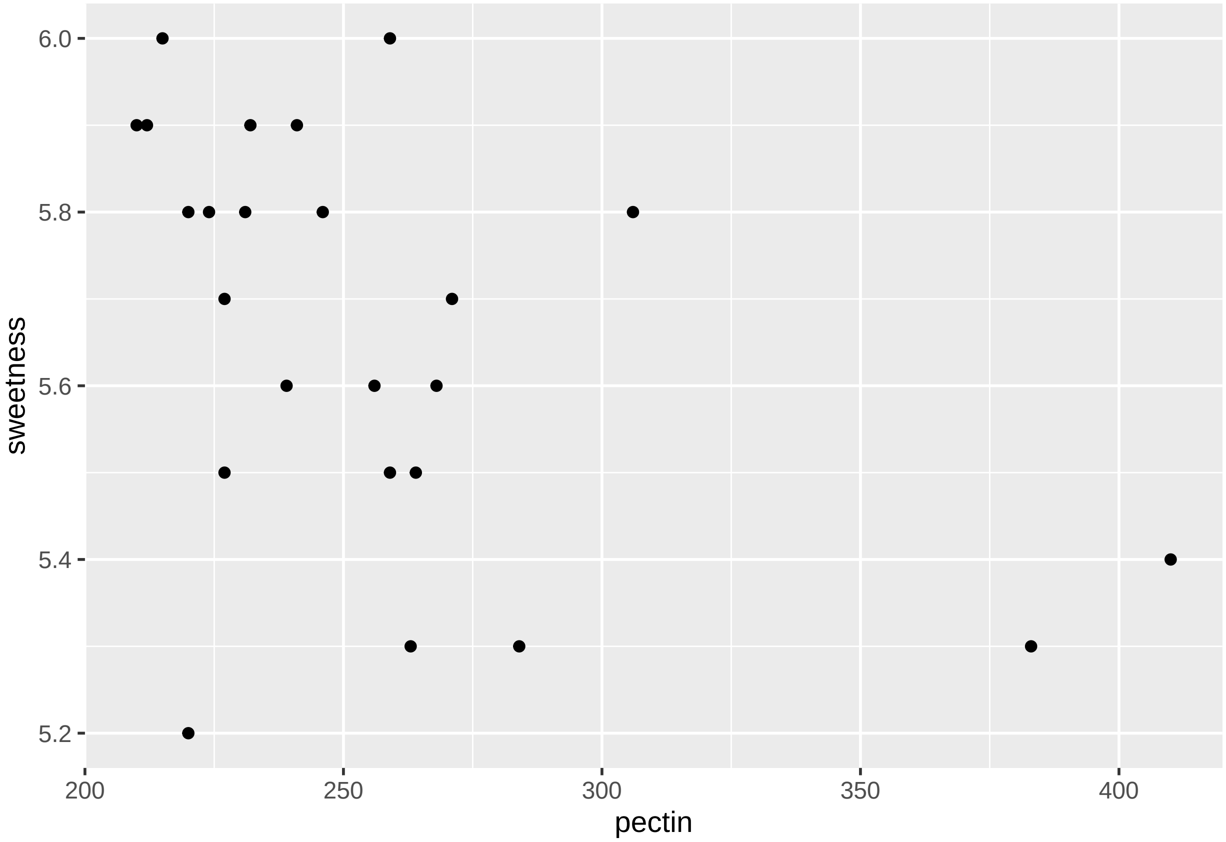
- Variables are -> **run, sweetness, pectin**

Goal: find a relation between the 2 variables

1) Note the TYPE of the variable involved. (Quantitative, sweetness appears to be continuous)
2) Choose Scatter plot to observe a trend
3) Then Draw a line between the plot to see the relation

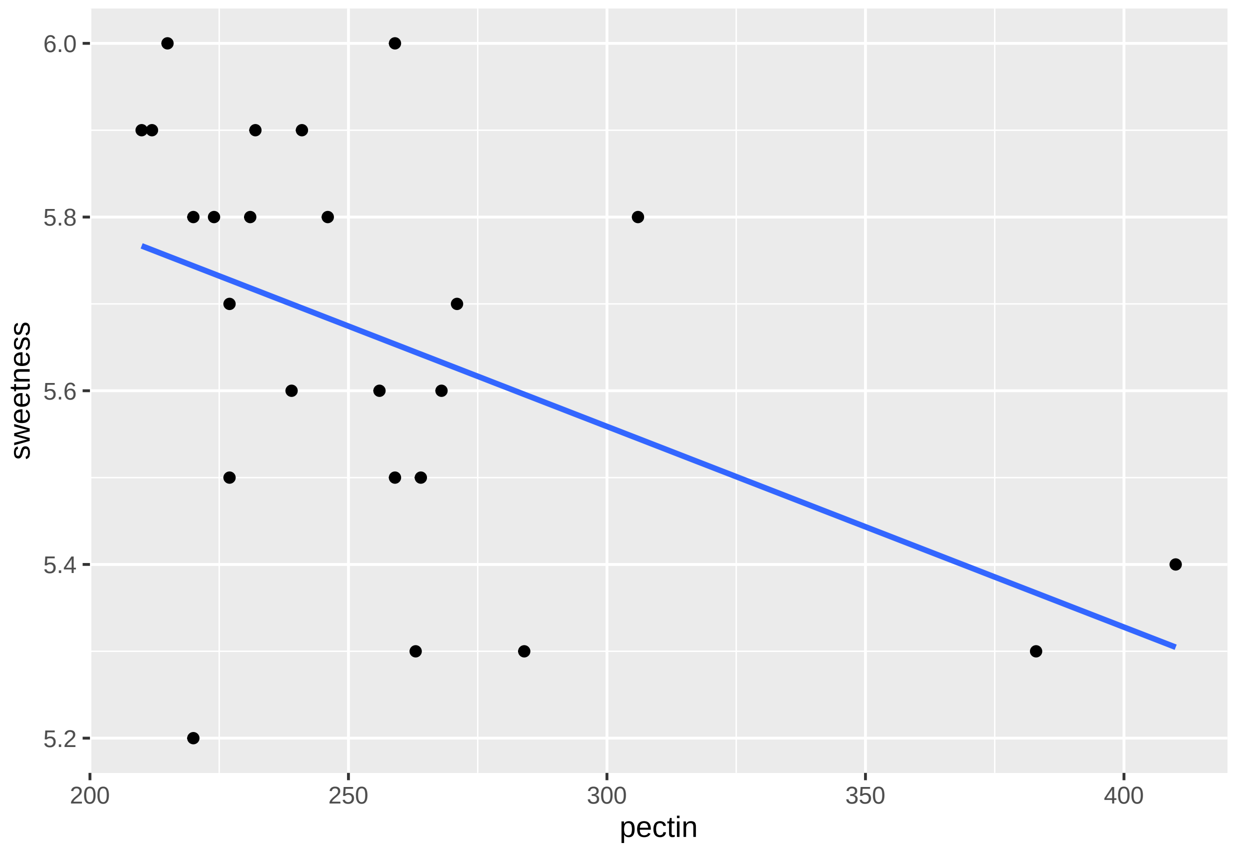
`ggpplot()` -> plotting package in tidyverse that provides helpful commands to create plots Think of this as a masterchef you call for a day who will be helping you make your meal, this chef will setup the place and the setting for you but you have to make the choice of using a knife vs a peeler depending on the task

```r
# just the scatter plot
ggplot(juice, aes(x = pectin, y = sweetness)) + geom_point()
```



```r
# with the line of best-fit
ggplot(juice, aes(x = pectin, y = sweetness)) + geom_point() + geom_smooth(method = lm, se= F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
## method = lm for best fit
## se = F removes the confidence bands around the line
```

- General downwards trend with the increase in pectin
- But not really convincing as there appears to be 2 extreme values above 350 BUT they are also follow the same trend

Summary

- When asked to come up with the relationship between 2 quantities(both or atleast 1 being Quantitative) we make a scatter plot and observe the trend between them

3.7 Making soap

```
url <- "http://ritsokiguess.site/datafiles/soap.txt"
soap <- read_delim(url, " ")
```

```
## Rows: 27 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: " "
## chr (1): line
## dbl (3): case, scrap, speed
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Lets look at some of the data now along with some properties of our data `head()` -> shows first few values `summary()` -> characteristics of the data `table()` -> shows the different UNIQUE values along with their counts, under a variable preferbly used for CATAGORICAL variables

```
head(soap)
```

```
## # A tibble: 6 x 4
##    case scrap speed line
##   <dbl> <dbl> <dbl> <chr>
## 1     1   218   100 a
## 2     2   248   125 a
## 3     3   360   220 a
## 4     4   351   205 a
## 5     5   470   300 a
## 6     6   394   255 a
```

```
summary(soap)
```

```
##       case          scrap           speed           line
##  Min.   : 1.0   Min.   :140.0   Min.   :100.0   Length:27
##  1st Qu.: 7.5   1st Qu.:256.0   1st Qu.:162.5   Class :character
##  Median :14.0   Median :331.0   Median :205.0   Mode  :character
##  Mean   :14.0   Mean   :315.5   Mean   :210.2
##  3rd Qu.:20.5   3rd Qu.:375.5   3rd Qu.:267.5
##  Max.   :27.0   Max.   :470.0   Max.   :320.0
```

```
# I use this as I found it on stackoverflow once don't use this when asked to
# show the result as the generated output is not clean
table(soap$line)
```

```
##
##  a  b
## 15 12
```

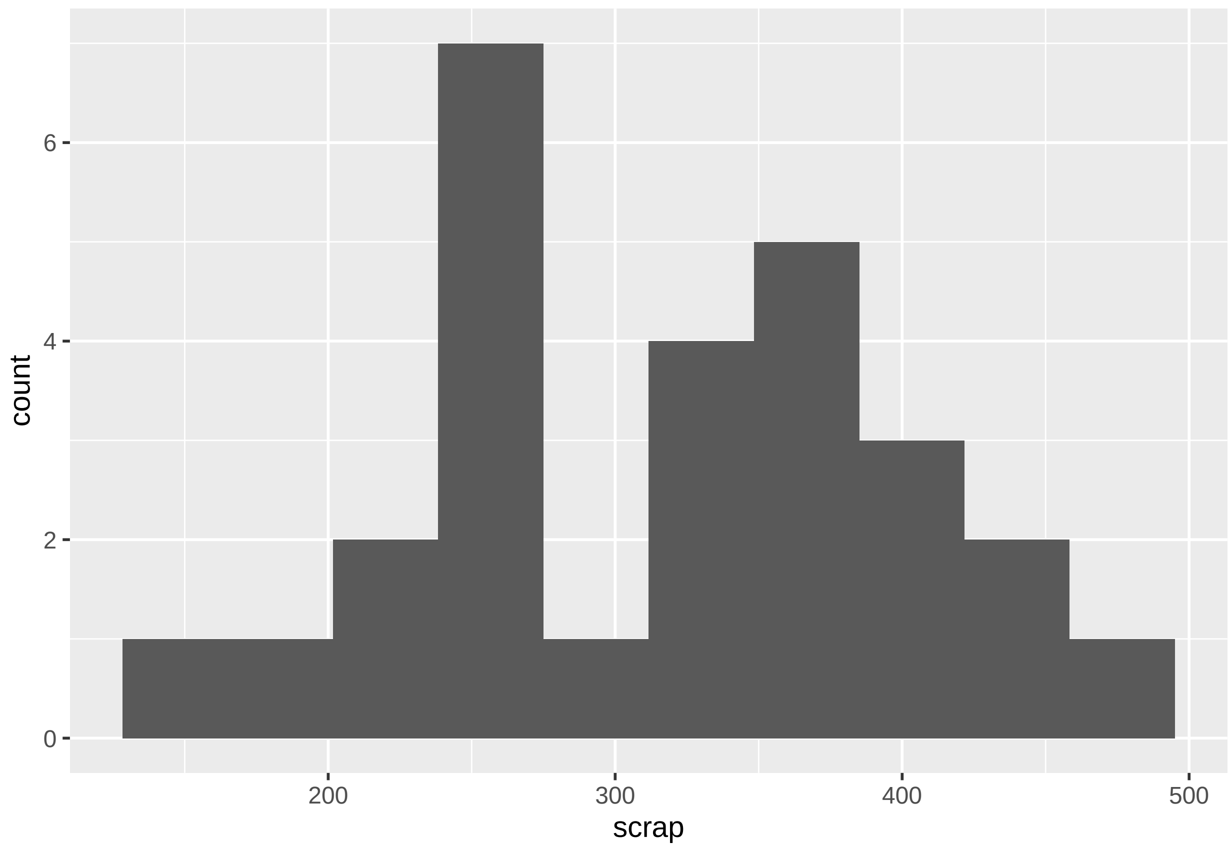
Make a histogram from SCRAP

To make histogram we use `geom_histogram()` with an appropriate binsize

1) what are bins?
   The count you specify for the bins is the basic partition you are willing to make for your x values

2) Ideal number?
   theory vs application. just play around and fit what is ideal
   make sure it's never more than your actual sample size :p

```
soap %>% ggplot(aes( x= scrap)) + geom_histogram(bins = 10)
```

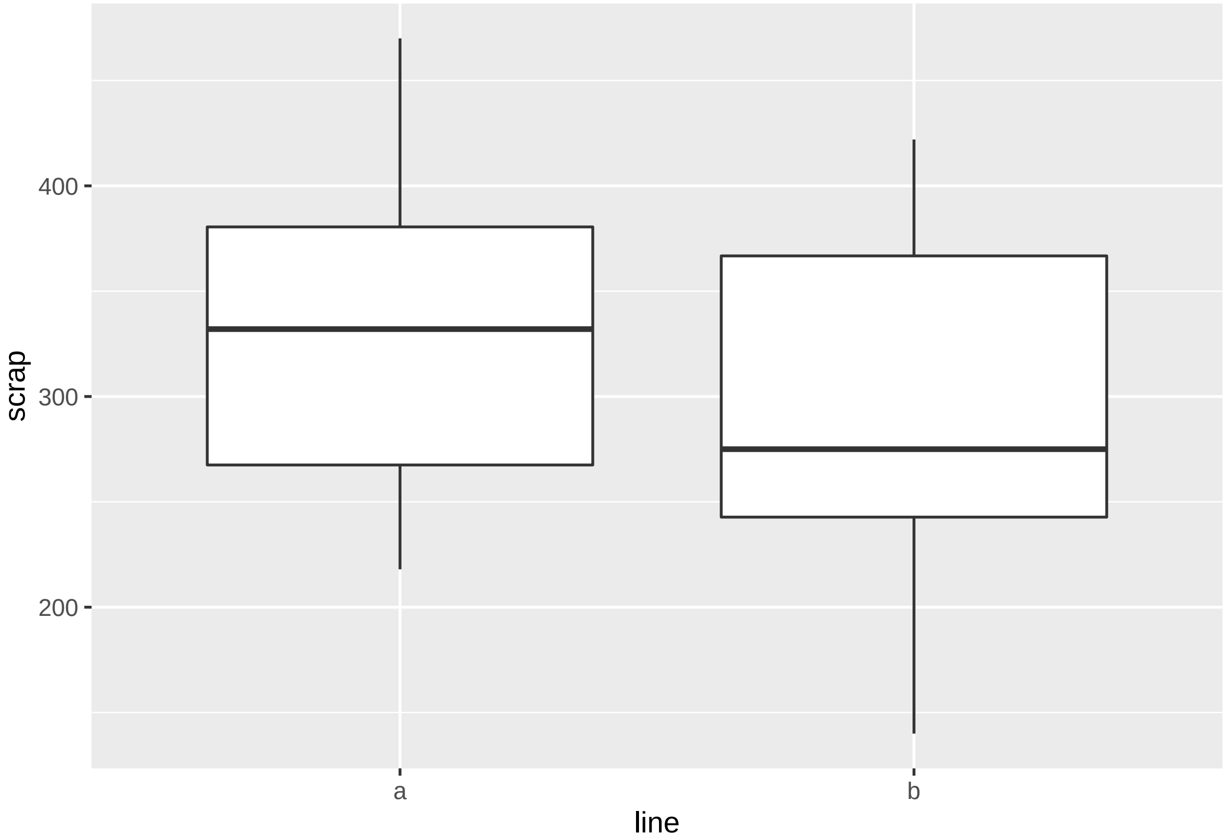


3) Features of the graph **Bimodail** ? type of modes **Range Skewness?** Sym

Make a side-by-side box-plot

`geom_boxplot()` is used to make the box-plot

```
ggplot(soap, aes(x = line , y = scrap)) + geom_boxplot()
```



3) Features of the graph **Range** between the 2 plants
Median, range of a > b
No **outlier**
BUT there is a huge overlap between their ranges
Never make inference about **MEAN** when discussing box-plot NOT reliable

Summary

Histogram for distribution and spread and mode
bins are for dividing the range of x-values
Box-plot for range if values + Q1 Q3 median + outliers

4.11

Reading the data and observing some of the data

```
my_url <- "https://raw.githubusercontent.com/nxskok/datafiles/master/compatt.txt"
anxiety <- read_delim(my_url," ")
```

```
## Rows: 35 Columns: 3
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: " "
## chr (1): gender
## dbl (2): CAS, CARS
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(anxiety)
```

```
## # A tibble: 6 x 3
##   gender   CAS  CARS
##   <chr>  <dbl> <dbl>
## 1 female  2.85  2.9
## 2 male    2.6   2.32
## 3 female  2.2   1
## 4 male    2.65  2.58
## 5 male    2.6   2.58
## 6 male    3.2   3.05
```

Number of males and females in our sample

```
# prefered way to use RESULT is a tibble
anxiety %>% group_by(gender) %>% summarise(count = n())
```

```
## # A tibble: 2 x 2
##   gender count
##   <chr>  <int>
## 1 female    15
## 2 male      20
```

```
# my hacky way from stackoverflow RESULT is a console output NOT RECOMENDED for displaying results
table(anxiety$gender)
```

```
##
## female   male
##     15     20
```

group_by() -> separates our data into different groups based on the specified variable summarise() -> used to make meaning summaries by declaring a new variable here we used n() and named that variable count

Mean and SD of CAS and CARS not seperated by gender

```
anxiety %>% summarise(mean_CAS = mean(CAS), mean_CARS = mean(CARS), sd_CAS = sd(CAS), sd_CARS = sd(CARS
```

```
## # A tibble: 1 x 4
##   mean_CAS mean_CARS sd_CAS sd_CARS
##      <dbl>     <dbl>  <dbl>   <dbl>
## 1     2.82      2.77  0.484   0.671
```

Doing it without mentioning the column name

```
anxiety %>% select(2,3) %>%
  summarise(across(everything(), list(mean = ~mean(.), sd = ~sd(.))))
```

```
## # A tibble: 1 x 4
##   CAS_mean CAS_sd CARS_mean CARS_sd
##      <dbl>  <dbl>     <dbl>   <dbl>
## 1     2.82  0.484      2.77   0.671
```

mean(), sd() -> calculates the mean and sd select() -> to choose the column with the position of the column in the dataframe across() -> to go over what the selected variables have in common list() ->

used because we are doing multiple opoerations on each variable

Same thing for all the quantitative variables

```
anxiety %>% summarize(across(where(is.numeric), list(m = ~mean(.), s = ~sd(.))))
```

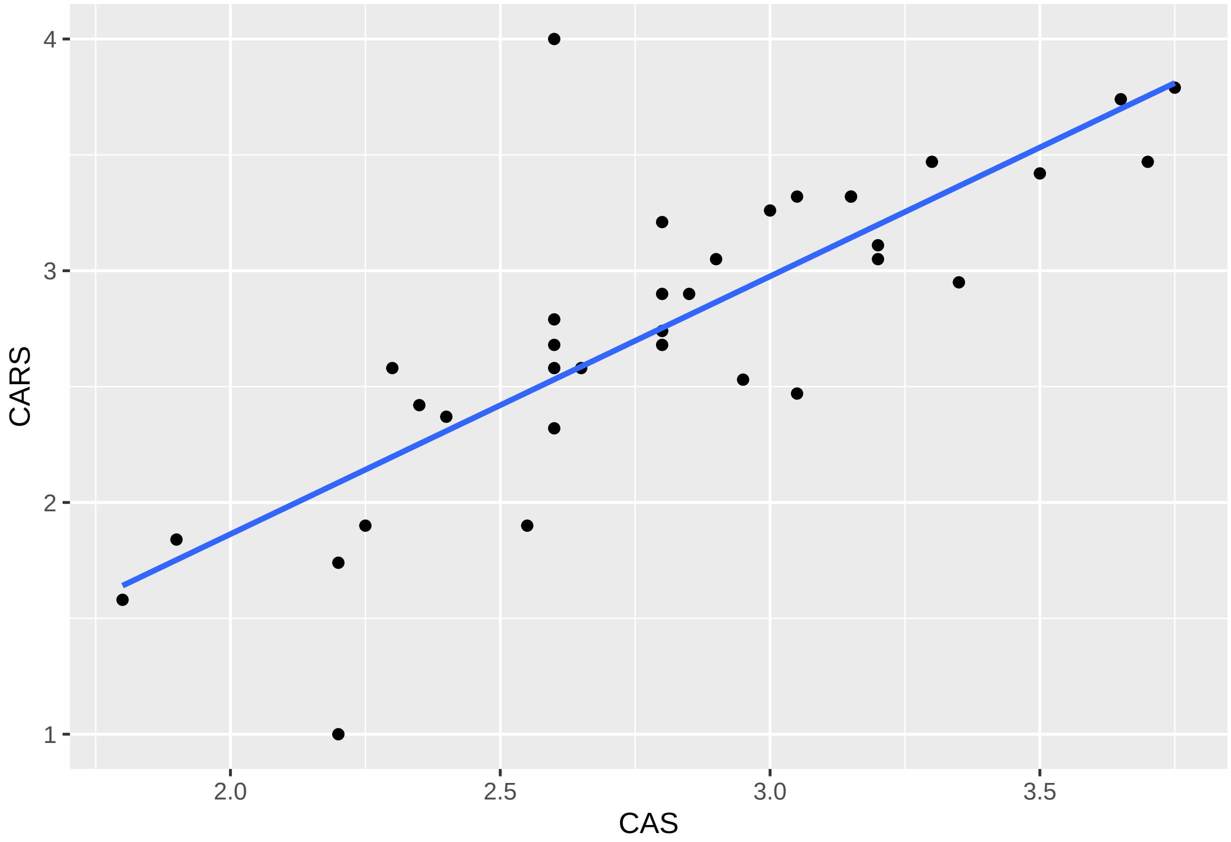
```
## # A tibble: 1 x 4
##   CAS_m CAS_s CARS_m CARS_s
##   <dbl> <dbl>  <dbl>  <dbl>
## 1  2.82 0.484   2.77  0.671
```

`where()` -> used to declare a conditional statement `is.numeric` -> the variable TYPE is numeric

Again to see and compare the relation between the 2 variables in each group

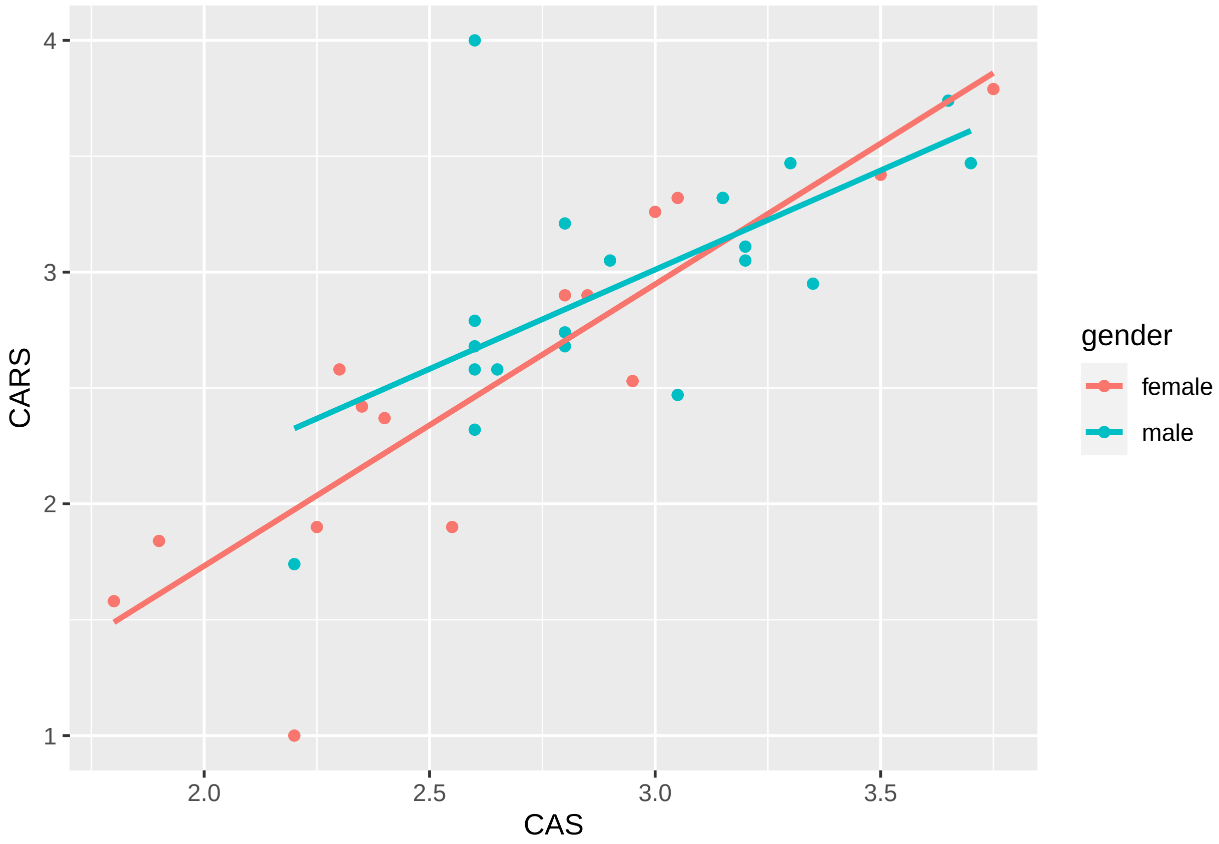
```
#general trend
ggplot(anxiety,aes(x=CAS,y=CARS)) + geom_point() + geom_smooth(method = lm, se=F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# trend in each gender group
ggplot(anxiety,aes(x=CAS,y=CARS, color = gender)) + geom_point() + geom_smooth(method = lm, se=F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



`color()` -> to separate the types in the categorical variable

Add and store the sum of CARS and CAS and compare the average of this SUM now between Male and Female

```
anxiety %>% mutate(sum = CARS + CAS) %>% group_by(gender) %>% summarise(avg = mean(sum))
```

```
## # A tibble: 2 x 2
##   gender   avg
##   <chr>  <dbl>
## 1 female  5.16
## 2 male    5.91
```

with `mutate()` we are first creating this new variable and then with the `summarise()` we are finding the mean of the sum for each gender.
Here note we are not storing any of these changes(notice environment) right now we are just temporarily working with them and discarding them after processing our results
To actually store the changes just declare it to a new variable.

`mutate()` -> used to make a new variable. You declare the name of the new variable and the assignment rule for that variable

Summary

`group_by()` -> use on catagorical variables and think of it and typing down knots based on the unique groups
`summarise()` -> creates summaries like mean mode median . . .

`where()` -> used for conditional statements think