Week 5.2 Visualizing Amounts

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.0.3
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

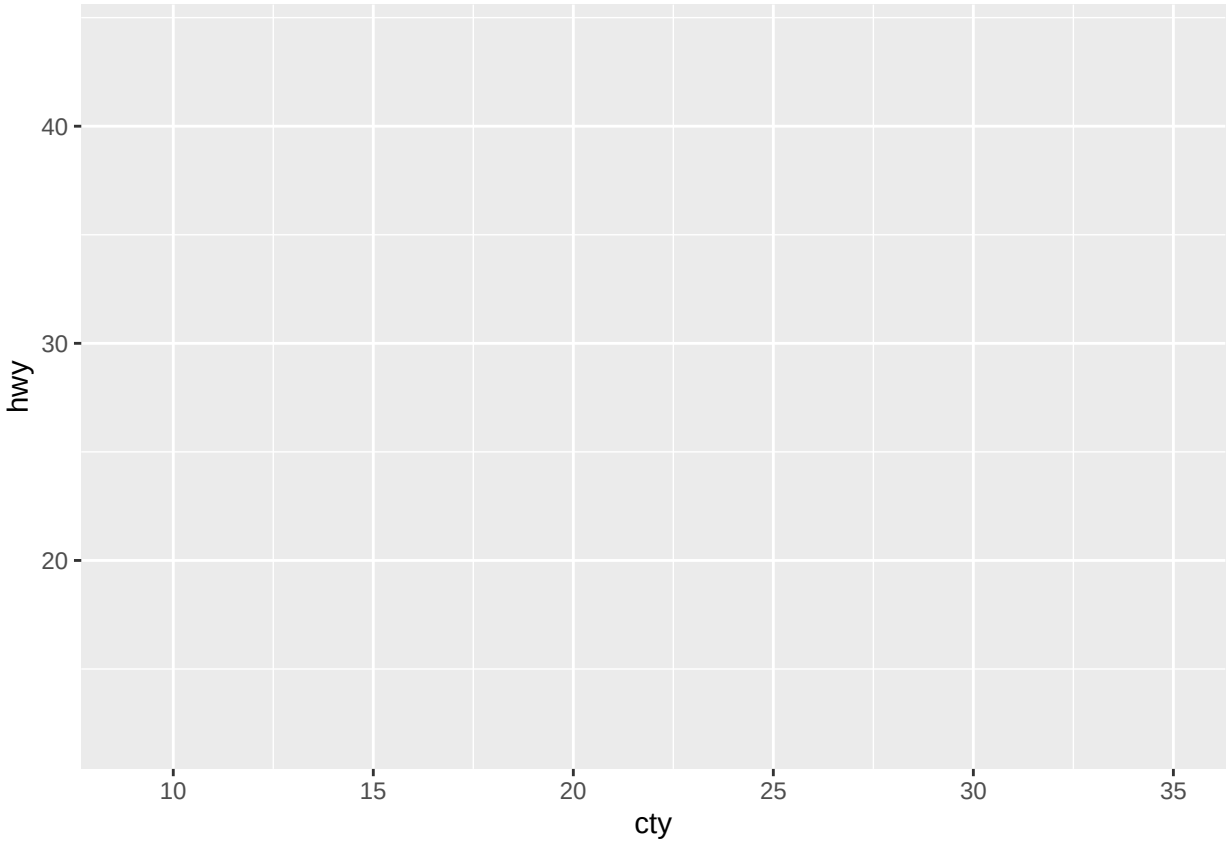
```
library (magrittr)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

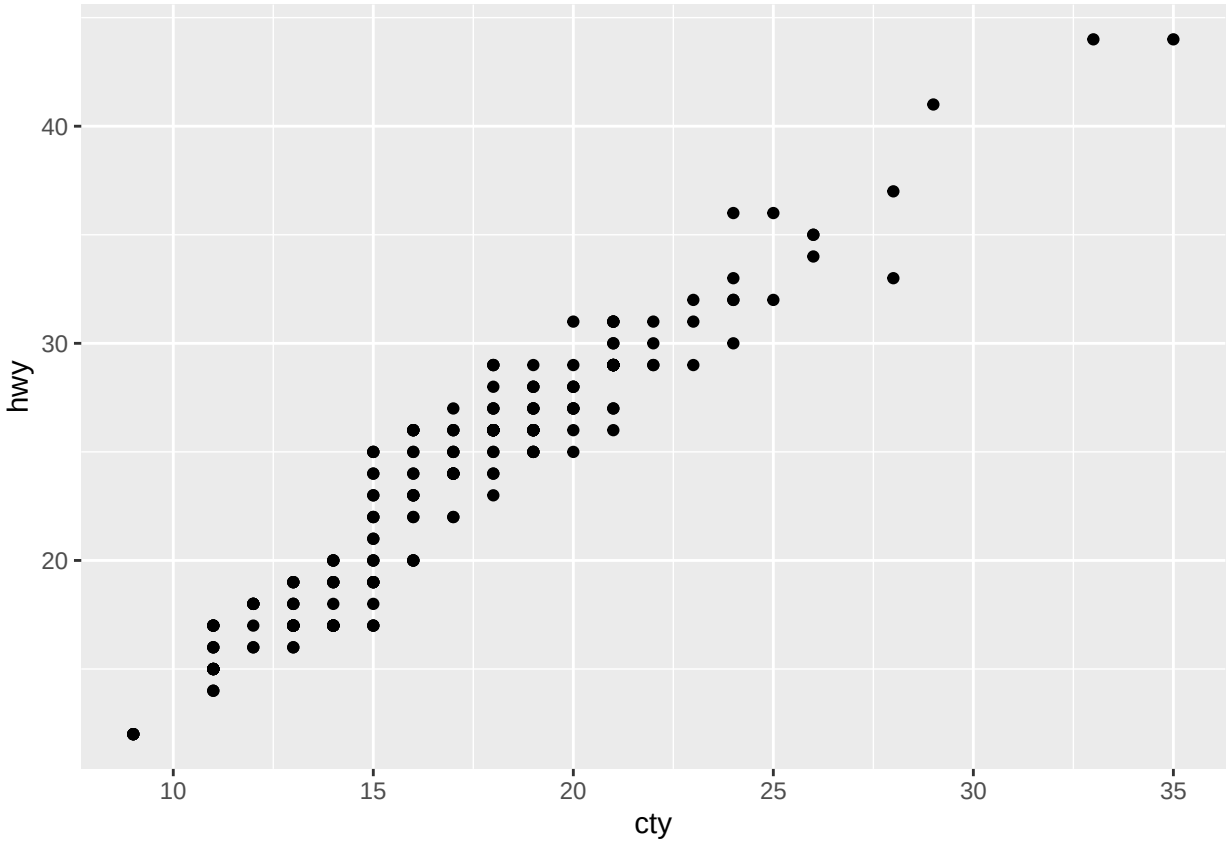
```
## The following object is masked from 'package:magrittr':
##
##     extract
```

```
# Use ggplot function to plot
# Using only ggplot function will create a blank canvas
data<-mpg
ggplot(mpg)
```

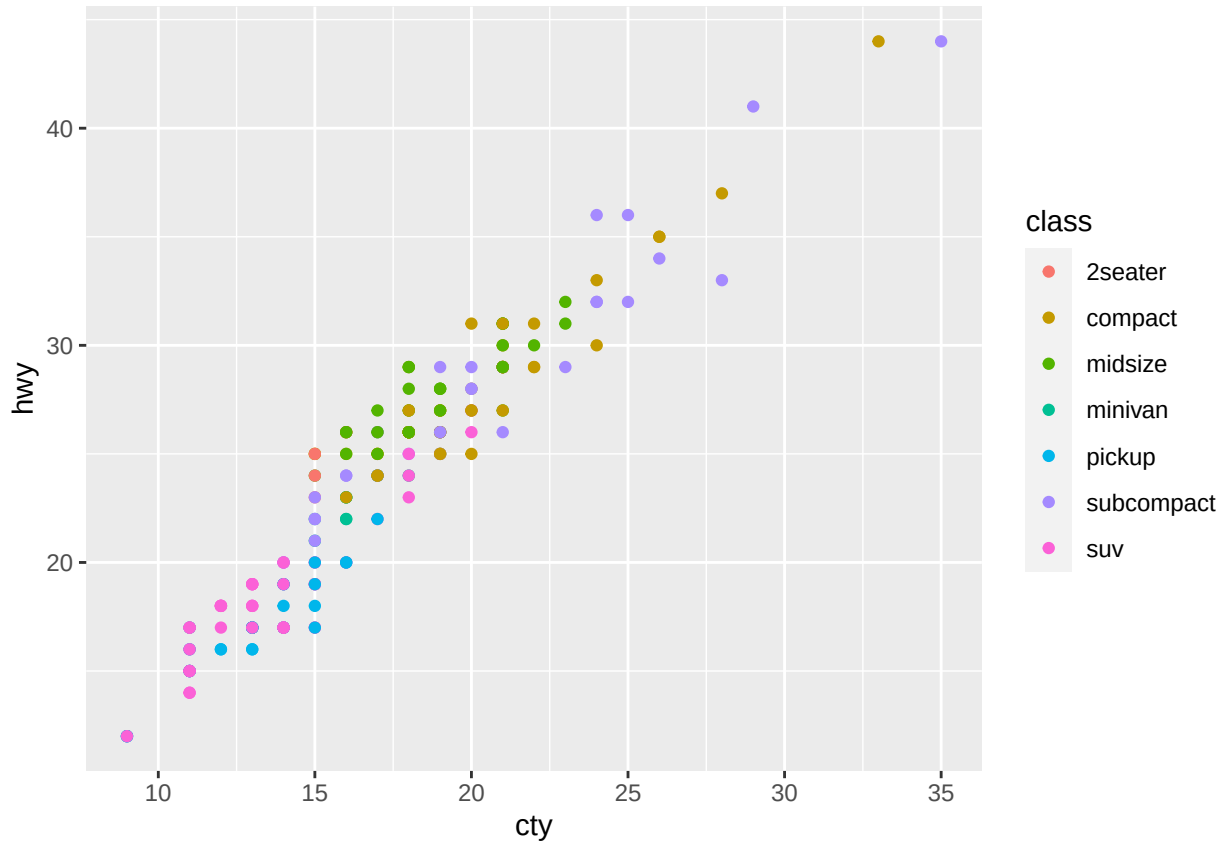
```
# Map variables to aesthetics
ggplot(mpg, aes(x=cty,y=hwy))
```



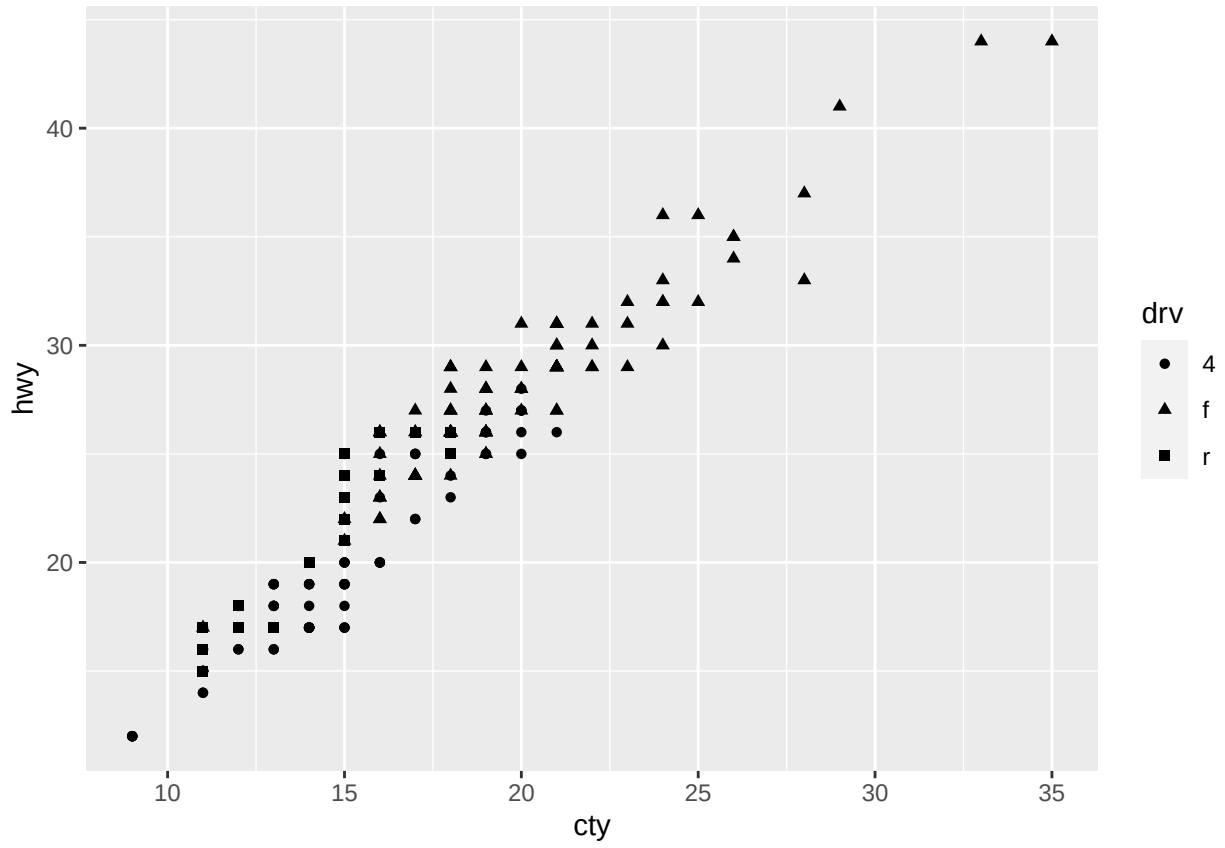

```
# Generate canvas + add axis + plot data
ggplot(mpg, aes(x=cty,y=hwy))+geom_point()
```



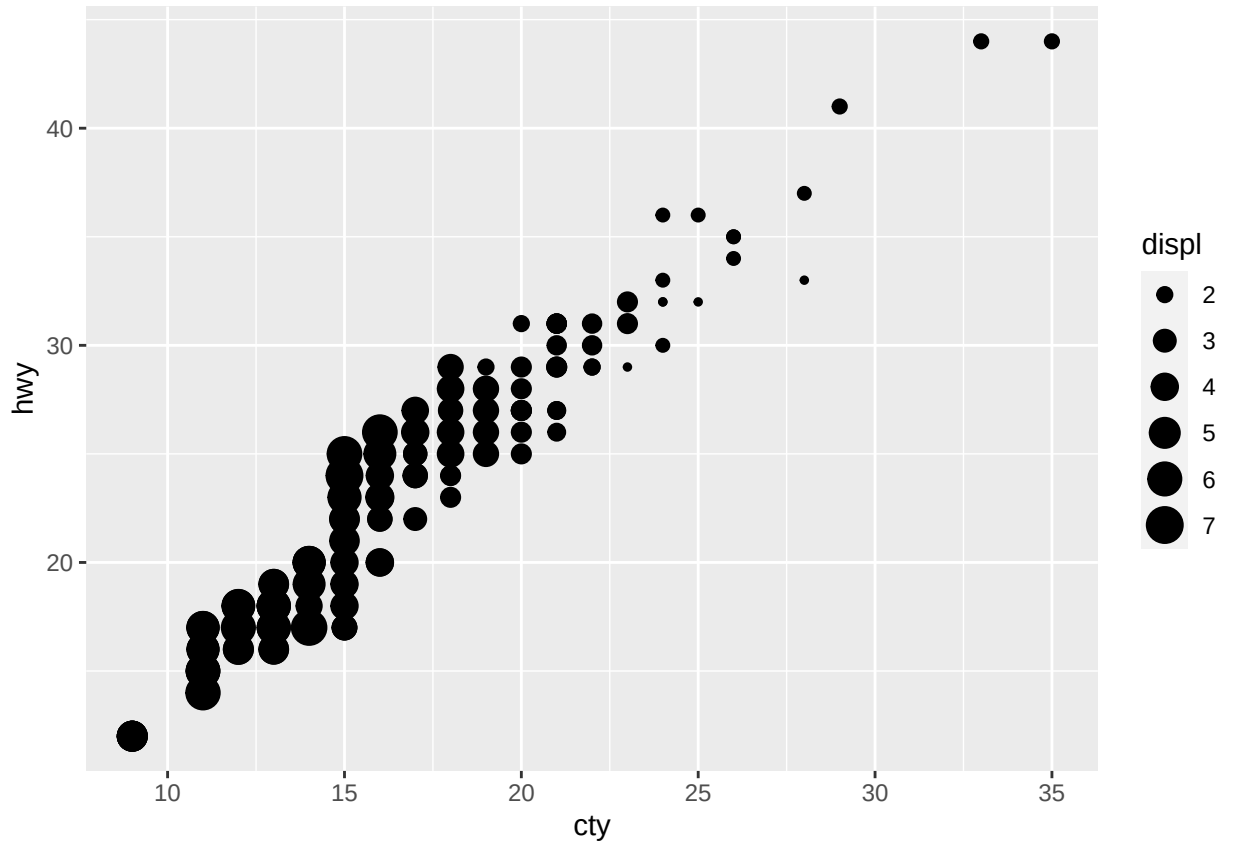

```
# Map additional dimension to aesthetics. In this case color
ggplot(mpg, aes(x=cty,y=hwy, color=class))+geom_point()
```



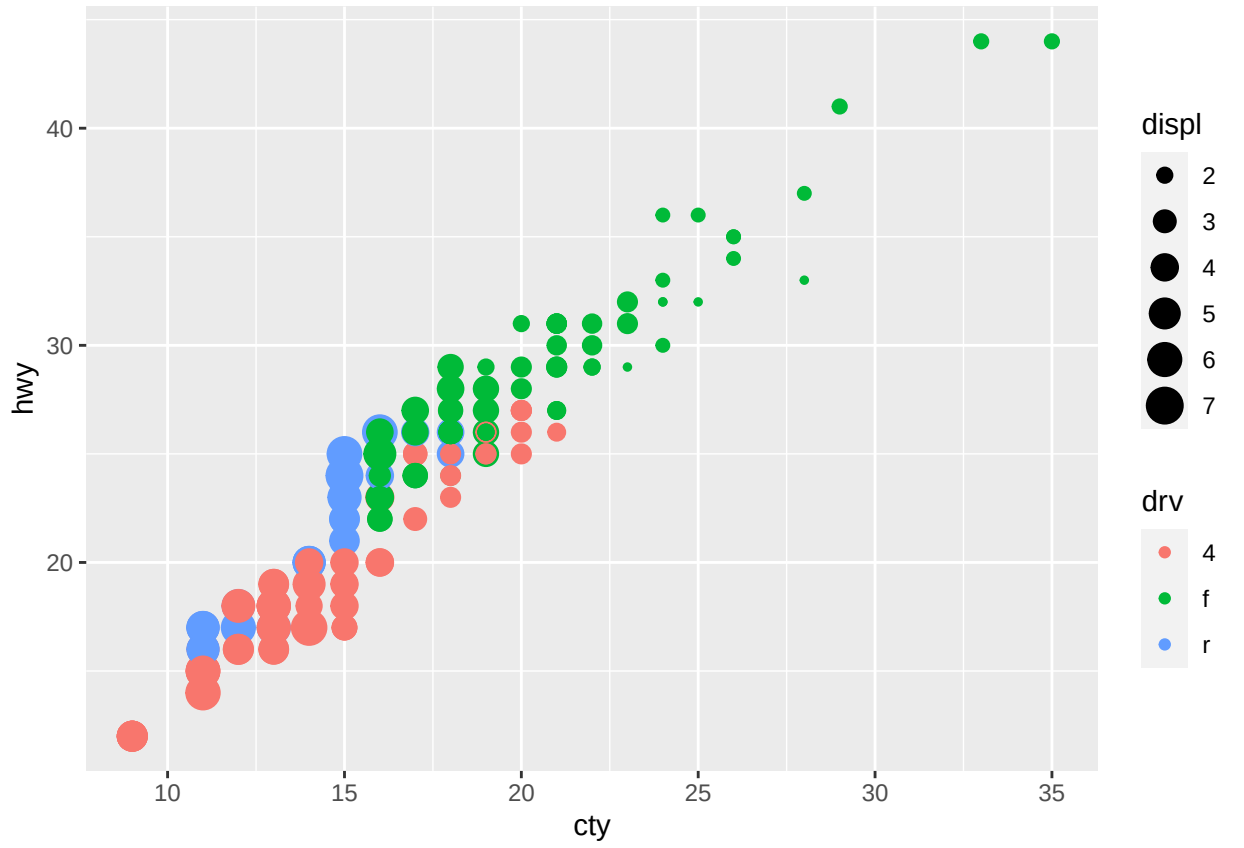

```
# Map dimension as shape
ggplot(mpg, aes(x=cty,y=hwy, shape=drv))+geom_point()
```




```
# Map dimension as size
ggplot(mpg, aes(x=cty,y=hwy, size=displ))+geom_point()
```

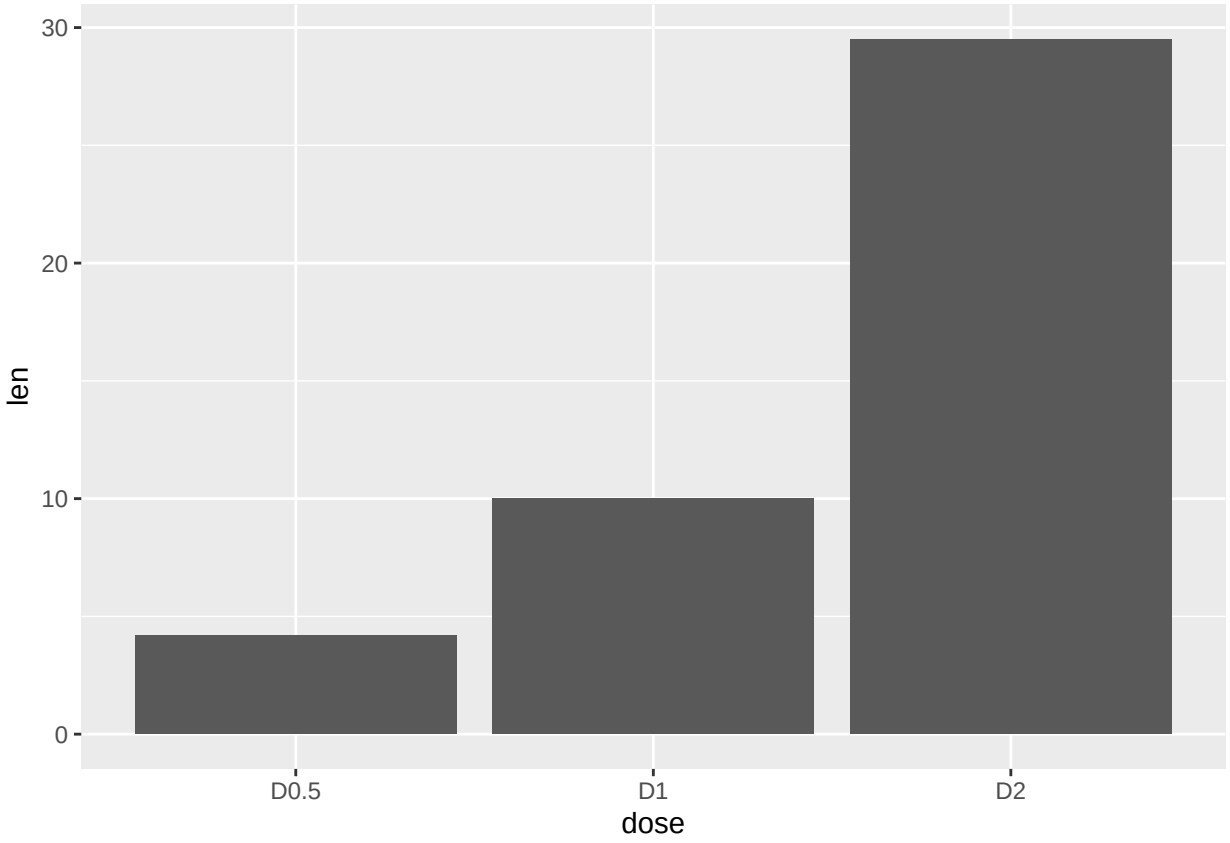



```
# Mapping multi dimensions
ggplot(mpg, aes(x=cty,y=hwy, size=displ, color=drv))+geom_point()
```

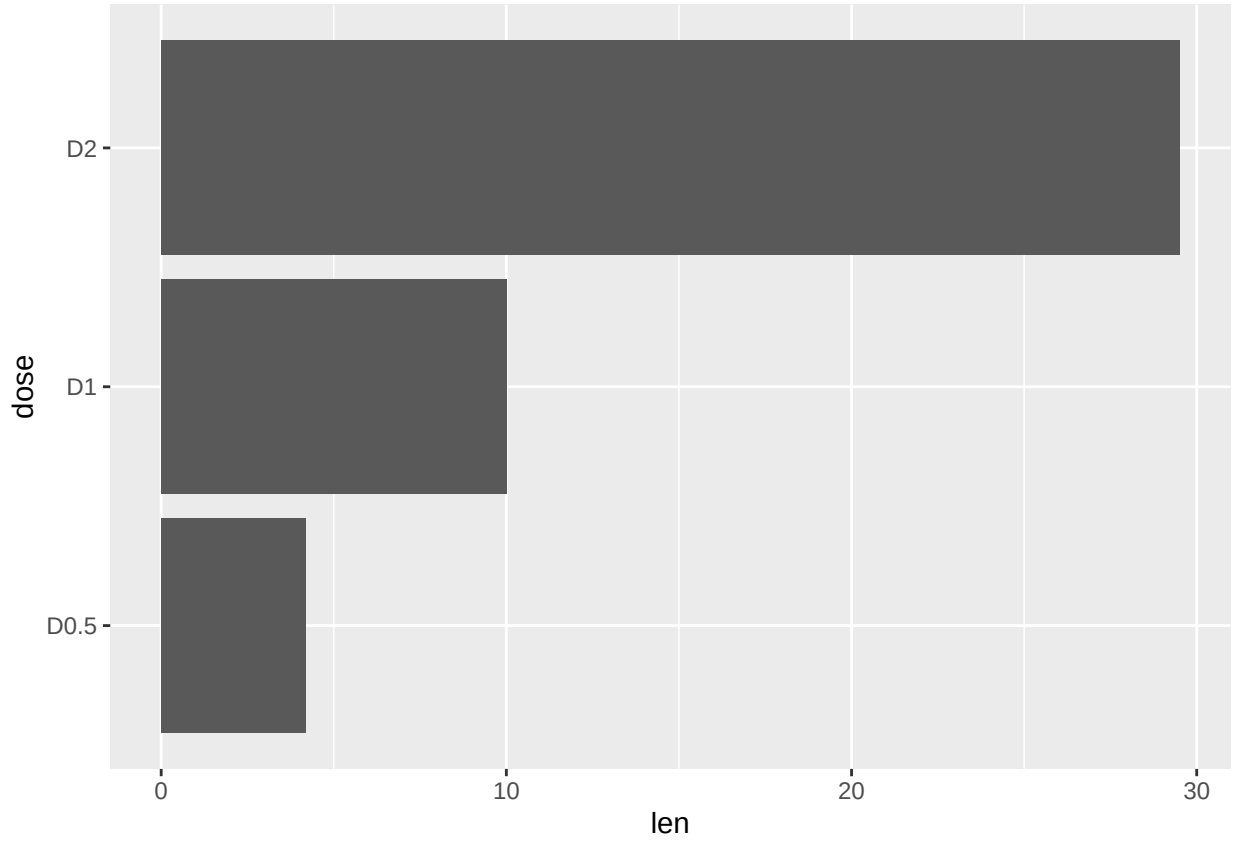


Bar Chart

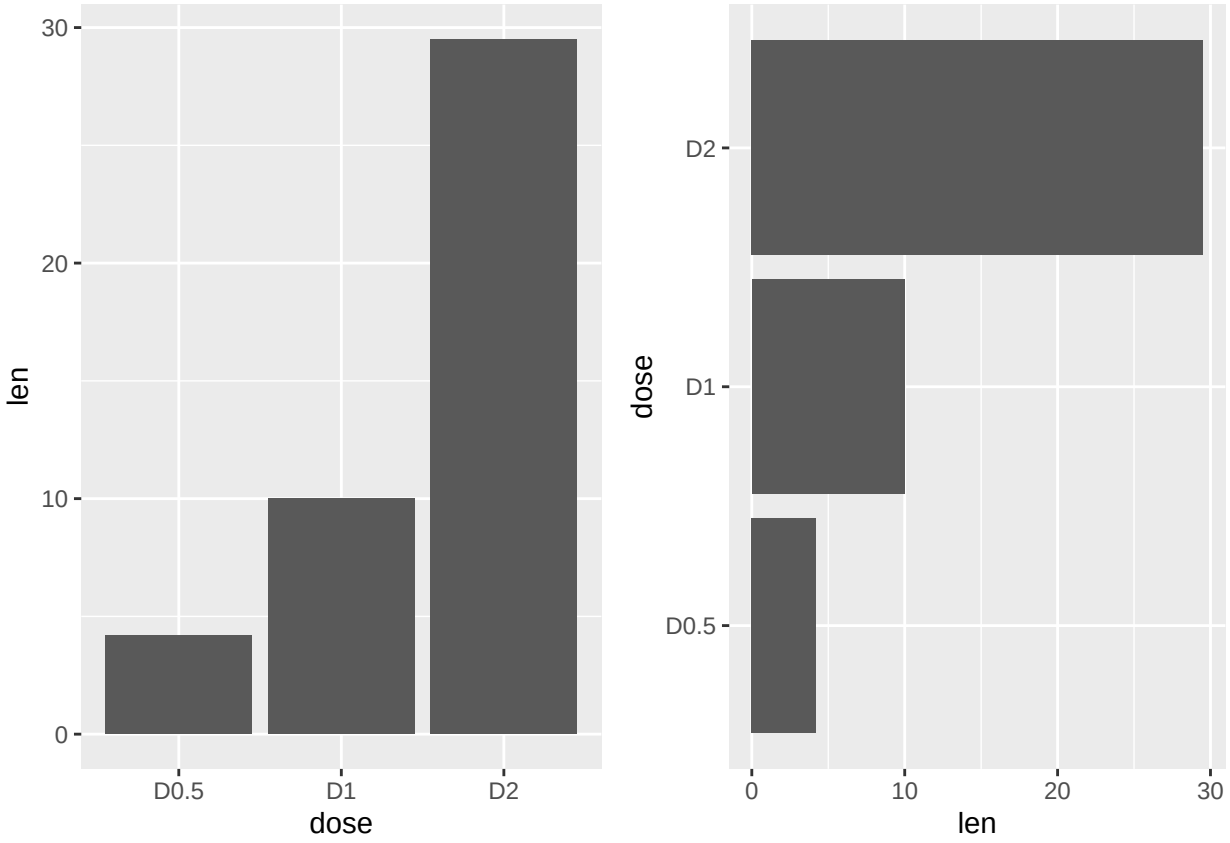
```
df <- data.frame(dose=c("D0.5", "D1", "D2"),
                 len=c(4.2, 10, 29.5))

ggplot(data=df, aes(x=dose, y=len)) +
  geom_bar(stat="identity")
```




```
ggplot(data=df, aes(x=dose, y=len)) +
  geom_bar(stat="identity")+coord_flip()
```




```
library(gridExtra)
p1<- ggplot(data=df, aes(x=dose, y=len)) +
  geom_bar(stat="identity")
p2<- ggplot(data=df, aes(x=dose, y=len)) +
  geom_bar(stat="identity")+coord_flip()
grid.arrange(p1,p2,ncol=2)
```

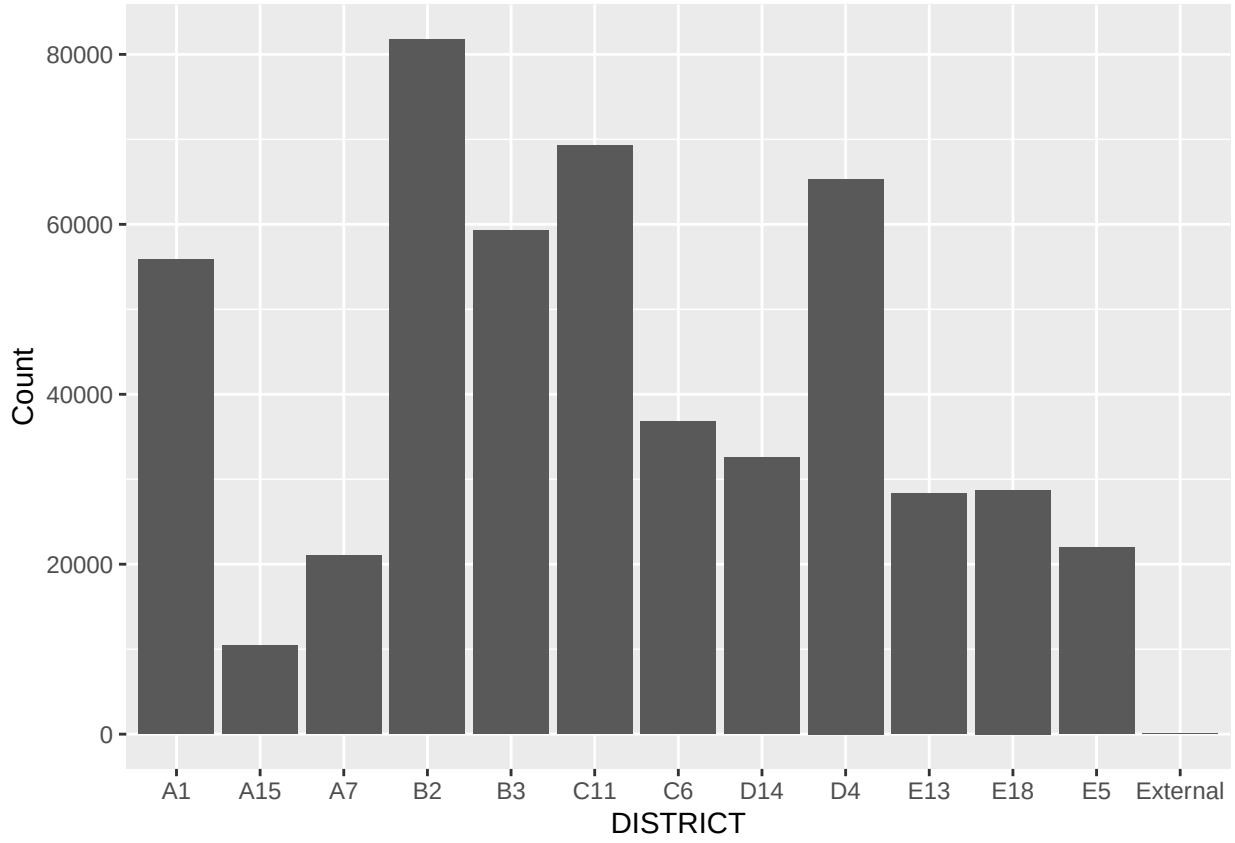


Real world example

```
crime<-read.csv("boston_crime.csv", stringsAsFactors = F, na.strings = "")
```

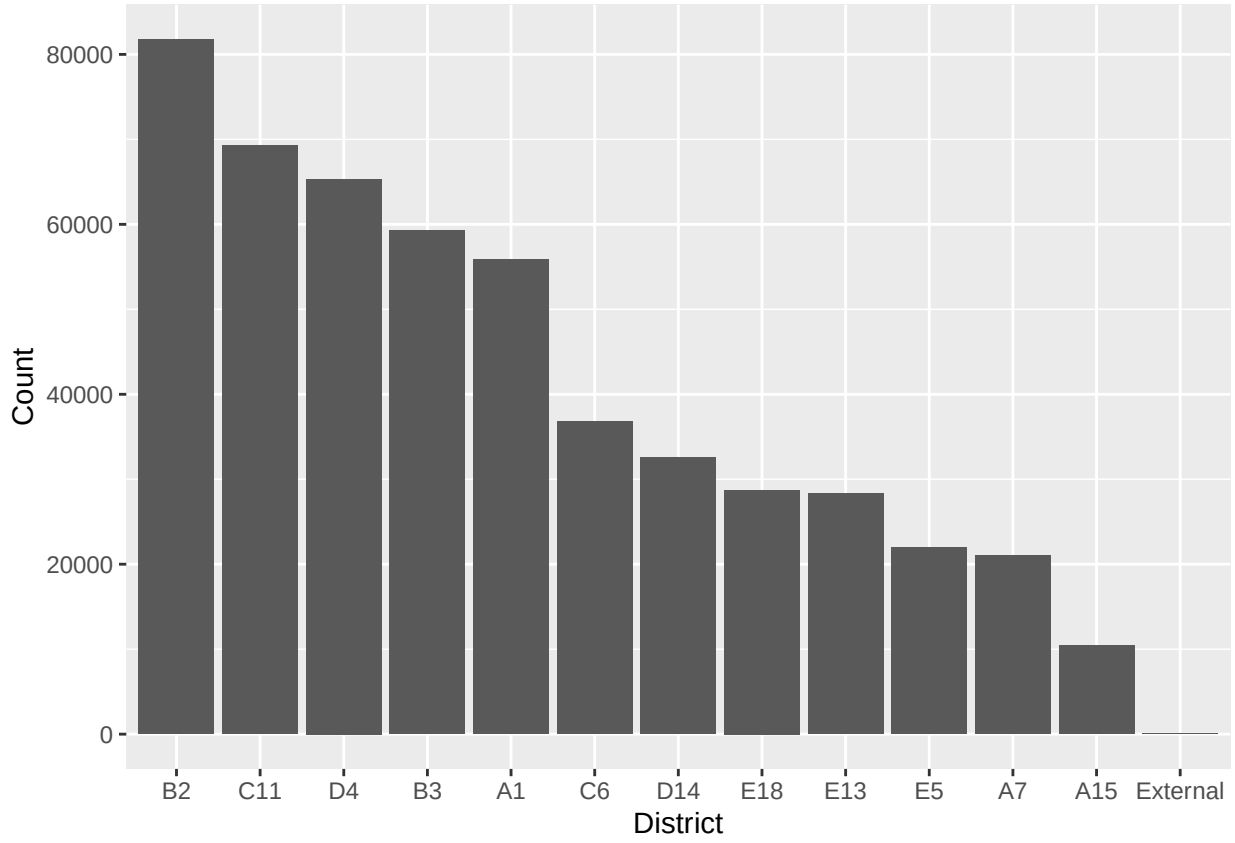
```
crime%>%
  group_by(DISTRICT)%>%
  summarise(Count=n()) %>%
  drop_na()%>%
  ggplot(aes(x=DISTRICT, y=Count))+geom_bar(stat='identity')
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
# Sort the bar graph
crime%>%
  group_by(DISTRICT)%>%
  summarise(Count=n()) %>%
  drop_na()%>%
  arrange(desc(Count))%>%
  ggplot(aes(x=reorder(DISTRICT,-Count), y=Count))+geom_bar(stat='identity')+xlab("District")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
# Sorting example
p1 <- ggplot(mtcars, aes(x = row.names(mtcars), mpg)) +
        geom_bar(stat = "identity") +
        coord_flip() +
        ggtitle("Fig. A: Default rotated x-axis")


p2 <- ggplot(mtcars, aes(x = reorder(row.names(mtcars), mpg), y = mpg)) +
        geom_bar(stat = "identity") +
        coord_flip() +
        ggtitle("Fig. B: Rotated ordered x-axis")

grid.arrange(p1, p2, ncol = 2)
```

Fig. A: Default rotated x

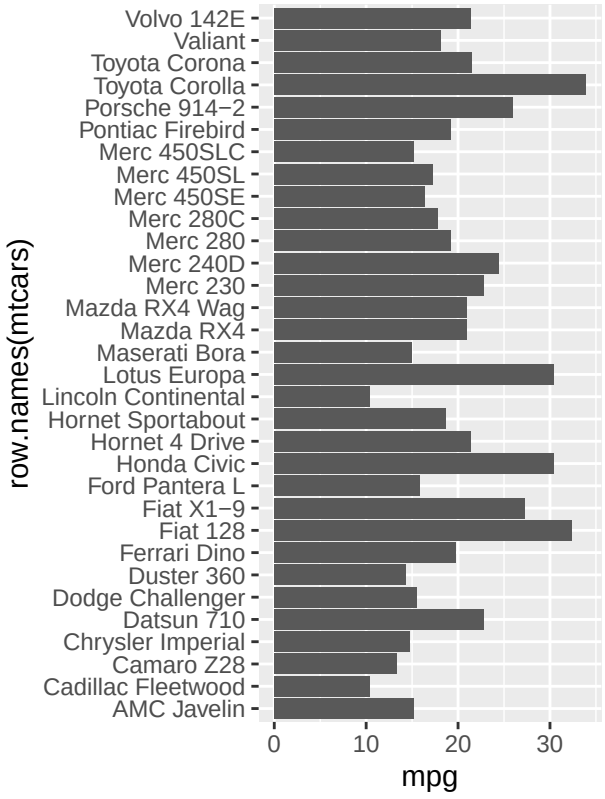
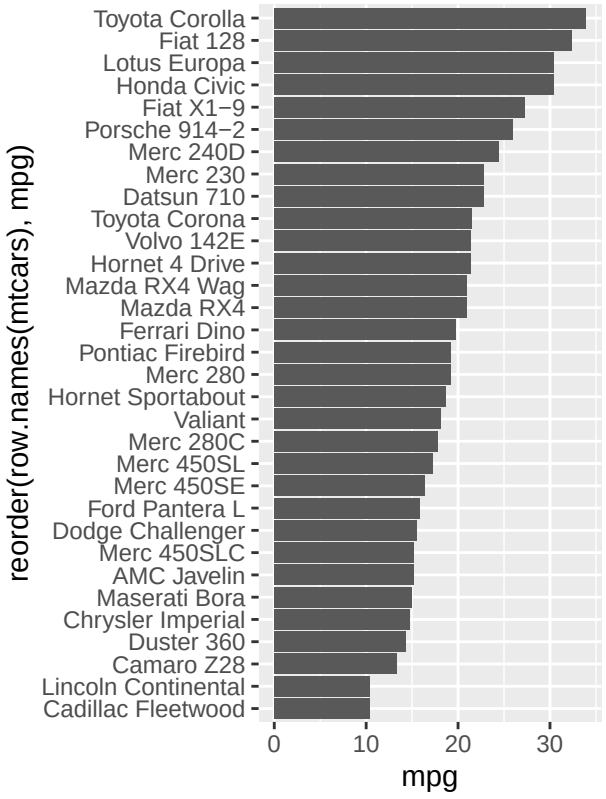
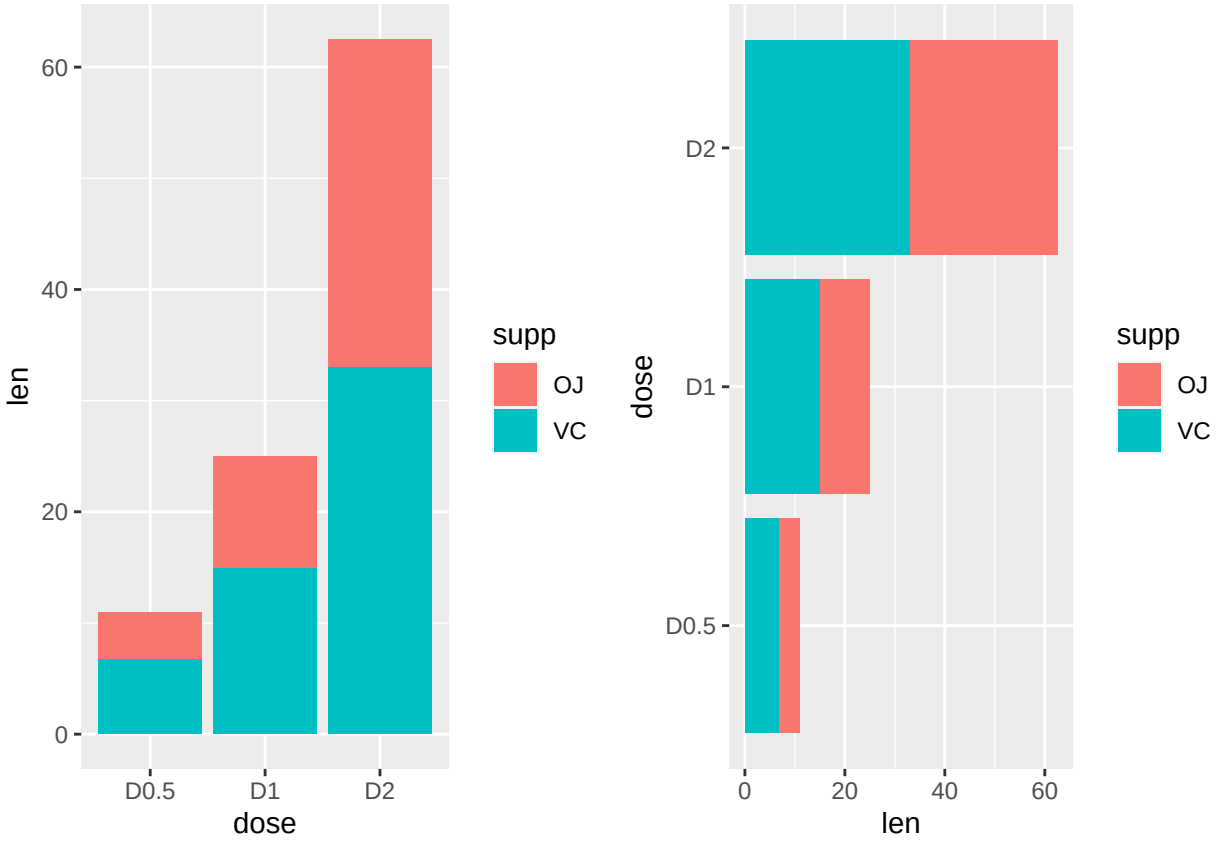


Fig. B: Rotated ordered



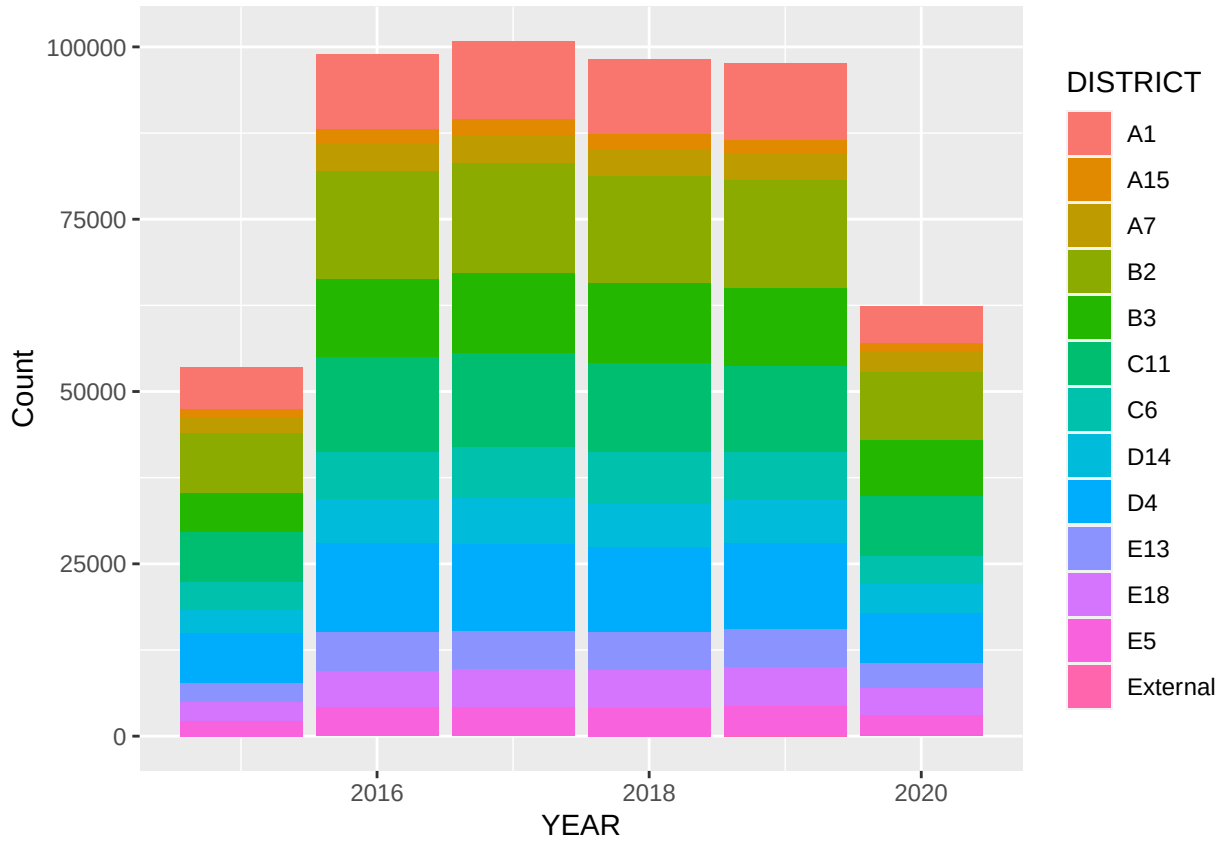
Stacked bar chart

```
df2 <- data.frame(supp=rep(c("VC", "OJ"), each=3),
                  dose=rep(c("D0.5", "D1", "D2"),2),
                  len=c(6.8, 15, 33, 4.2, 10, 29.5))

p1<- ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
  geom_bar(stat="identity")
p2<- p1+coord_flip()
grid.arrange(p1,p2,ncol=2)
```



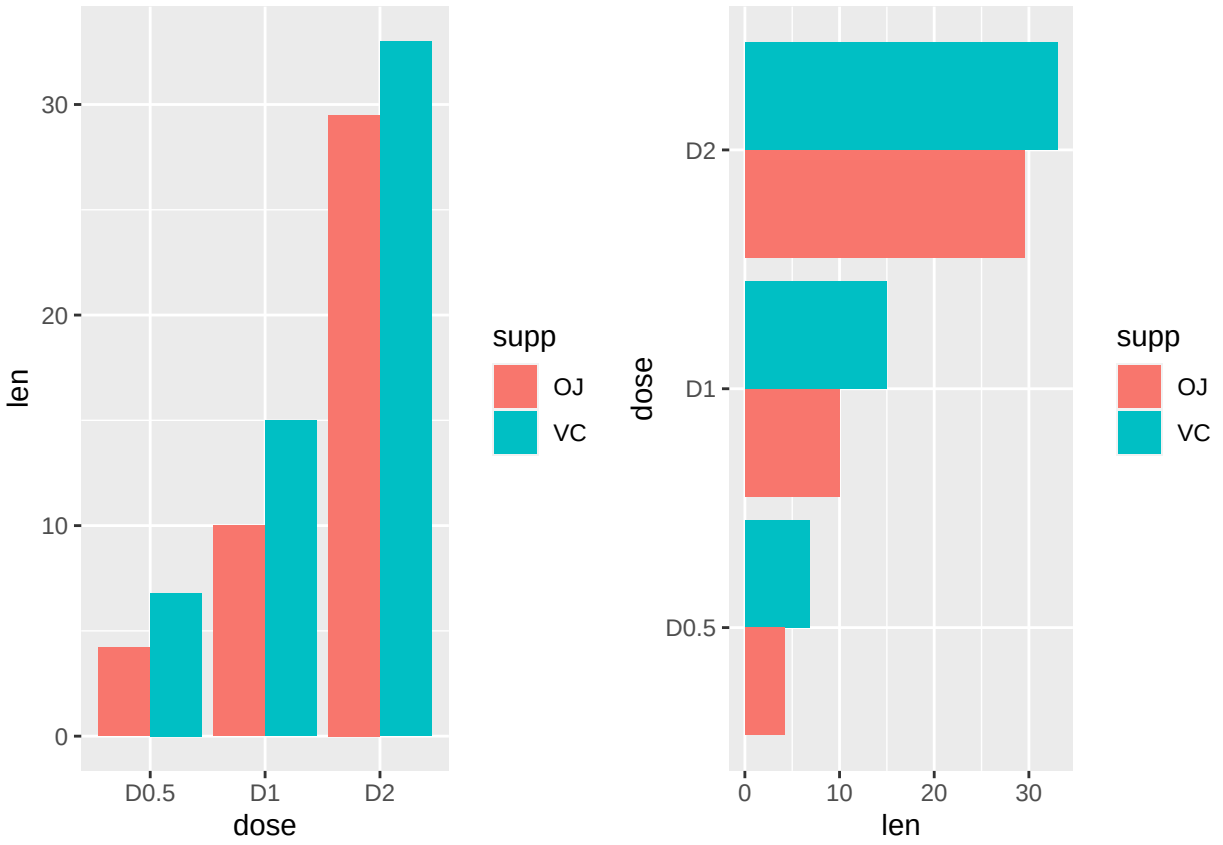

```
# Stacked bar chart: Issue 1-too many factors
crime%>%
  group_by(DISTRICT, YEAR)%>%
  summarise(Count=n()) %>%
  drop_na()%>%
  arrange(desc(Count))%>%
  ggplot(aes(x=YEAR, y=Count, fill=DISTRICT))+geom_bar(stat='identity')
```

```
## `summarise()` regrouping output by 'DISTRICT' (override with `.groups` argument)
```



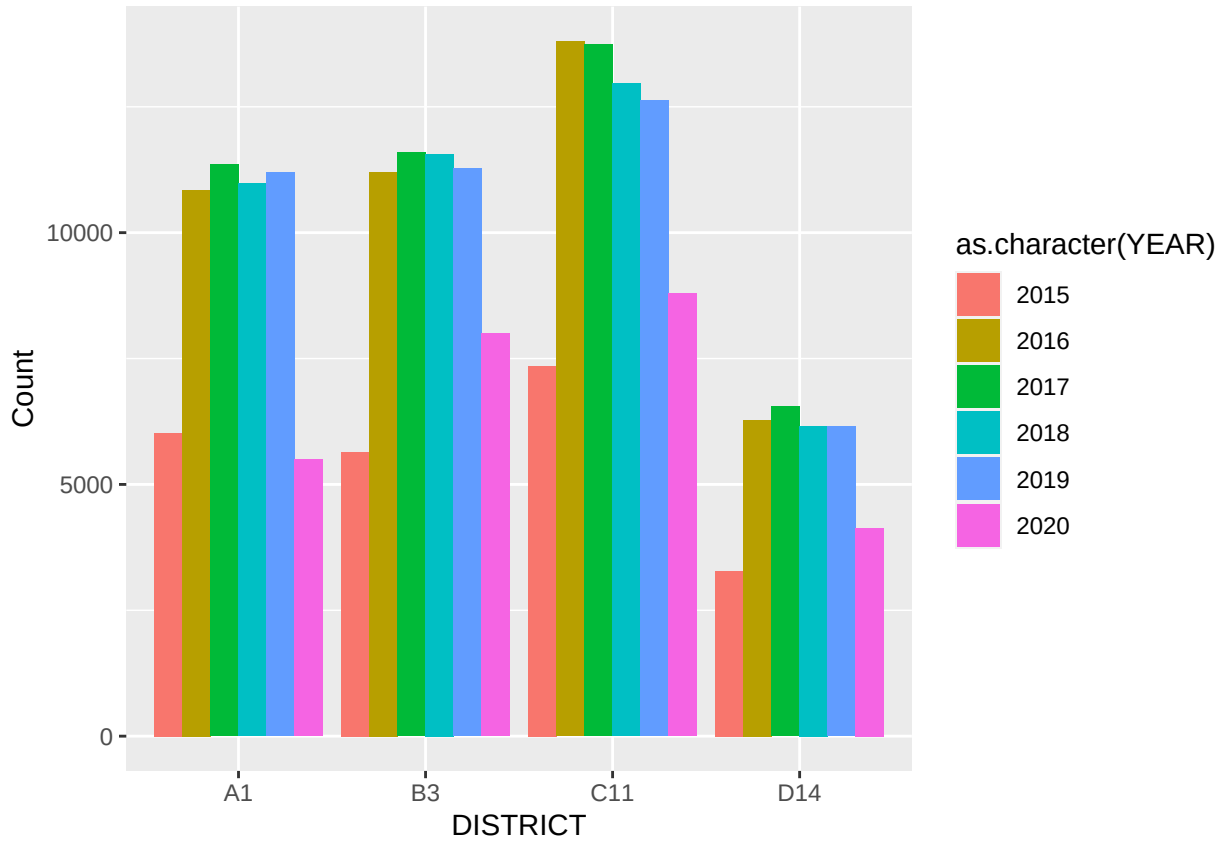
Grouped bar chart

```
p1<- ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
  geom_bar(stat="identity", position="dodge")
p2<- p1+coord_flip()
grid.arrange(p1,p2,ncol=2)
```



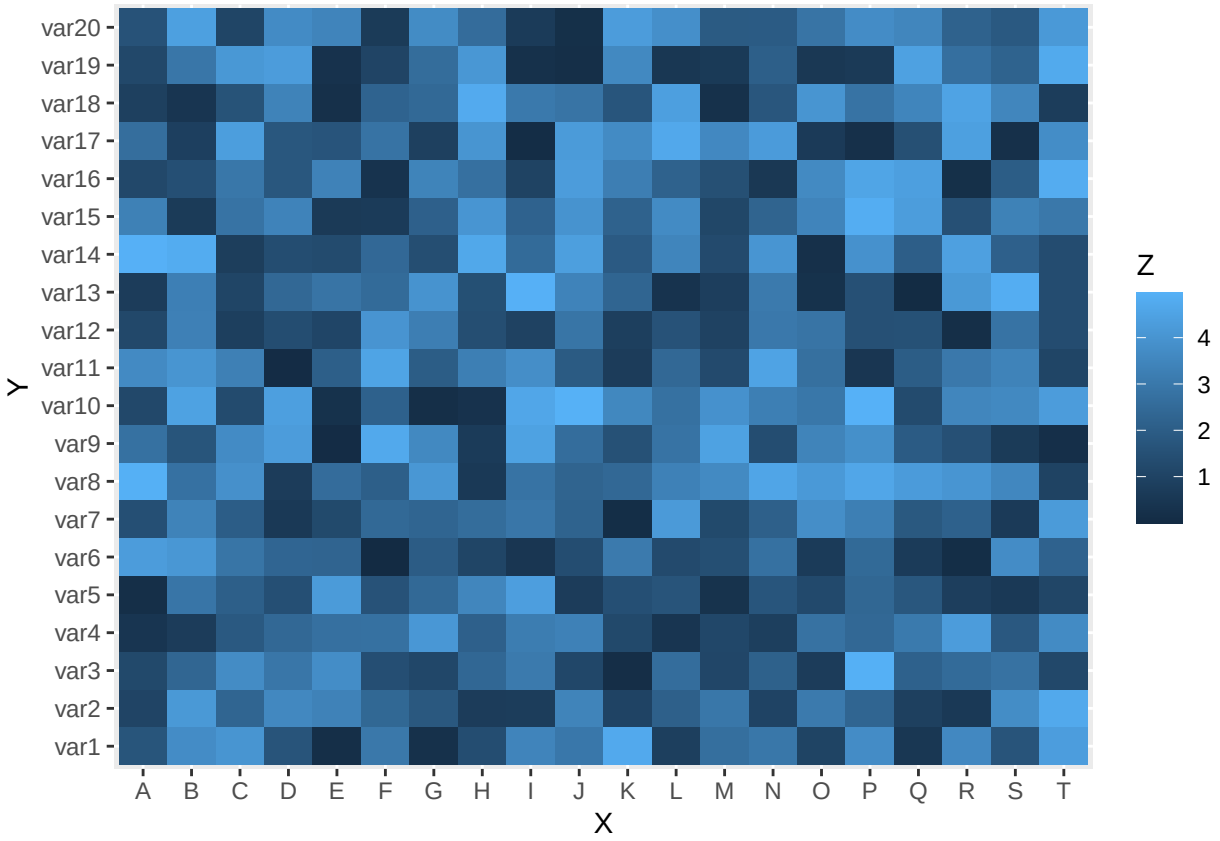

```
# Avoid sorting within factors while using grouped bar plot
crime%>%
  group_by(DISTRICT, YEAR)%>%
   filter(DISTRICT=="A1" | DISTRICT=="B3"| DISTRICT=="D14"| DISTRICT=="C11") %>%
  summarise(Count=n()) %>%
  drop_na()%>%
  ggplot(aes(x=DISTRICT, y=Count, fill=as.character(YEAR)))+geom_bar(stat='identity', position="dodge")
```

```
## `summarise()` regrouping output by 'DISTRICT' (override with `.groups` argument)
```



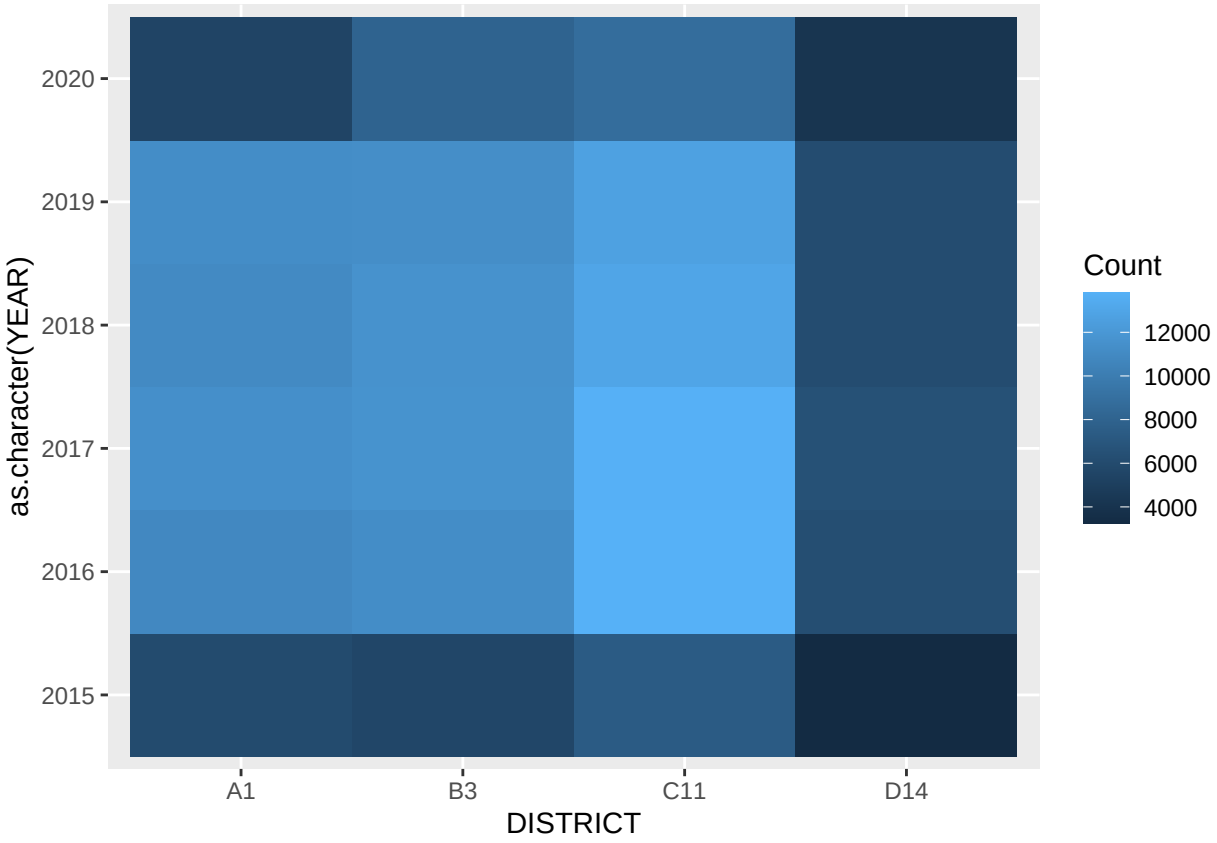
Heat Maps

```
x <- LETTERS[1:20]
y <- paste0("var", seq(1,20))
data <- expand.grid(X=x, Y=y)
data$Z <- runif(400, 0, 5)


ggplot(data, aes(X, Y, fill= Z)) +
  geom_tile()
```

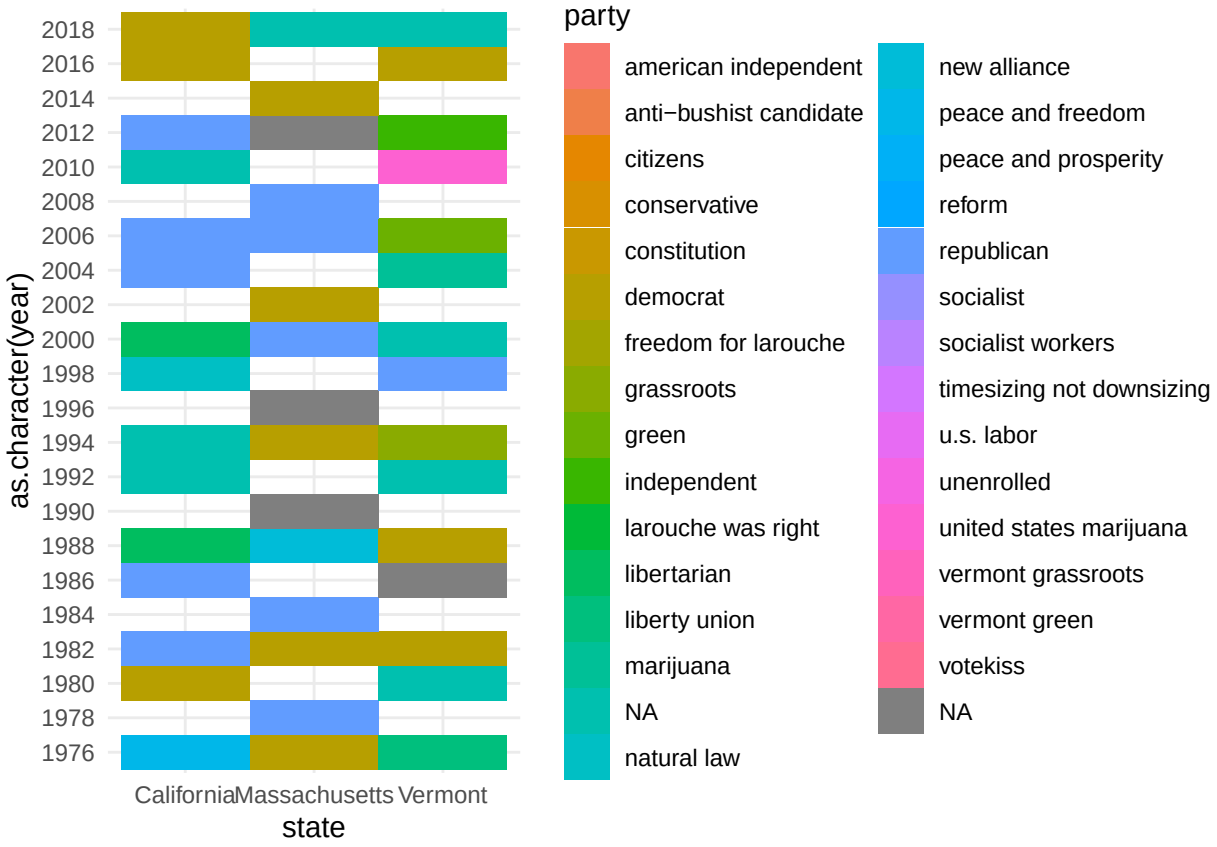



```
crime%>%
  group_by(DISTRICT, YEAR)%>%
   filter(DISTRICT=="A1" | DISTRICT=="B3"| DISTRICT=="D14"| DISTRICT=="C11") %>%
  summarise(Count=n()) %>%
  drop_na()%>%
  ggplot(aes(x=DISTRICT, y=as.character(YEAR), fill=Count))+geom_tile()
```

```
## `summarise()` regrouping output by 'DISTRICT' (override with `.groups` argument)
```

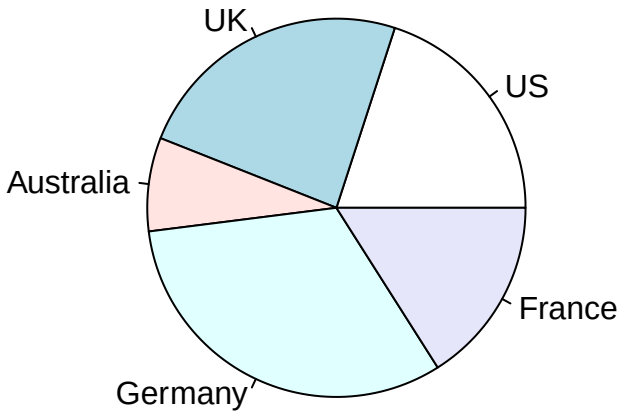



```
senate<-read.csv("senate.csv", stringsAsFactors = F, na.strings = "",  sep = "\t")

senate %>%
  filter(state=="California"| state=="Vermont" |state=="Massachusetts")%>%
  ggplot(aes(x=state, y=as.character(year), fill=party))+geom_tile()+theme_minimal()
```

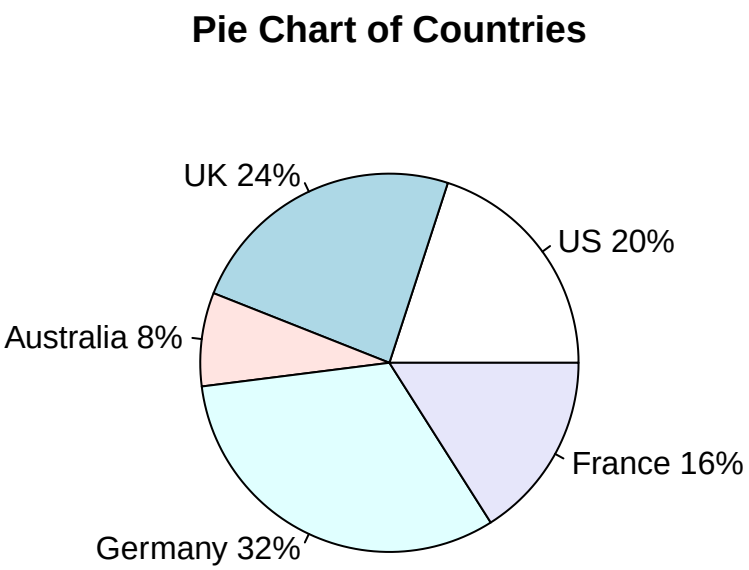


Pie Charts

```
# Simple Pie Chart
slices <- c(10, 12,4, 16, 8)
lbls <- c("US", "UK", "Australia", "Germany", "France")
pie(slices, labels = lbls, main="Pie Chart of Countries")
```




```
slices <- c(10, 12, 4, 16, 8)
lbls <- c("US", "UK", "Australia", "Germany", "France")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie(slices,labels = lbls, main="Pie Chart of Countries")
```



Tree Maps

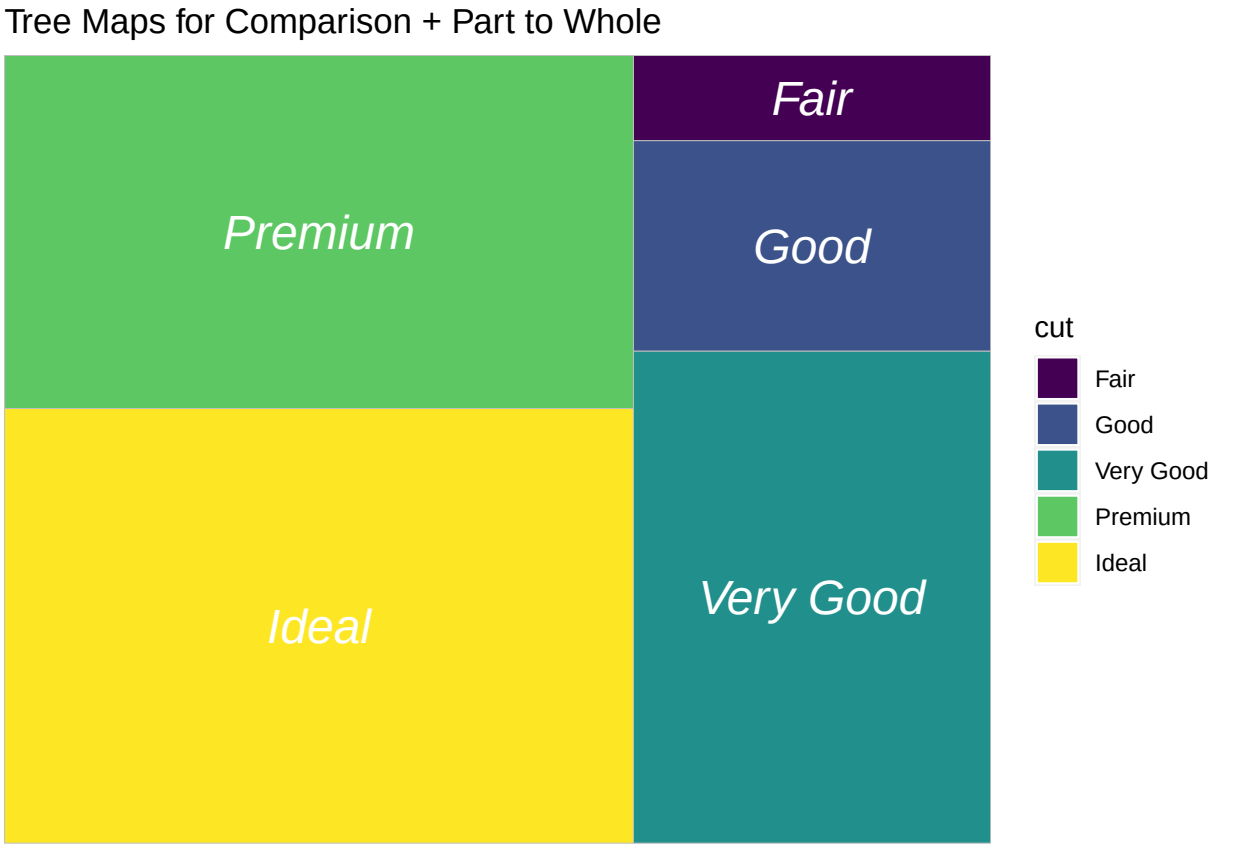
```
library(treemapify)
```

```
## Warning: package 'treemapify' was built under R version 4.0.3
```

```
library(dplyr)
temp<-summarise(group_by(diamonds,cut),total_value=sum(price), total_carat=sum(carat))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
ggplot(temp, aes(area=total_carat,label=cut,fill=cut))+geom_treemap()+ geom_treemap_text(fontface = "ita
```



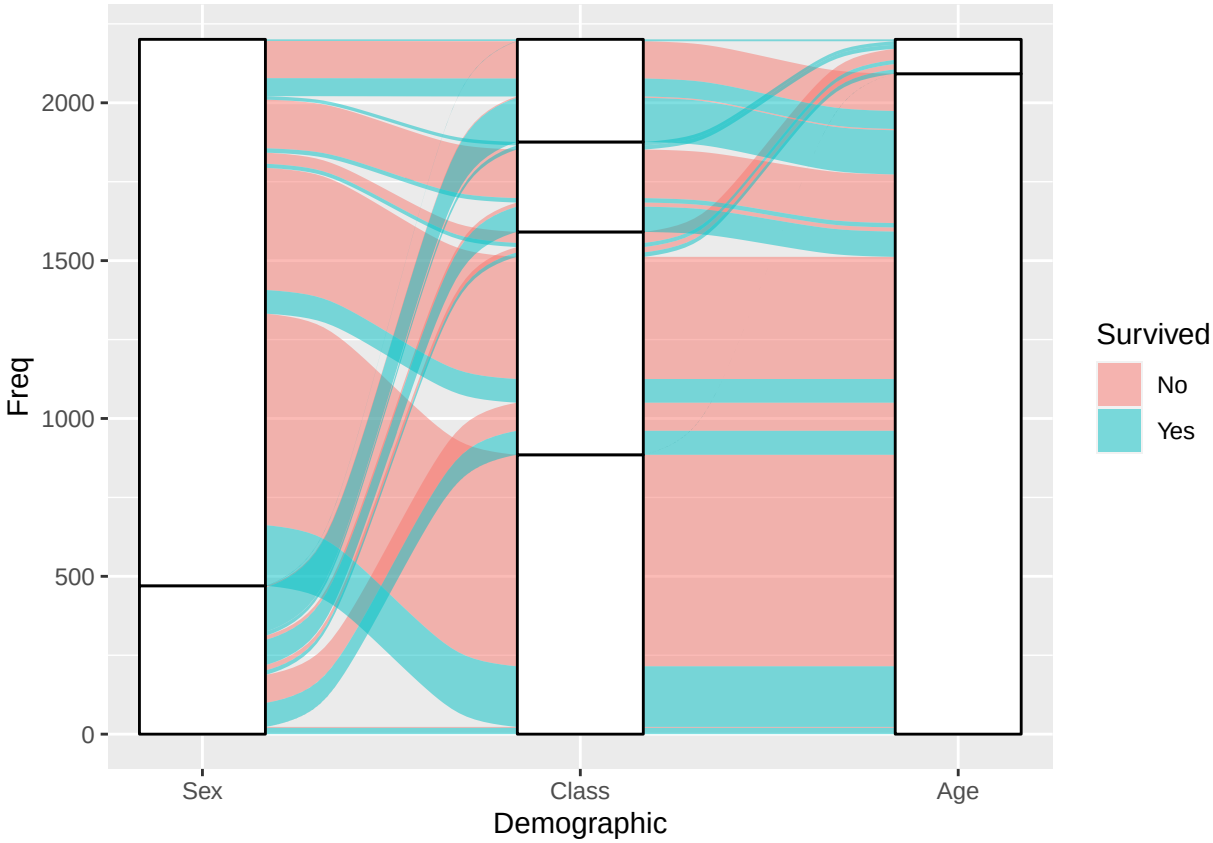
Alluvial Charts

```
library(ggalluvial)
```

```
## Warning: package 'ggalluvial' was built under R version 4.0.3
```

```
titanic_wide <- data.frame(Titanic)
ggplot(data = titanic_wide,
       aes(axis1 = Sex, axis2 = Class, axis3 = Age,
           y = Freq)) +
  scale_x_discrete(limits = c("Sex", "Class", "Age"), expand = c(.1, .05)) +
  xlab("Demographic") +
  geom_alluvium(aes(fill = Survived)) +
  geom_stratum() + geom_text(stat = "stratum", label.strata = TRUE)
```

```
## Warning: Computation failed in `stat_stratum()`:
## The parameter `label.strata` is defunct.
## use `aes(label = after_stat(stratum))`.
```


2000
1500
1000
500
0
Freq
Sex
Class
Age
Demographic
Survived
No
Yes