Sleep Study Data

An example of Exploratory Data Analysis

Stat 212

Sleep Study

Data Description

Circadian rhythms refer to biological processes that have a period of around 24 hours. These processes allow organisms to coordinate behavior with the day/night cycle.

Circadian rhythms vary across individuals and have been categorized into three chronotypes; morning type, evening type and intermediate type. These refer to the time of day when individuals are most active. A number of studies have indicated that chronotypes differ in aspects of personality, such as depression, creativity, and sociosexuality.

Today we'll explore data collected by Onyper et al on a number of variables associated with sleep. The data was collected from 253 students from a small, Northeastern liberal arts college. Students recorded their sleep habits for the previous week, took a survey about attitudes and habits, and performed cognitive tests. Information about GPA and demographic characteristics were also recorded.

Statistical Process: Step 1

Import data to R

The line of code below is what allows us to take the file with the sleep data, named `SleepStudy.csv`, and bring it into our R session so we can explore it under the name `sleep`. All of the data we use will be stored in the `Data` folder under `Class`.

> This line of code will be very useful as you import *different* data sets for future examples and homework.

```
sleep <- read.csv("~/Stats 212 summer20/Class/Data/SleepStudy.csv")
```

A full list of variables is provided below.

Variable	Description
Gender	Gender (1 = male, 0 = female)
ClassYear	Year in school
LarkOwl	Early riser or night owl or neither
NumEarlyClass	Number classes per week before 9:00 AM
EarlyClass	Indicator for early classes
GPA	GPA
ClassesMissed	Number classes missed
CognitionZscore	Z score for cognitive skills test
PoorSleepQuality	Measure of sleep quality
DepressionScore	Measure of degree of depression
AnxietyScore	Measure of amount of anxiety
StressScore	Measure of amount of stress
DepressionStatus	Depression score
AnxietyStatus	Anxiety score
Stress	Stress score
DASScore	Combined score (depression + anxiety + stress)
Happiness	Measure of degree of happiness

Variable	Description
AlcoholUse	Self reported alcohol use
Drinks	Number alcoholic drinks per week
WeekdayBed	Average weekday bed time (24 = midnight)
WeekdayRise	Average weekday rise time (8.0 = 8:00 AM)
WeekdaySleep	Average hours of sleep on weekdays
WeekendBed	Average weekend bed time (24 = midnight)
WeekendRise	Average weekend rise time (8.0 = 8:00 AM)
WeekendSleep	Average hours of sleep on weekends
AverageSleep	Average hours of sleep for all days
AllNighter	Indicator for all nighter (1 = yes, 0 = no)

How many variables and cases are in the data set `sleep`? The `dim` function gives us the dimensions of a data table (# rows x # columns). The function `head` is also useful to look at the first few rows and check the data.

```
dim(sleep)
```

```
## [1] 253  28
```

```
head(sleep)
```

```
##   Gender ClassYear LarkOwl NumEarlyClass EarlyClass  GPA ClassesMissed
## 1      0         4 Neither             0          0 3.60             0
## 2      0         4 Neither             2          1 3.24             0
## 3      0         4     Owl             0          0 2.97            12
## 4      0         1    Lark             5          1 3.76             0
## 5      0         4     Owl             0          0 3.20             4
## 6      1         4 Neither             0          0 3.50             0
##   CognitionZscore PoorSleepQuality DepressionScore AnxietyScore
## 1           -0.26                4               4            3
## 2            1.39                6               1            0
## 3            0.38               18              18           18
## 4            1.39                9               1            4
## 5            1.22                9               7           25
## 6           -0.04                6              14            8
##   StressScore DepressionStatus AnxietyStatus Stress DASScore Happiness
## 1           8           normal        normal normal       15        28
## 2           3           normal        normal normal        4        25
## 3           9         moderate        severe normal       45        17
## 4           6           normal        normal normal       11        32
## 5          14           normal        severe normal       46        15
## 6          28         moderate      moderate   high       50        22
##   AlcoholUse Drinks WeekdayBed WeekdayRise WeekdaySleep WeekendBed
## 1   Moderate     10      25.75        8.70         7.70      25.75
## 2   Moderate      6      25.70        8.20         6.80      26.00
## 3      Light      3      27.44        6.55         3.00      28.00
## 4      Light      2      23.50        7.17         6.77      27.00
## 5   Moderate      4      25.90        8.67         6.09      23.75
## 6    Abstain      0      23.80        8.95         9.05      26.00
##   WeekendRise WeekendSleep AverageSleep AllNighter     Class
## 1        9.50         5.88         7.18          0    Senior
## 2       10.00         7.25         6.93          0    Senior
## 3       12.59        10.09         5.02          0    Senior
## 4        8.00         7.25         6.90          0 Firstyear
## 5        9.50         7.00         6.35          0    Senior
## 6       10.75         9.00         9.04          0    Senior
```

Statistical Process: Step 2

Identify the Research question(s)

For now we will focus on developing skills to summarize and visualize different types of data. But with such a rich dataset, it would be easy to come up with many different questions (some of which we will come back to and answer later in the course).

- Do students average the recommended 8 hours of sleep?
- Are the number of classes missed related to GPA?
- How is drinking alcohol associated with stress, anxiety, and depression?
- What other research questions can you think of using combinations of the variables above?

We also want to provide the context to the data in this step. One crucial aspect of this is to clearly identify the population our sample is meant to represent. Another is to determine what type of study this data came from.

What is the best population to assume for this sample?[1]

a) All adults
b) All college-aged adults
c) All college students
d) The 253 students at this college

What type of study is this?[2]

a) Observational study
b) Designed experiment

Exploratory Data Analysis (EDA) in `R`

Statistical Process: Step 3

We'll use `R` to develop visualizations and summary statistics to help us explore the sleep data.

> Remember, the goal of EDA is to *identify patterns* in the sample data. It's all about describing what we have already observed. We want to be careful not to assume these patterns exist for everyone in the population.

This tutorial will walk you through the analysis. The code is also available in the `SleepScript.Rmd` file. This contains the `R` code necessary to read in the Sleep Study data and generate summary statistics and visualizations. Visualizations demonstrated are:

- histograms
- side-by-side histograms
- scatterplots
- grouped scatterplots
- box plots

[1] c
[2] a

- side-by-side box plots
- and a few fancier visualizations using the `ggplot2` package.

Packages

One thing we use a lot in R is supplementary *packages* with nice built-in functions for common tasks we'd like to perform. We need to first load these packages before using the functions. The code below does this for the common packages we'll use.

```r
library(openintro)
library(mosaic)
library(dplyr)
library(ggplot2)
library(ggmosaic)
library(readr)
```

These need to be loaded *for each* document you'd like to use the functions for.

EDA for single numerical variable

Summarizing and Visualizing `GPA`

> **Summarize with:** Mean, median, standard deviation, interquartile range, and many more (but these are the main ones)
>
> **Visualize with:** Histograms and boxplots (also not the only options)

To do the EDA in R, we need to use **functions**. Each function we use needs to know:

1. What *variable* you want to use
2. What *data set* that variable comes from

There are two ways to provide this information. Using the `GPA` variable from the `sleep` data set. The first line of code shows the **formula template**. For many functions, we will write:

```
function_name(_________ ~ _________, data = _________)
```

Where the response variable goes on the left of the `~` and the explanatory variable goes on the right. In this instance (the code below), with only one variable, it can go on the right with nothing to the left of the `~`.

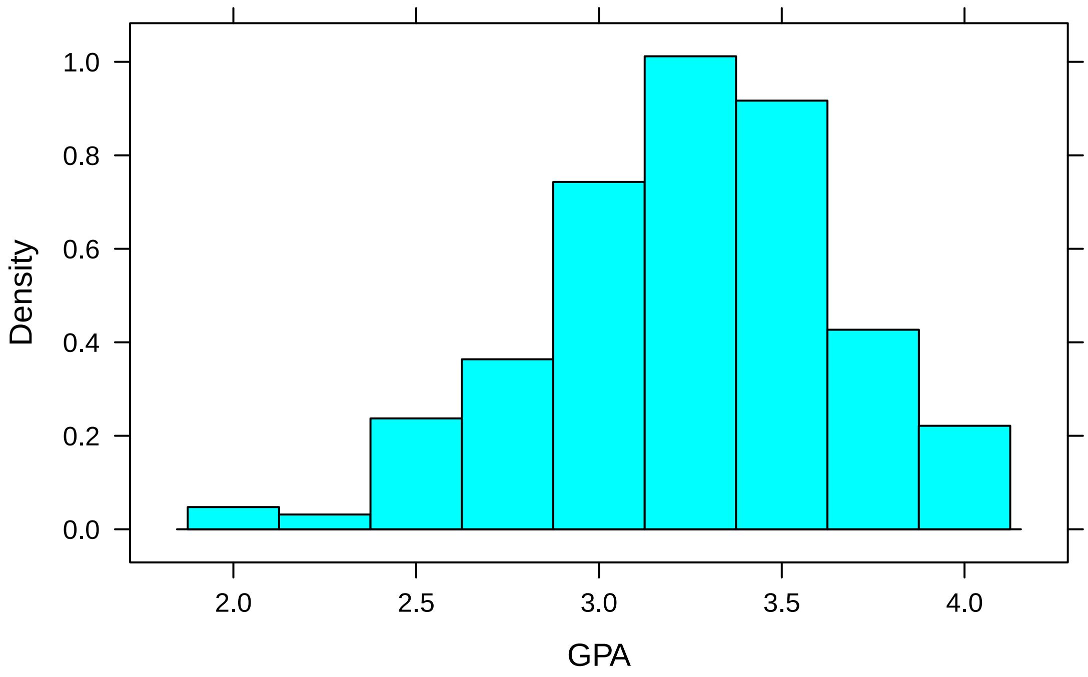
The second line of code uses the same variable, `GPA`, and calculates some common descriptive statistics. In this case the function uses an older format:

```
function_name(_______$_______)
```

Where the data set name is on the left of the `$` and the variable name is on the right.

We can use the `histogram` function to get a histogram and `favstats` to get a bunch of common numeric summaries.

```r
histogram( ~ GPA, data = sleep)
```



```r
favstats(sleep$GPA)
```

```
##  min Q1 median  Q3 max     mean        sd   n missing
##    2  3    3.3 3.5   4 3.243794 0.4042844 253       0
```

Each function is a little different, but don't worry about memorizing which function uses which template. Just understand how it works when you see each type.

EDA for single categorical variable

Summarizing and Visualizing `LarkOwl`

> **Summarize with:** Frequency and proportion tables
>
> **Visualize with:** Bar plots (one categorical variable is usually pretty boring and doesn't need to be visualized in most cases)

A simple **frequency table** that counts the number of responses for each *level* of a categorical variable is a nice way to summarize categorical data. But finding the proportion of cases that belong to each *level* is even more informative.

Only use the `table` functions for categorical variables. Notice the `mytable <-` line, often we need to save data in R to refer to it in later code or analysis. Here, you could use (almost) anything in place of `mytable`.

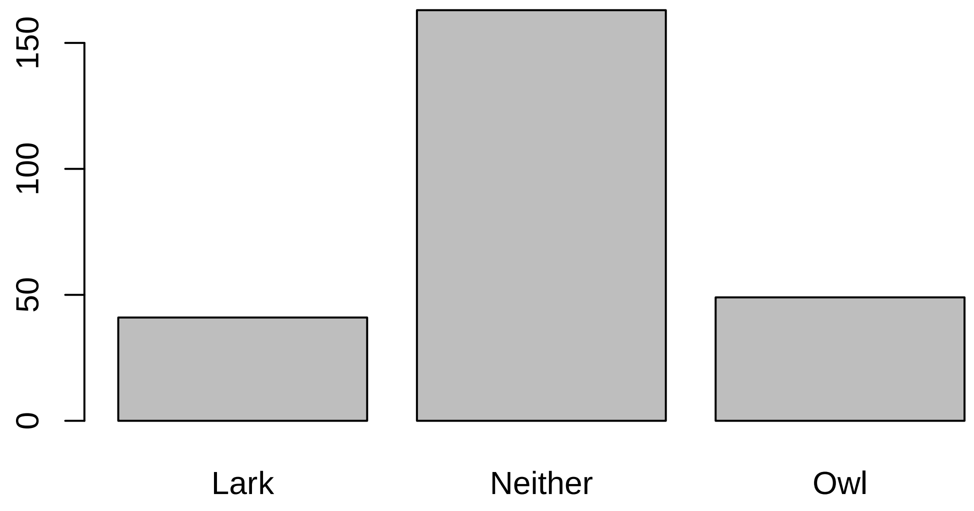
```r
table(sleep$LarkOwl)  # a table can be created directly from the data
```

```
## 
##    Lark Neither     Owl 
##      41     163      49
```

```r
mytable <- table(sleep$LarkOwl)  # a table of proportions needs to saved as a table first
prop.table(mytable)  # then converted to a table of proportions
```

```
## 
##      Lark   Neither       Owl
## 0.1620553 0.6442688 0.1936759
```

```r
barplot(mytable) # somewhat boring...
```



Never use Pie Charts!!!

Here are some articles if you need some convincing:

- Pie Charts Are Bad
- Death to Pie Charts!

But sometimes pie charts are appropriate…

EDA for one numerical and one categorical variable

Summarizing and Visualizing `ClassesMissed` and `LarkOwl`

> **Summarize with:** For each group, find the mean, median, standard deviation, interquartile range
>
> **Visualize with:** Side-by-side or stacked histograms and box plots

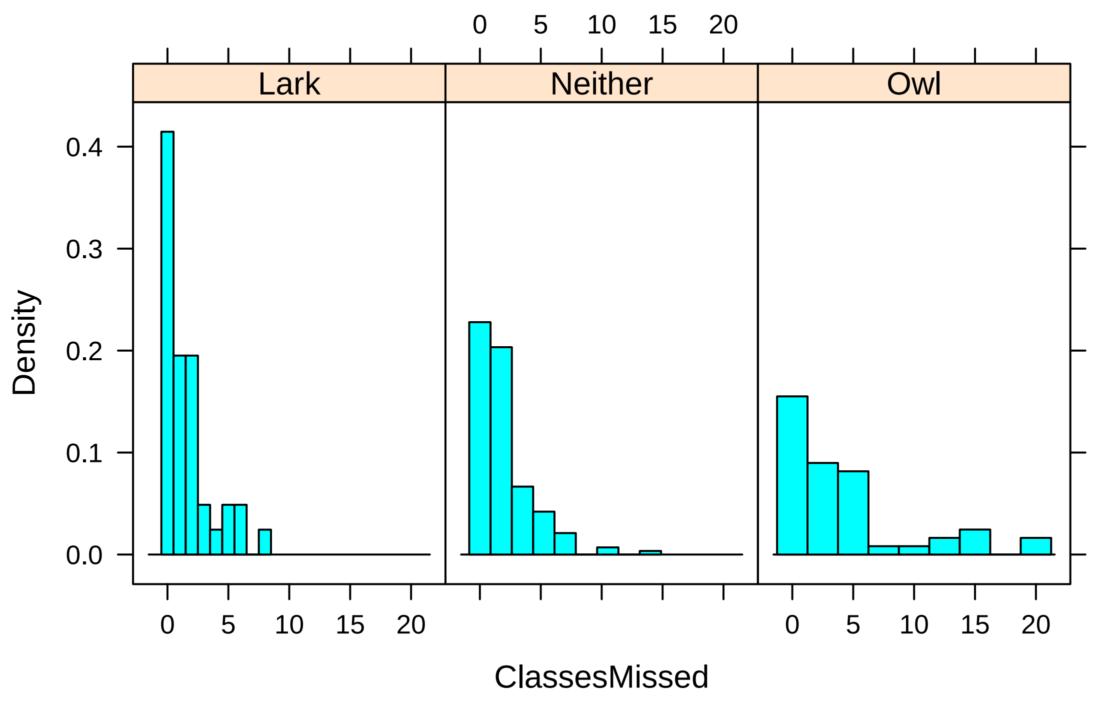
When there are one of each type of variables, we typically summarize the numerical one and separate it into the different levels of the categorical variable.

Be careful with the code here, notice that `histogram` and `favstats` split the different levels of `LarkOwl` with a `|`. While the box and whisker plot, `bwplot`, just separates them by the `~`. It can be tricky sometimes, but just try to follow these examples provided when handling other data sets.

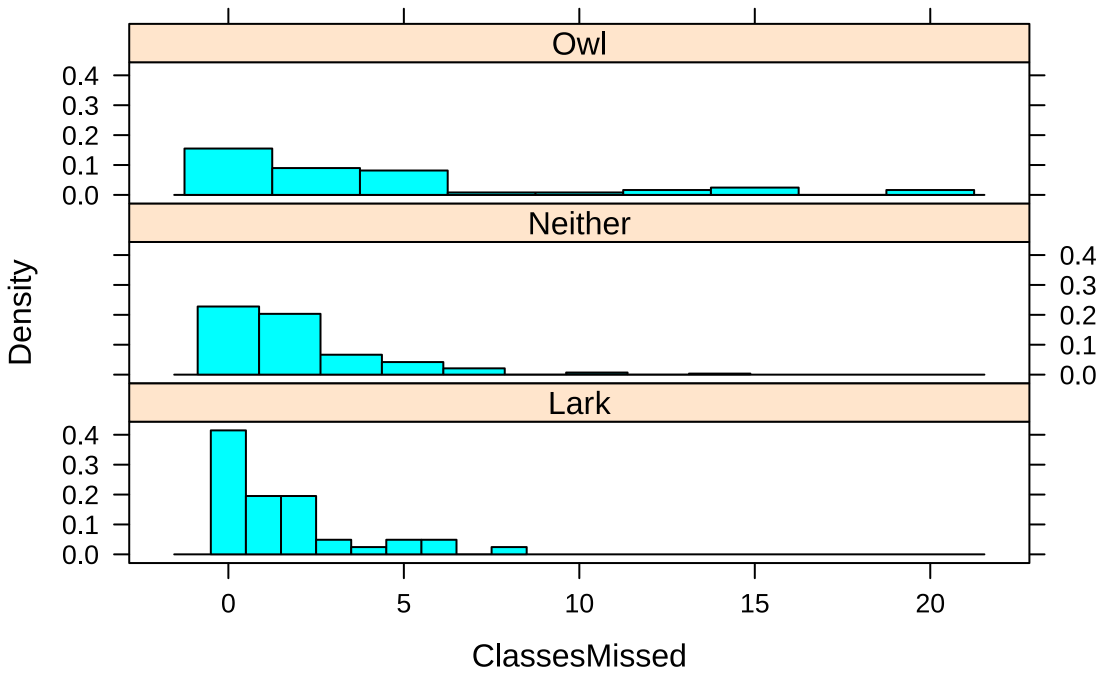
```r
favstats(~ ClassesMissed | LarkOwl, data = sleep)
```

```
##   LarkOwl min Q1 median Q3 max     mean       sd   n missing
## 1    Lark   0  0      1  2   8 1.560976 2.000610  41       0
## 2 Neither   0  0      1  2  14 1.773006 2.357810 163       0
## 3     Owl   0  1      2  5  20 4.204082 5.287484  49       0
```

```r
histogram(~ ClassesMissed | LarkOwl, data = sleep)
```

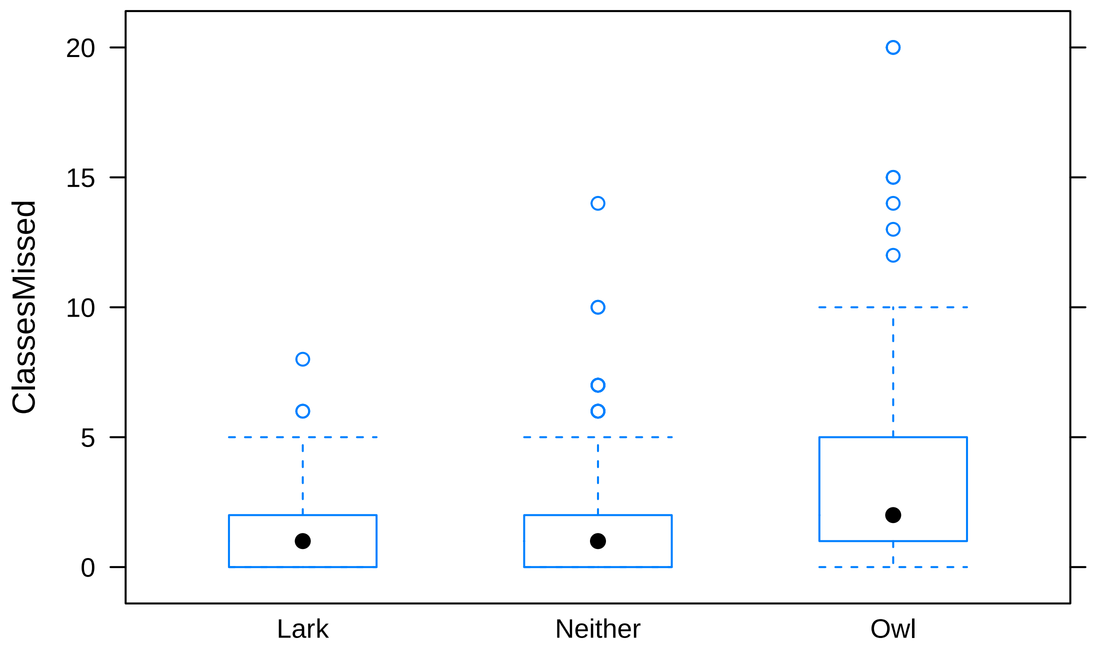


```r
histogram(~ ClassesMissed | LarkOwl, layout = c(1,3), data = sleep) # layout changes the number of rows
```

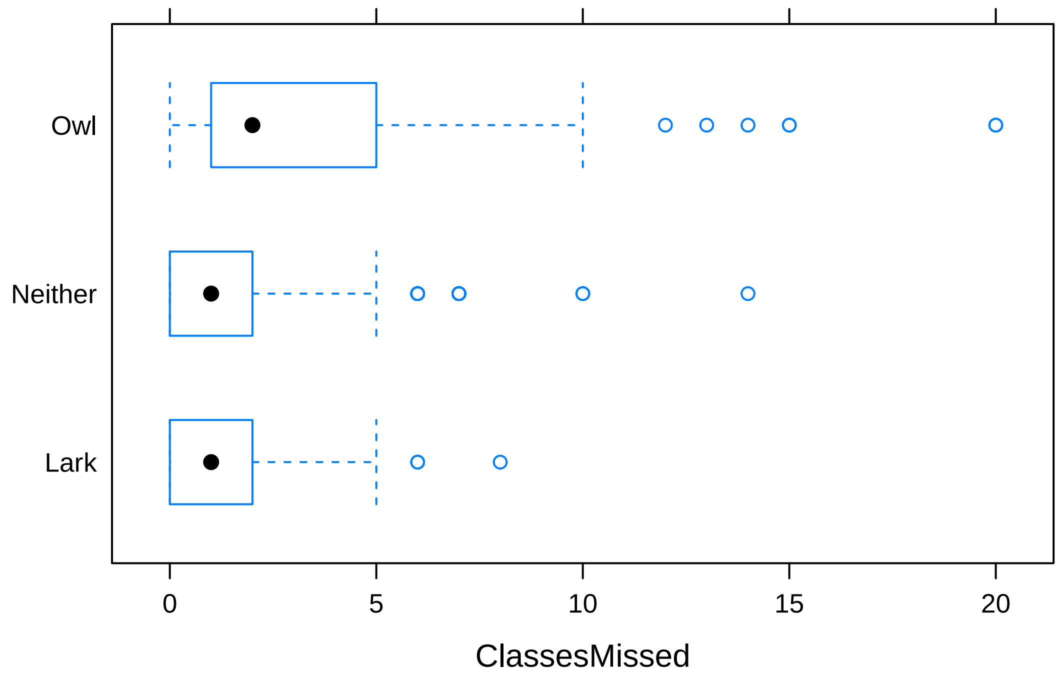


```r
  # stacked layout allows for better comparison
```

```r
bwplot(ClassesMissed ~ LarkOwl, data = sleep)
```



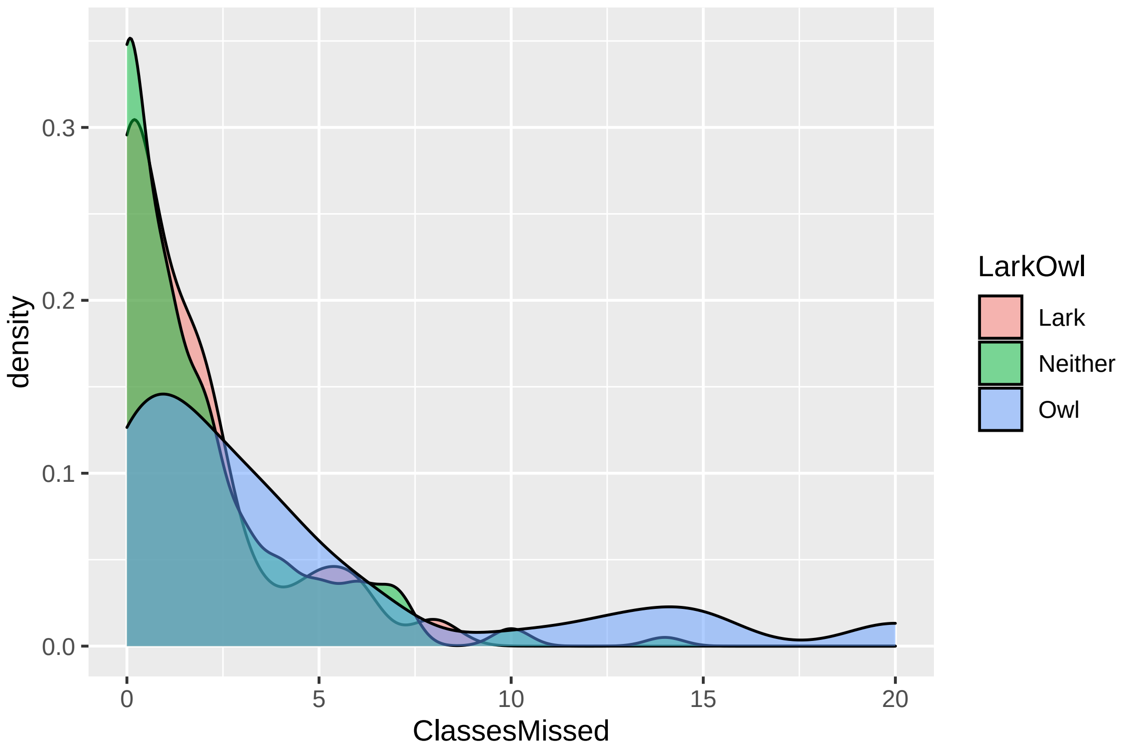
```
bwplot(LarkOwl ~ ClassesMissed, data = sleep)
```



```
    # LarkOwl on y-axis allows for easier comparisons
```

This is a slightly more advanced way to plot things. Not necessary, but a nicer way to display the relationship.

```
ggplot(sleep, aes(x = ClassesMissed, fill = LarkOwl)) + geom_density(alpha=0.5)
```



```
  # above and beyond - if you want to get fancy with overlayed density plots
```

EDA for two numerical variables

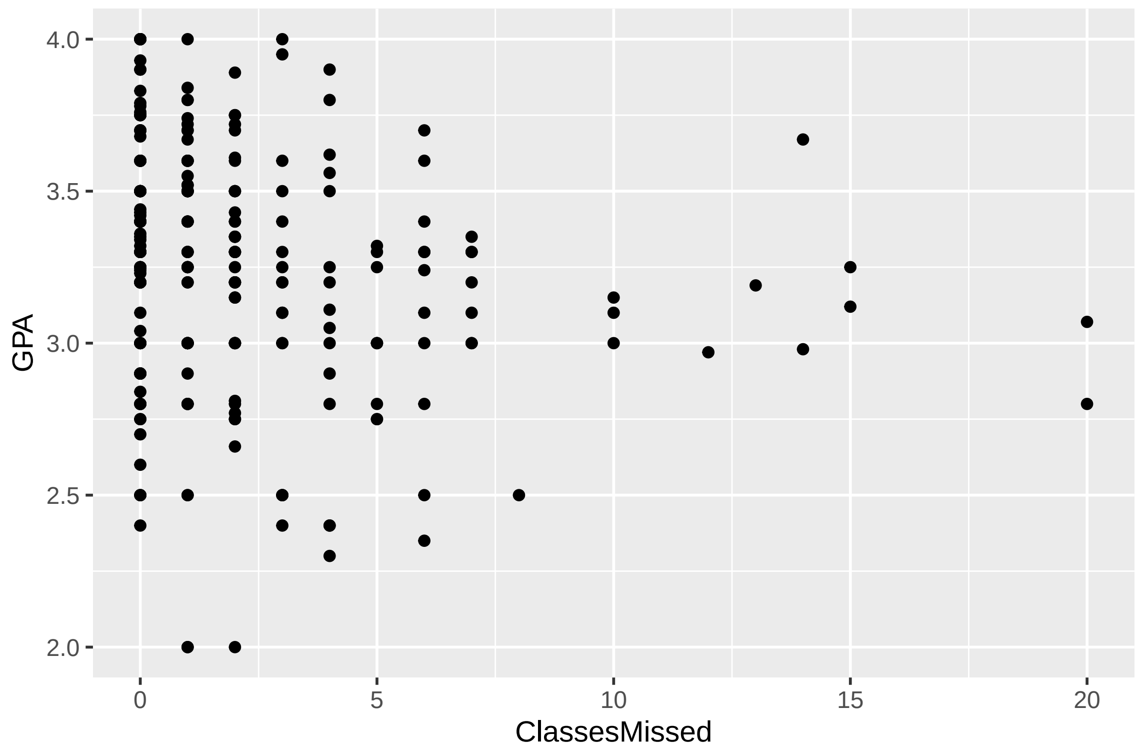
Summarizing and Visualizing `ClassesMissed` and `GPA`

Summarize with: Correlation coefficient (covered in more detail in Ch. 5)

Visualize with: Scatterplot

Scatterplots can display two numerical variables. And one way to summarize them is with a line that describes the pattern (more on this in Ch. 5). **Correlation** is also a nice numerical summary for these types of variables.

```
gf_point(GPA ~ ClassesMissed, data = sleep)    # This plots the points
```



```
cor(sleep$GPA, sleep$ClassesMissed)
```

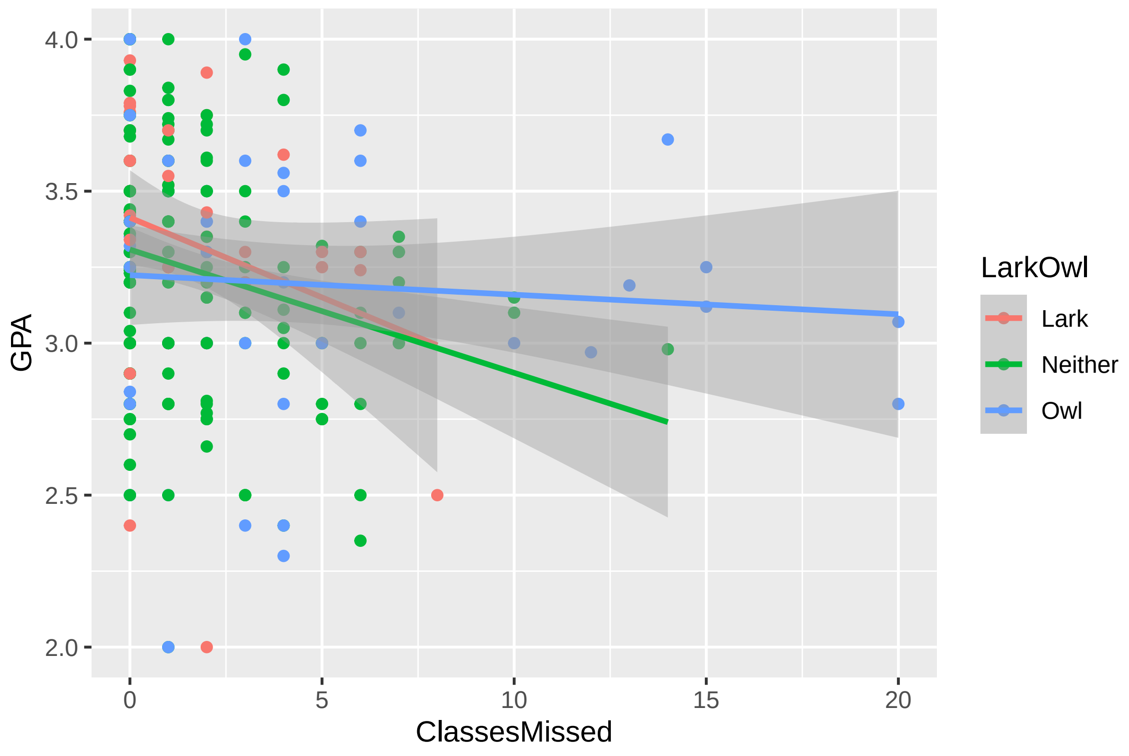
```
## [1] -0.1815426
```

```
  # find correlation coefficient between two variables
```

Again, you can add certain features to visualize better. (This actually is two numerical variables and a third categorical variable)

```
ggplot(sleep, aes(y = GPA, ClassesMissed, color = LarkOwl)) +
  geom_point() +
  geom_smooth(method = lm)
```

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



```
  # above and beyond - if you want to include third variable
```

EDA for two categorical variables

Summarizing and Visualizing `ClassYear` and `LarkOwl`

Summarize with: Two-way tables of frequencies and proportions

Visualize with: Segmented bar plots and mosaic plots

Two-way, or **Contingency tables** can organize two categorical variables into how frequently they occur. The `labels` part of the code below changes the variable values from numbers, `1`, `2`, `3`, `4` to names like `Firstyear`...

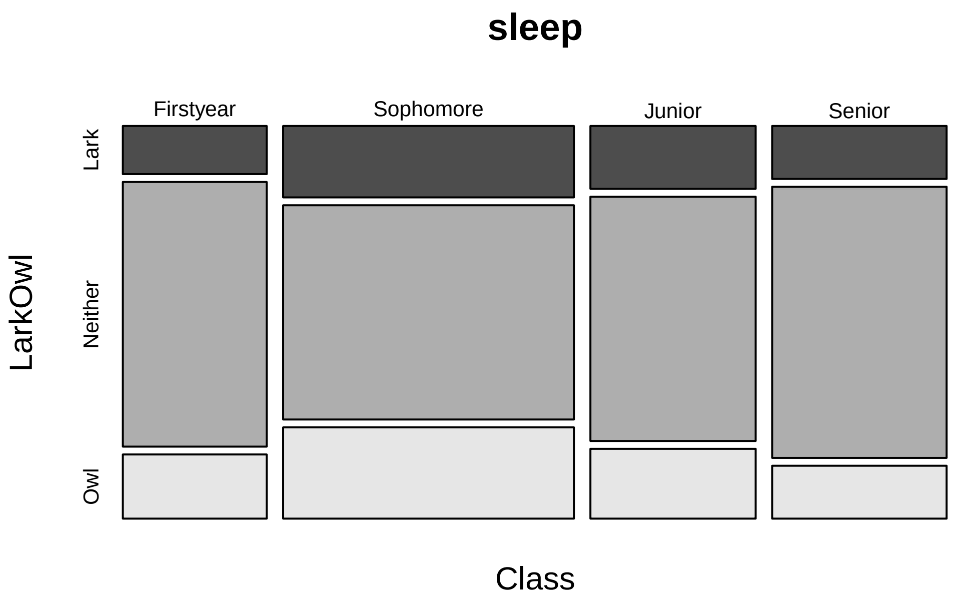
```
sleep$Class = factor(sleep$ClassYear,
  labels=c("Firstyear","Sophomore","Junior","Senior"))
  # create new variable Class that uses categories and not numbers

table(sleep$LarkOwl, sleep$Class)
```

```
##
##            Firstyear Sophomore Junior Senior
##   Lark             6        18      9      8
##   Neither         33        54     35     41
##   Owl              8        23     10      8
```

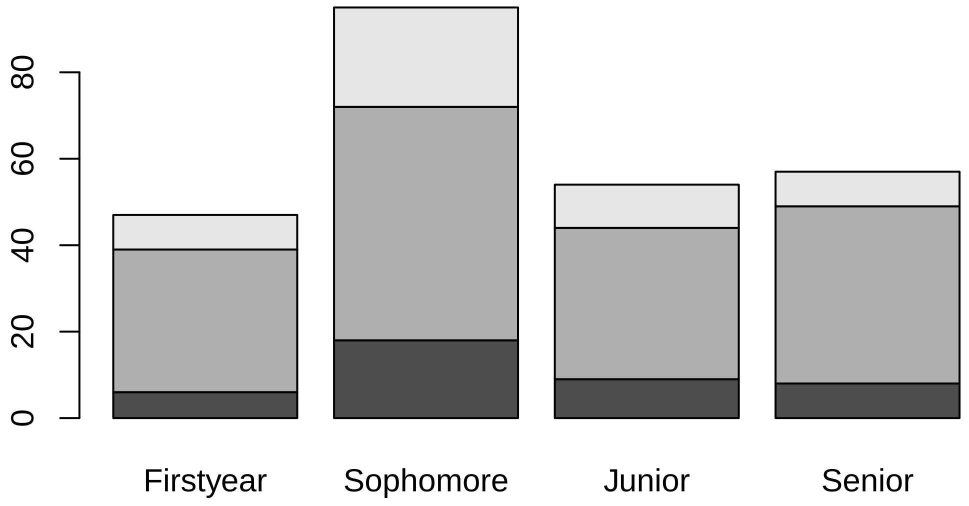
```
mytable <- table(sleep$LarkOwl, sleep$Class) # save the table to use later
```

Usually the level counts don't mean much unless we know what the total count is. We can often get better information by looking at proportions. `prop.table` turns a table of counts into a table of proportions. We can then use **bar plots** and **mosaic plots** to visualize the data and any patterns that might be present.

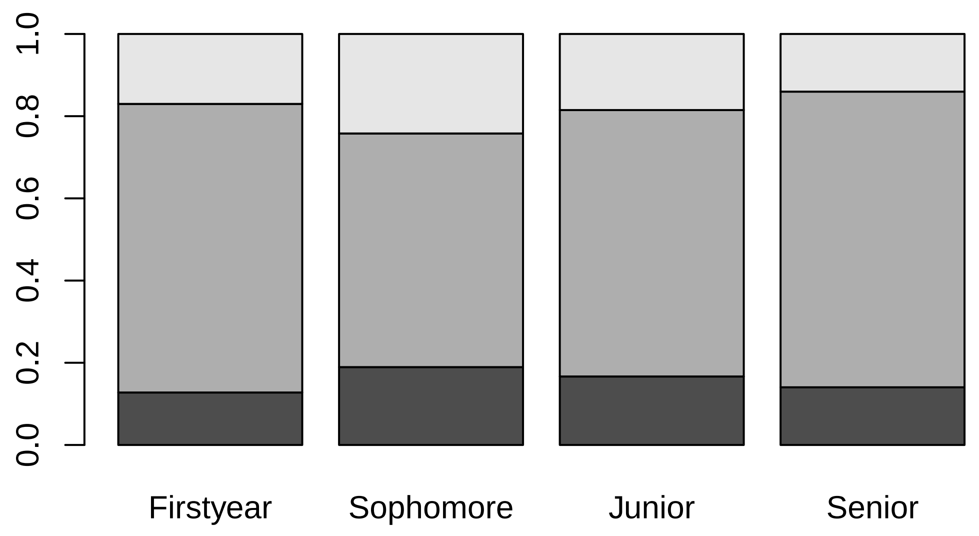
```r
mosaicplot(~ Class + LarkOwl, data = sleep, color = TRUE)
```



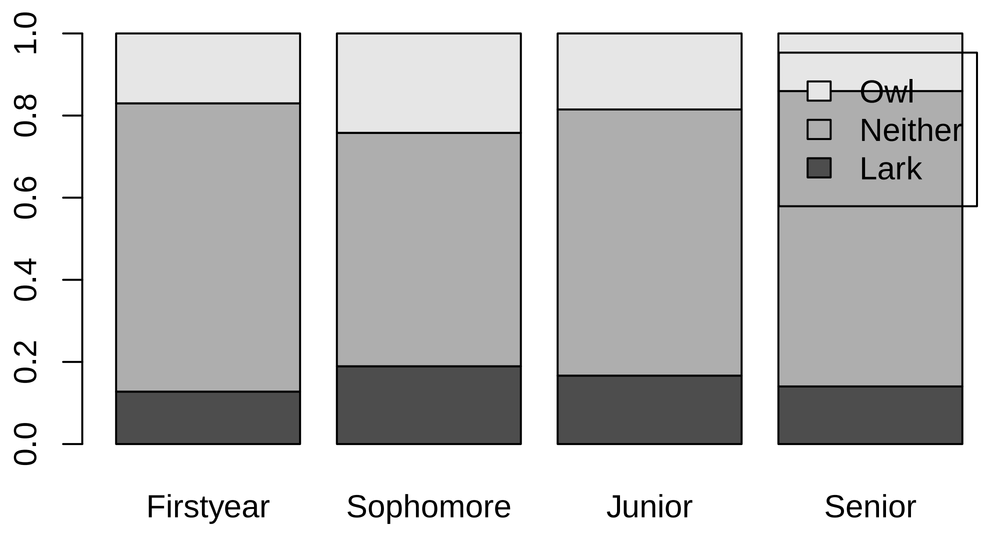
```r
barplot(mytable)
```



```r
barplot(prop.table(mytable, margin = 2))
```



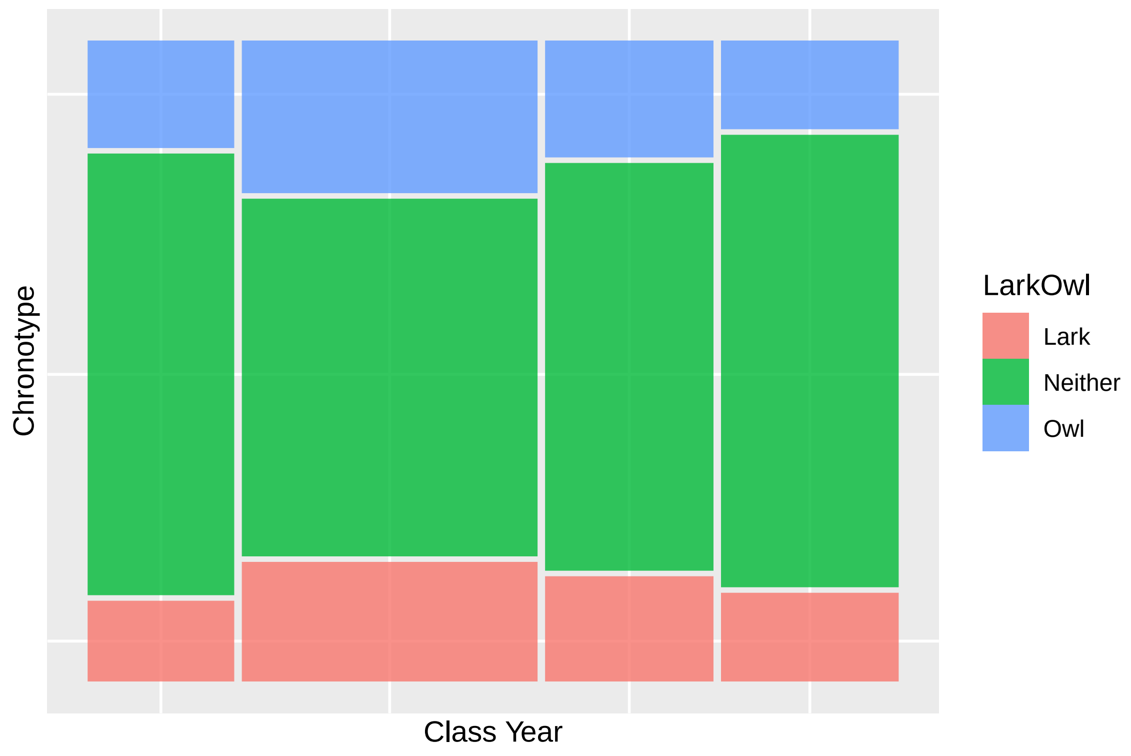
```r
barplot(prop.table(mytable, margin = 2), legend = TRUE)
```



```r
  # 3 options for plotting relationship - compare and contrast
```

The fancy-pants version…

```r
ggplot(data=sleep)+
    geom_mosaic(aes(x=product(Class), fill=LarkOwl))+
    labs(x="Class Year",y="Chronotype")
```



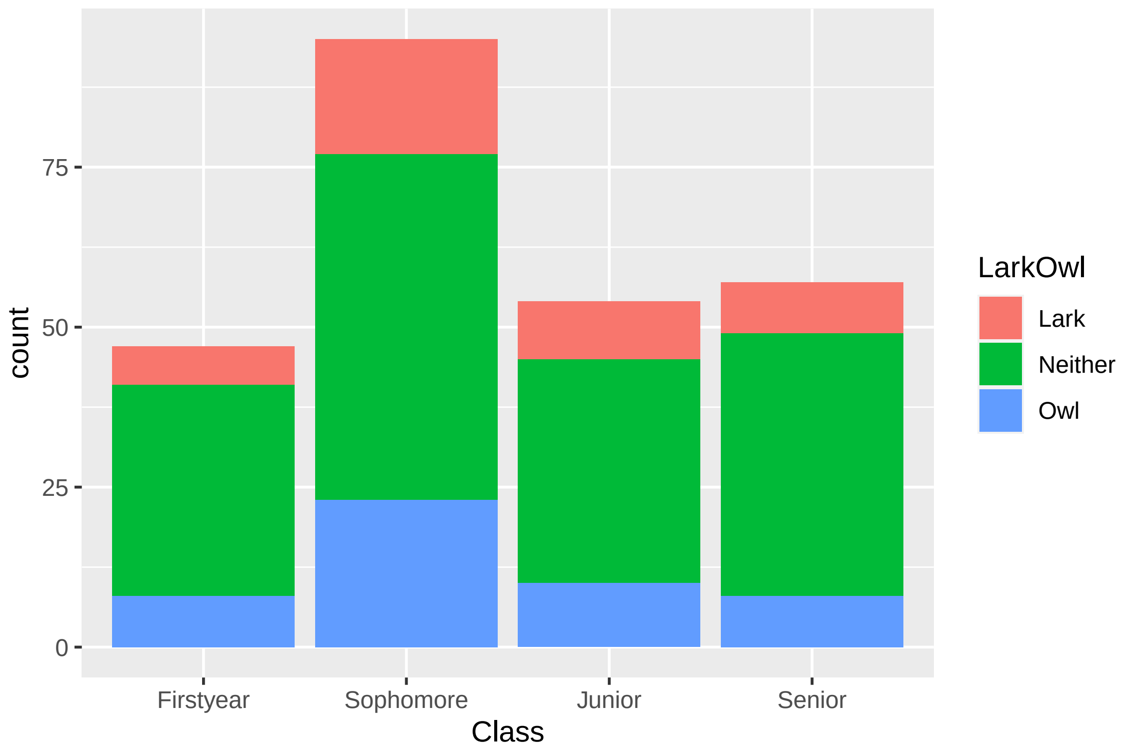
```
prop.table(mytable, 1)
```

```
##
##            Firstyear Sophomore    Junior    Senior
##    Lark    0.1463415 0.4390244 0.2195122 0.1951220
##    Neither 0.2024540 0.3312883 0.2147239 0.2515337
##    Owl     0.1632653 0.4693878 0.2040816 0.1632653
```

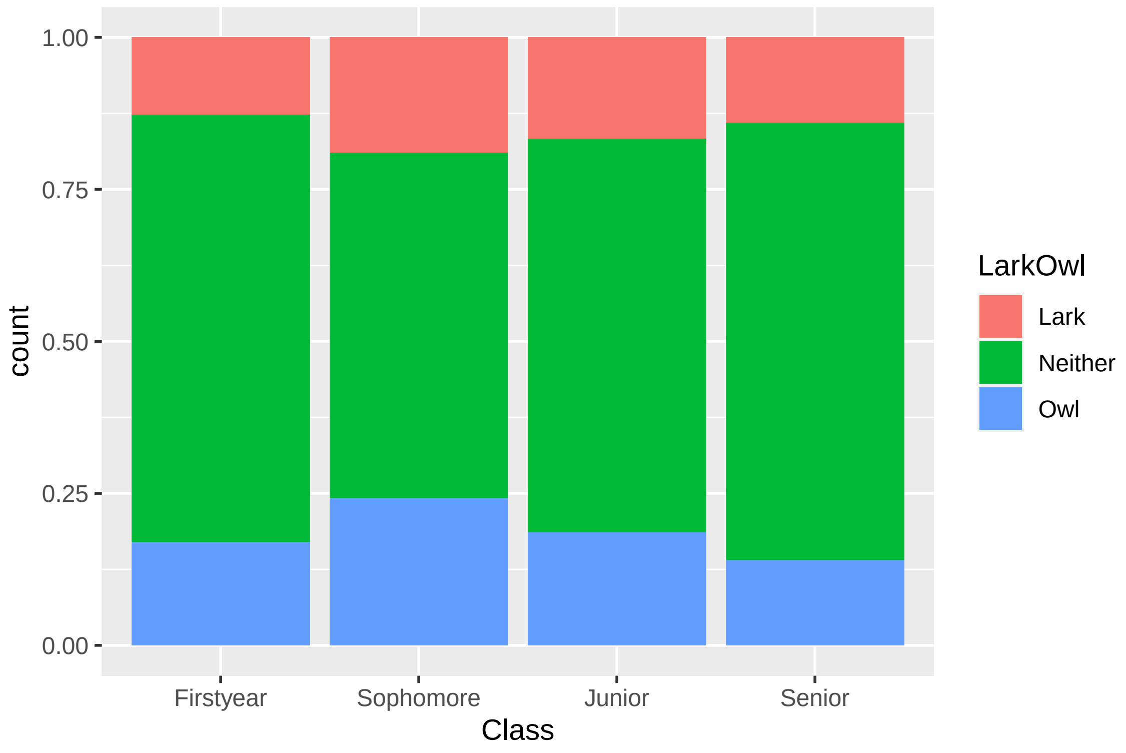
```
  # proportions by rows
prop.table(mytable, 2)
```

```
##
##            Firstyear Sophomore    Junior    Senior
##    Lark    0.1276596 0.1894737 0.1666667 0.1403509
##    Neither 0.7021277 0.5684211 0.6481481 0.7192982
##    Owl     0.1702128 0.2421053 0.1851852 0.1403509
```

```
  # proportions by columns

ggplot(data = sleep) +
  geom_bar(mapping = aes(x = Class, fill = LarkOwl))
```



```r
ggplot(data = sleep) +
  geom_bar(mapping = aes(x = Class, fill = LarkOwl), position = "fill")
```



```r
  # above and beyond - a bit fancier than barplot
```

Extra

A lot of the material below we will get to in due time. But if you enjoyed the taste of R coding above, here are some extras for handling data. If what you did above was enough, you can skip this section for now.

Data Wrangling Hints

The art of cleaning data and preparing it to work with is called *Data Wrangling*. Here is some code that can help with making your data easier to deal with. As well as more code to make plots more informative and presentable. This code is also available as an R script in the server. We will have more tutorials in the future for data wrangling.

```r
# Load in data
sleep <- read.csv("~/Stats 212b S20/Class/Data/SleepStudy.csv")
summary(sleep)

# Select only a few variables to keep in a dataset
sleep2 <- sleep %>% select(Gender, ClassYear, AverageSleep, AllNighter)


# Change AllNighter from 0/1 to No/Yes
sleep$AllNighter = factor(sleep$AllNighter, labels=c("No", "Yes"))
# check that code worked
table(sleep$AllNighter)


# Create new variable Class which is categorical class year
sleep$Class = factor(sleep$ClassYear,
                     labels=c("Firstyear","Sophomore","Junior","Senior"))
# check that code worked
head(sleep %>% select(Class, ClassYear))
```

Filtering

Filtering out some of the cases is another vital ability for someone dealing with data. Often we only want to consider a subset of the entire group.

```r
######## Two ways to keep only students who are Larks or Owls
newdata <- sleep %>% filter(LarkOwl == "Lark" | LarkOwl == "Owl")
newdata <- sleep %>% filter(LarkOwl != "Neither")
```

Changing Categorical Levels

Combining groups in a variable can help to simplify things.

```r
#### Recode depression and alcohol use into 2 levels each; two different methods

# method 1: using recode()
sleep <- sleep %>%
  mutate(depress2 = recode(DepressionStatus, moderate = "high",
                           severe = "high", normal = "normal"),
         alcohol2 = recode(AlcoholUse, Abstain = "low",
                           Light = "low", Moderate = "high",
                           Heavy = "high"))

# method 2: using if_else()
```

```
sleep <- sleep %>%
            mutate( depress2b = if_else( DepressionStatus %in% c("moderate","severe"),"high","low"  ),
                    alcohol2b = if_else( AlcoholUse %in% c("Moderate","Heavy"),"high","low")
                    )


# check that code worked
table(sleep$AlcoholUse)
table(sleep$alcohol2)
table(sleep$alcohol2b)

# Reorder the categories in AnxietyStatus so plots make more sense than simply
#   alphabetical - normal then moderate then severe
histogram(~ PoorSleepQuality | AnxietyStatus, layout = c(1,3), data = sleep)
sleep$AnxietyStatus <- factor(sleep$AnxietyStatus,
                              levels = c("normal", "moderate", "severe"))
histogram(~ PoorSleepQuality | AnxietyStatus, layout = c(1,3), data = sleep)

# Create a categorical variable by binning a numeric variable
newdata <- sleep %>%
  mutate(MissYN = cut(ClassesMissed, breaks=c(-Inf, 0, Inf),
                      labels=c("No misses","Missed at least one")))
newdata <- sleep %>%
  mutate(RiseGroups = cut(WeekdayRise, breaks=c(-Inf, 8, 9.01, Inf),
                     labels=c("Before 8:00","8:00-9:00", "After 9:00")))
# check that code worked
head(newdata %>% select(WeekdayRise, RiseGroups))
```

Plotting Extras

```
# Boxplot with third variable
sleep$Gender <- factor(sleep$Gender, labels = c("Female", "Male"))
bwplot(PoorSleepQuality ~ AnxietyStatus | Gender, data = sleep)

# coded scatterplot with points sized by fourth variable
ggplot(sleep, aes(y = GPA, ClassesMissed, color = LarkOwl)) +
  geom_point(aes(size = NumEarlyClass)) +
  geom_smooth(method = lm)

# label plots
bwplot(PoorSleepQuality ~ AnxietyStatus, data = sleep, xlab = "Anxiety Status",
       main = "Figure 1")

plot(GPA ~ ClassesMissed, data = sleep, xlab = "Number of missed classes",
     ylab = "Grade point average", main = "Figure 2")
fitline <- lm(GPA ~ ClassesMissed, data = sleep)
abline(fitline)

ggplot(sleep, aes(y = GPA, ClassesMissed, color = LarkOwl)) +
  geom_point() + geom_smooth(method = lm) +
  labs(x = "Number of missed classes", y = "Grade point average",
       title = "Figure 3", color = "Sleep preference")
```

Play around

Feel free to open the `SleepScript.Rmd` file and play around with it on the server. Change variables and produce different plots!