Laboratory 3

by Joshua Williams

Load the packages

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## Warning: package 'tibble' was built under R version 4.0.4

## Warning: package 'tidyr' was built under R version 4.0.4

## Warning: package 'dplyr' was built under R version 4.0.4

## Warning: package 'forcats' was built under R version 4.0.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(RColorBrewer)
```

multiplot for brevity (not my code)

```
## This isn't my code, I just learned how to apply it online.
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)
```

```
  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

  if (numPlots==1) {
    print(plots[[1]])

  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}
```

Question 3

part a

```
Tesla$Submodel <- as.factor(Tesla$Submodel)
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM))+
  geom_boxplot()+
    labs(title = "Without `Fill`")

g2 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
    labs(title = "With `Fill`")
g3 <-multiplot(g1,g2,cols=2)
```

```
#To make clean up and labeling easier for plots
g3
```

```
## NULL
```

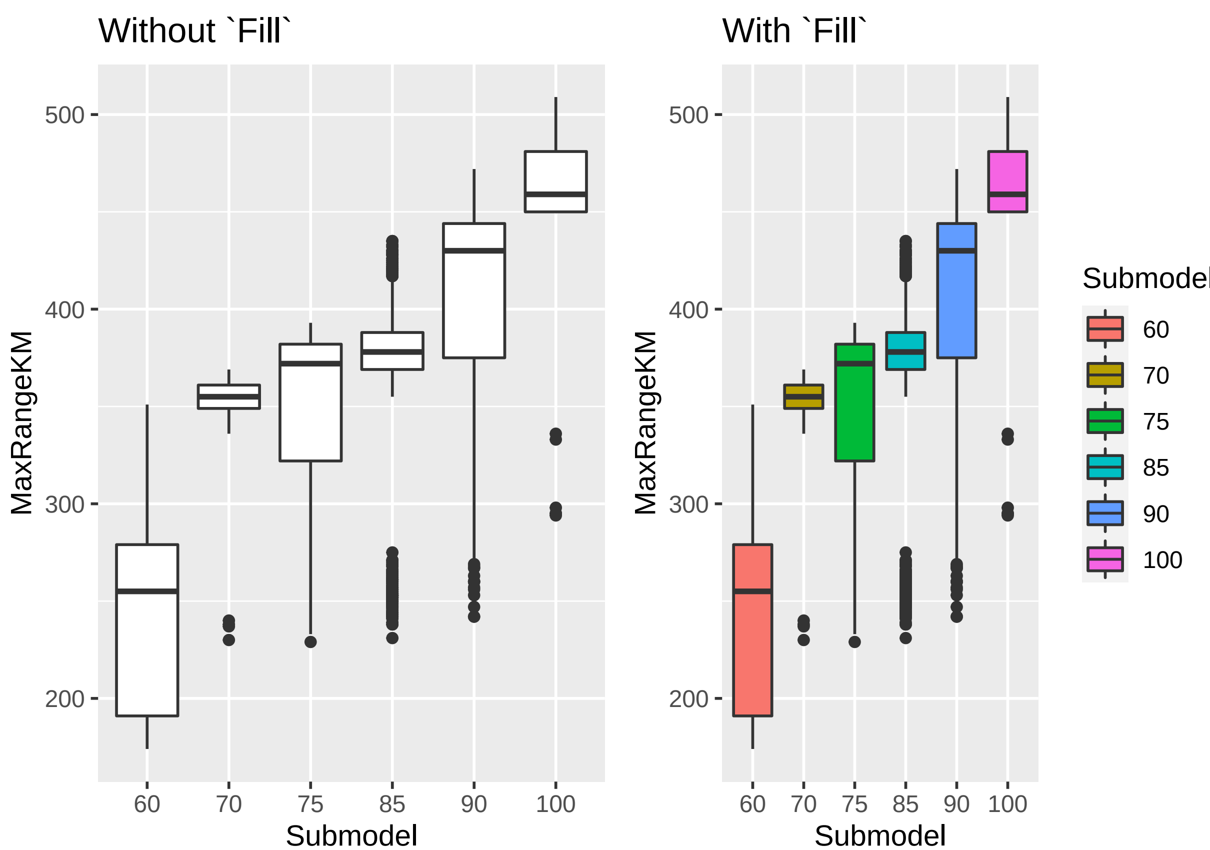


Figure 1: Compares chart with and without the use of `fill`

```
rm(g1,g2,g3)
```

part b

```
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "Set1")+
  ggtitle("palette = \'Set1\'")
g2 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "RdBu")+
  ggtitle("palette = \'RdBu\'")
g3 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "YlGnBu")+
  ggtitle("palette = \'YlGnBu\'")
g4 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "Paired")+
  ggtitle("palette = \'Paired\'")
multiplot(g1,g2,g3,g4,cols=2)
```

```
rm(g1,g2,g3,g4)
```


```
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "Set1")+
  ggtitle("palette = \'Set1\'")
g2 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "RdBu")+
  ggtitle("palette = \'RdBu\'")
g3 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "YlGnBu")+
  ggtitle("palette = \'YlGnBu\'")
g4 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "Paired")+
  ggtitle("palette = \'Paired\'")
multiplot(g1,g2,g3,g4,cols=2)
```

```
rm(g1,g2,g3,g4)
```


part c

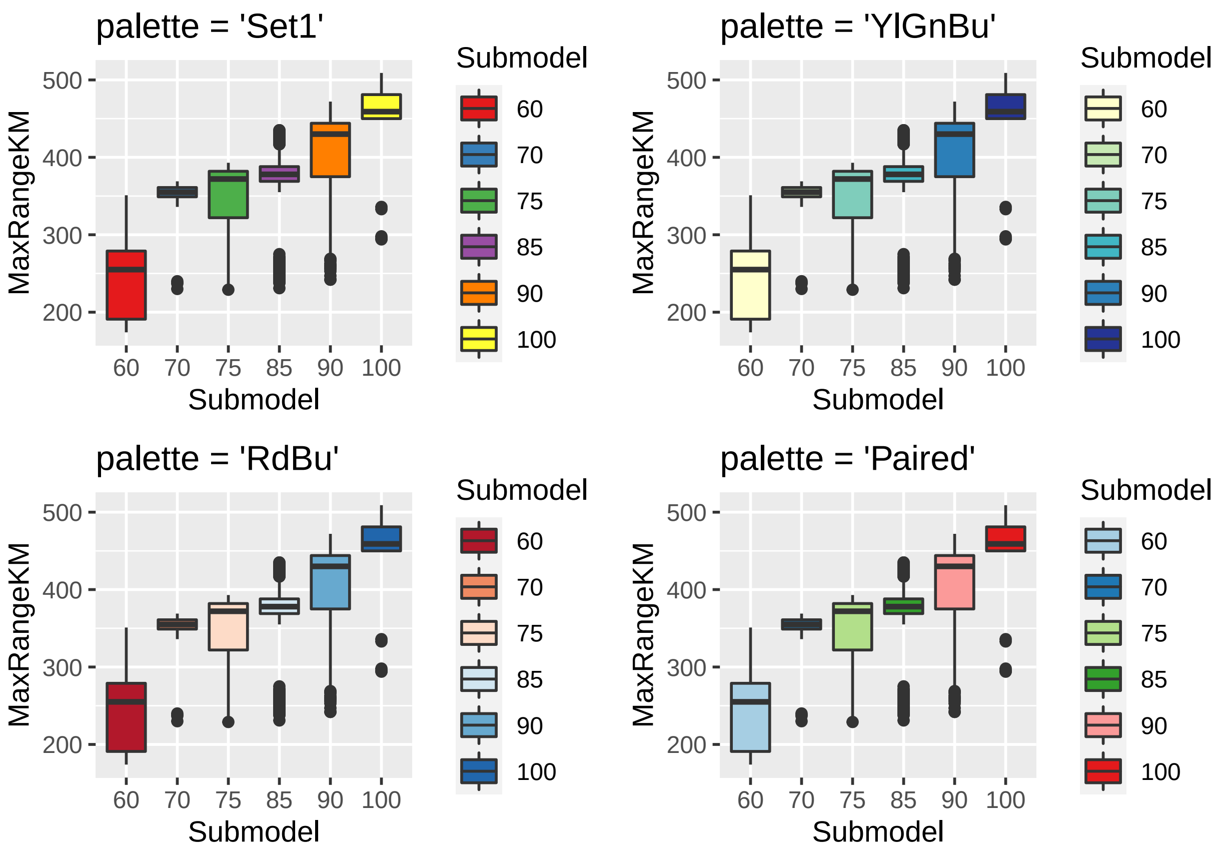


Figure 2: Shows multiple optional colour palettes for the same data

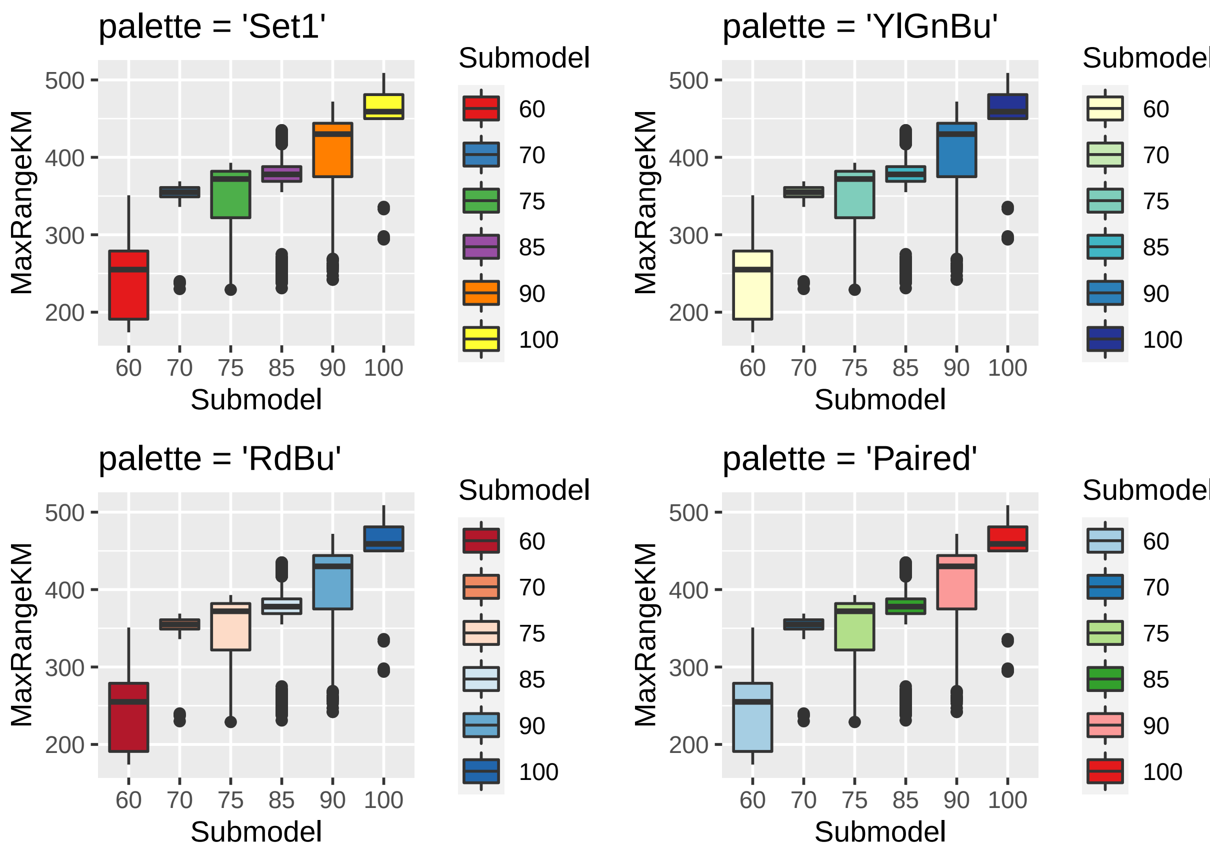


Figure 3: Shows multiple optional colour palettes for the same data examples from the lab script

```
library(viridis)
```

```
## Loading required package: viridisLite

##
## Attaching package: 'viridis'

## The following object is masked from 'package:viridisLite':
##
##     viridis.map
```

```
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_viridis(discrete = TRUE, option = "viridis")+
  ggtitle("Relationship between the Tesla Submodel and the Range (km)")+
  xlab("Submodel of Tesla")+
  ylab("Maximum Range (km)")
g1
```

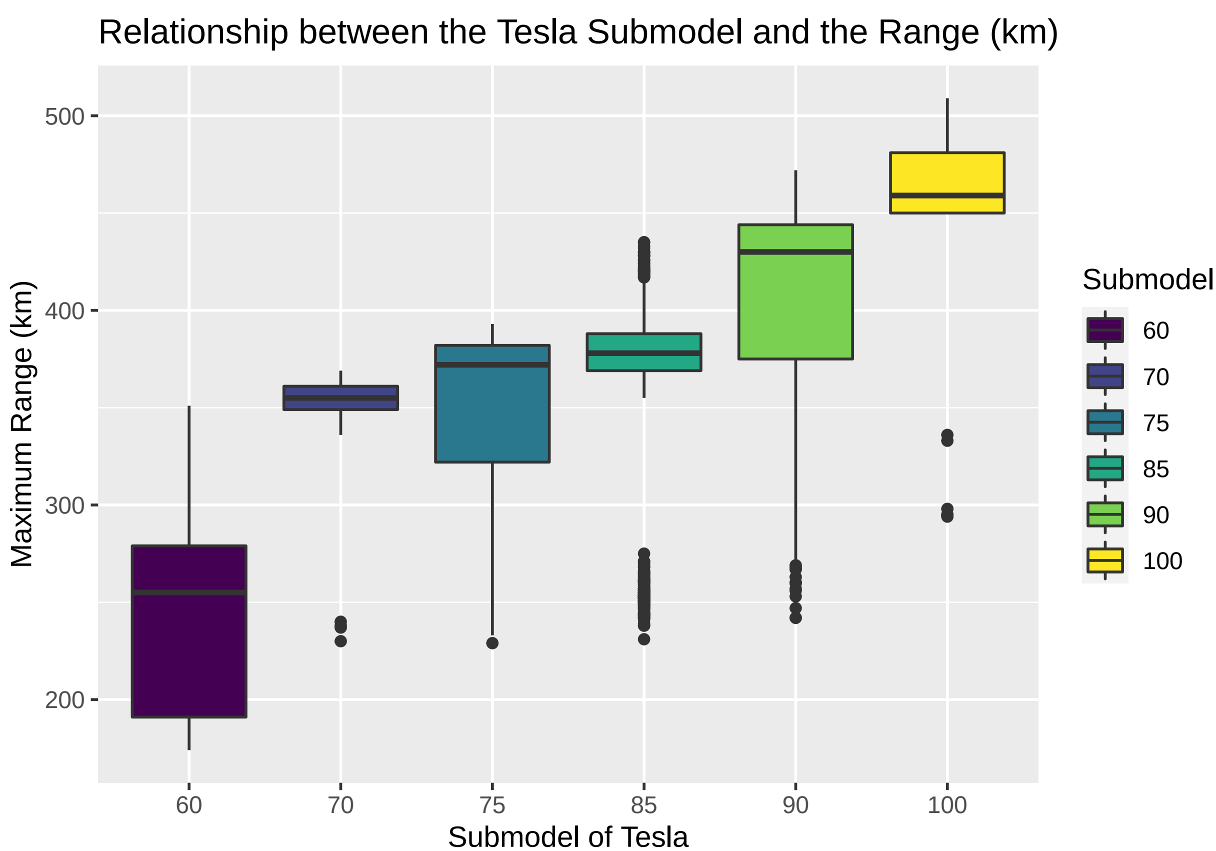


Figure 4: show the data with added labels and a more descriptive title

```
library("viridis")
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_viridis(discrete = TRUE, option = "viridis")+
  ggtitle("palette = \'viridis\'", subtitle="best for colourblindness")
g2 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_viridis(discrete = TRUE, option = "magma")+
  ggtitle("palette = \'magma\'")
g3 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_viridis(discrete = TRUE, option = "plasma")+
  ggtitle("palette = \'plasma\'")
g4 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_boxplot()+
  scale_fill_viridis(discrete = TRUE, option = "inferno")+
  ggtitle("palette = \'inferno\'")
multiplot(g1,g2,g3,g4,cols=2)
```

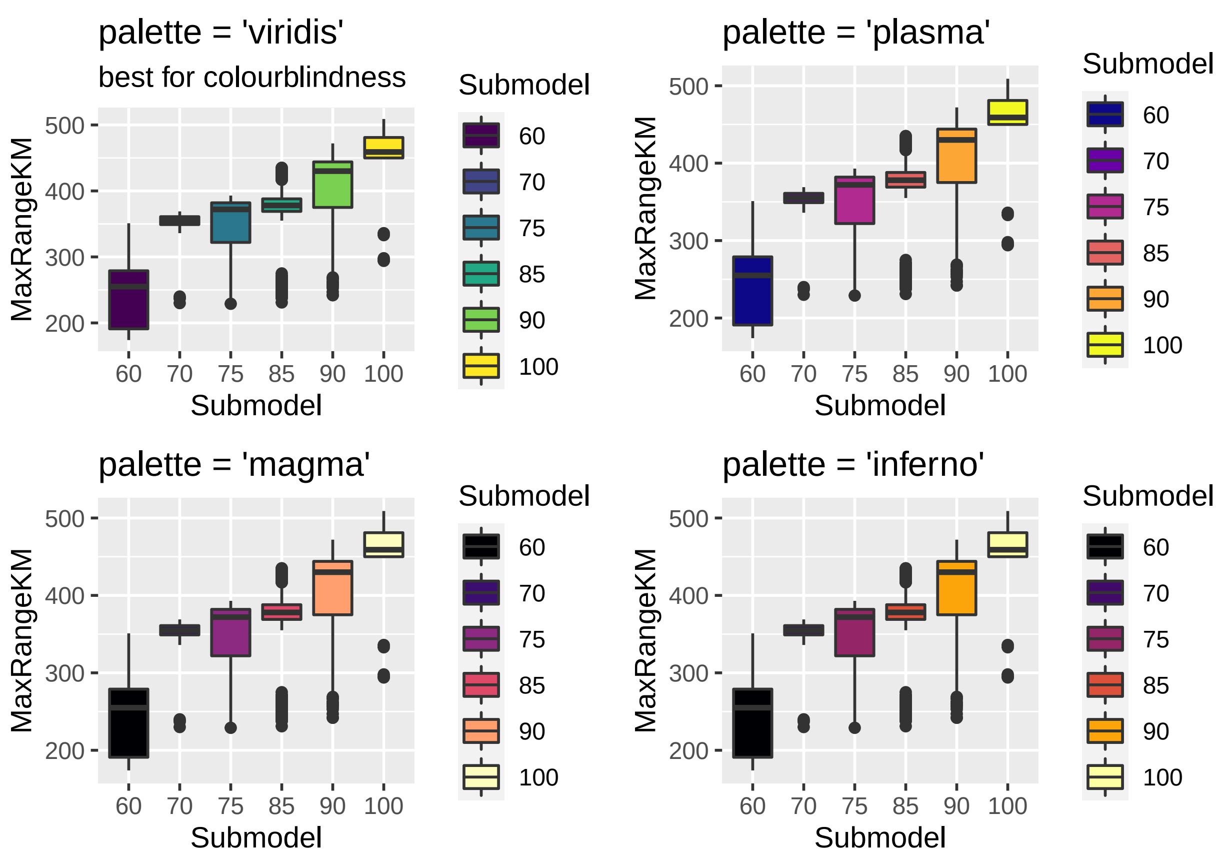


Figure 5: Shows multiple optional colour palettes for the same data examples from the lab script

```
rm(g1,g2,g3,g4)
```

part d

```
library("viridis")
g1 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_violin()+
  scale_fill_viridis(discrete = TRUE, option = "viridis")+
  ggtitle("palette = \'viridis\'", subtitle="best for colourblindness")
g2 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_violin()+
  scale_fill_viridis(discrete = TRUE, option = "magma")+
  ggtitle("palette = \'magma\'")
g3 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_violin()+
  scale_fill_viridis(discrete = TRUE, option = "plasma")+
  ggtitle("palette = \'plasma\'")
g4 <- ggplot(data=Tesla, aes(x=Submodel,y=MaxRangeKM, fill = Submodel))+
  geom_violin()+
  scale_fill_viridis(discrete = TRUE, option = "inferno")+
  ggtitle("palette = \'inferno\'")
multiplot(g1,g2,g3,g4,cols=2)
```

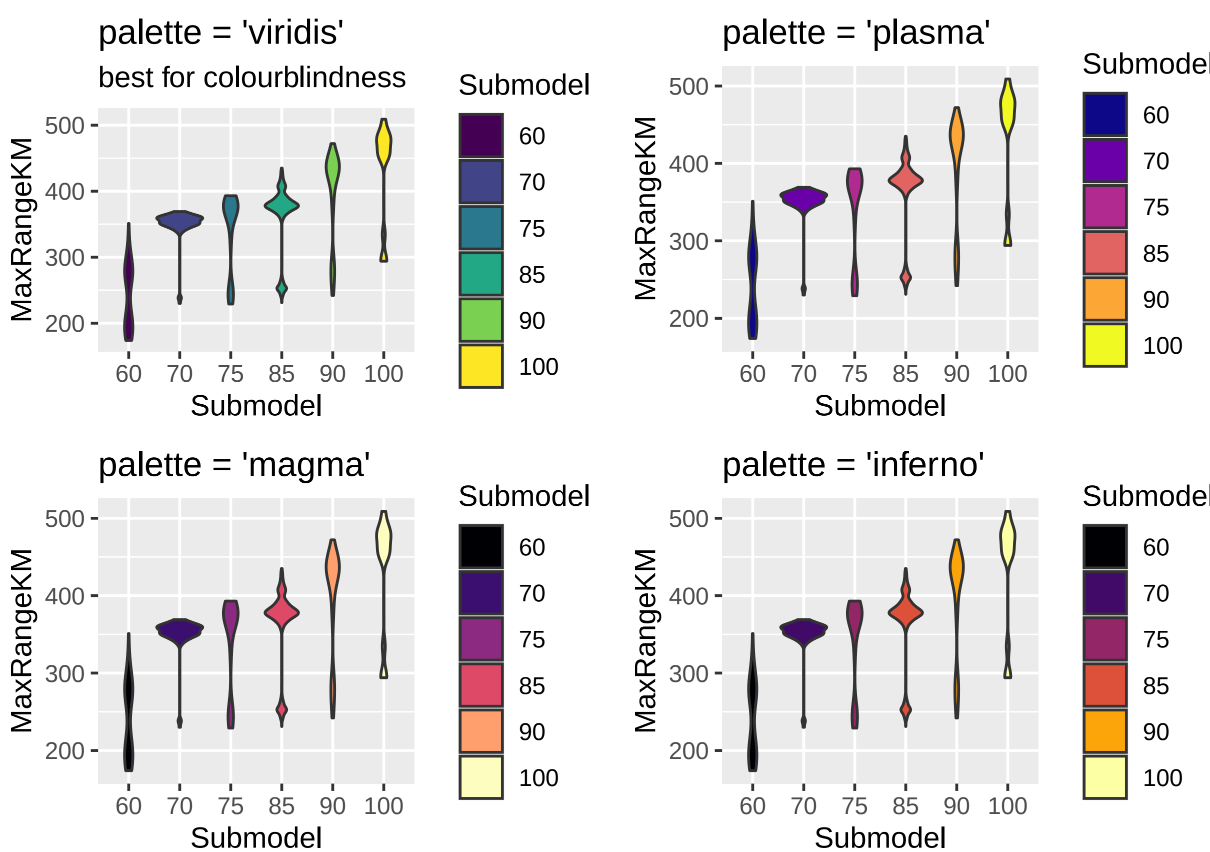


Figure 6: The data are now in the form of a violin plot

```r
rm(g1,g2,g3,g4)
```

part e

```r
g1<- ggplot(data=Tesla,aes(x=Submodel, y = MaxRangeKM, fill = Model))+
  geom_boxplot()+
  ggtitle(label="Boxplot By Model")
g1
```

```
## Warning: Removed 1 rows containing non-finite values (stat_boxplot).
```

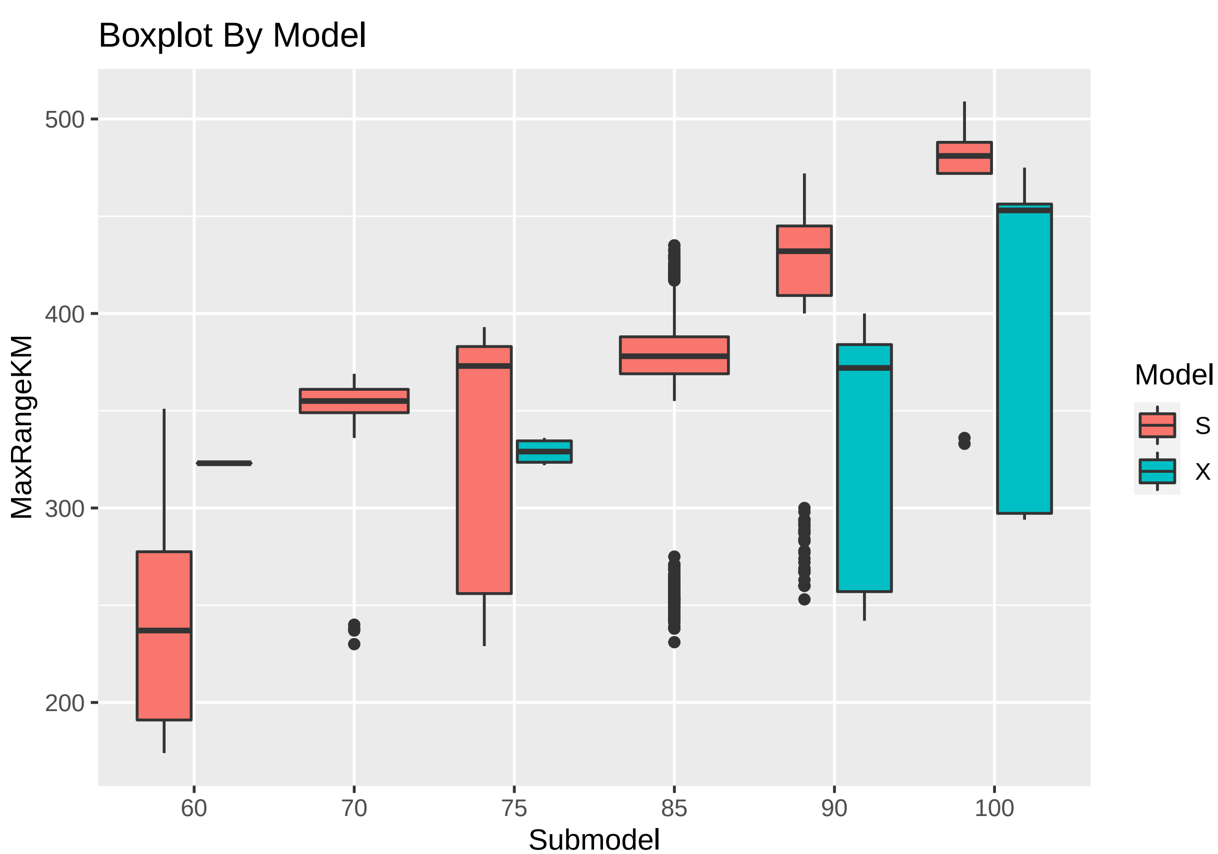


Figure 7: Alternative Data Configuration of Data When Grouped By Model

```r
rm(g1)
```

When may this be useful?

Some patterns become hidden when grouping distinct groups of related data together. To avoid missing these patterns, it helps to group data in several different ways so as to see any patterns that may emerge.

Question 4

Part a

```
g1<-ggplot(data = Tesla, aes(x=Submodel, fill = Model))+
  geom_bar()+
  labs(title = "default setting for geom_bar()")
g2<- ggplot(data = Tesla, aes(x=Submodel, fill = Model))+
  geom_bar(aes(y=100*..count../sum(..count..)), position = 'dodge')+
labs(title="dodge setting for geom_bar()")+
  xlab("Submodel")+
  ylab("percentage")
g3<- ggplot(data = Tesla, aes(x=Submodel, fill = Model))+
  geom_bar(position = 'fill')+
  labs(title="fill setting for geom_bar()")
multiplot(g1,g2,g3,cols=1)
```

Question 5

part a

```
g1 <- ggplot(data = Tesla, aes(x=AgeInDays, y = MileageKM))+
  geom_point()+
  ggtitle(label = "Simple Scatterplot")+
  xlab("Age in Days")+
  ylab("Mileage in Kilometers")
g2 <- ggplot(data = Tesla, aes(x=AgeInDays, y = MileageKM))+
  geom_point(col = 'lightblue')+
  geom_smooth(method = "lm", color = "purple")+
  ggtitle(label = "Scatterplot with Linear Regression")+
  xlab("Age in Days")+
  ylab("Mileage in Kilometers")

multiplot(g1,g2,cols=1)
```

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

Question 6

Which Models are more popular on which continent (what variable is this in the dataset?)?

```
g1<- ggplot(data = Tesla, aes(x = Location, fill=Model))+
  geom_bar()
```

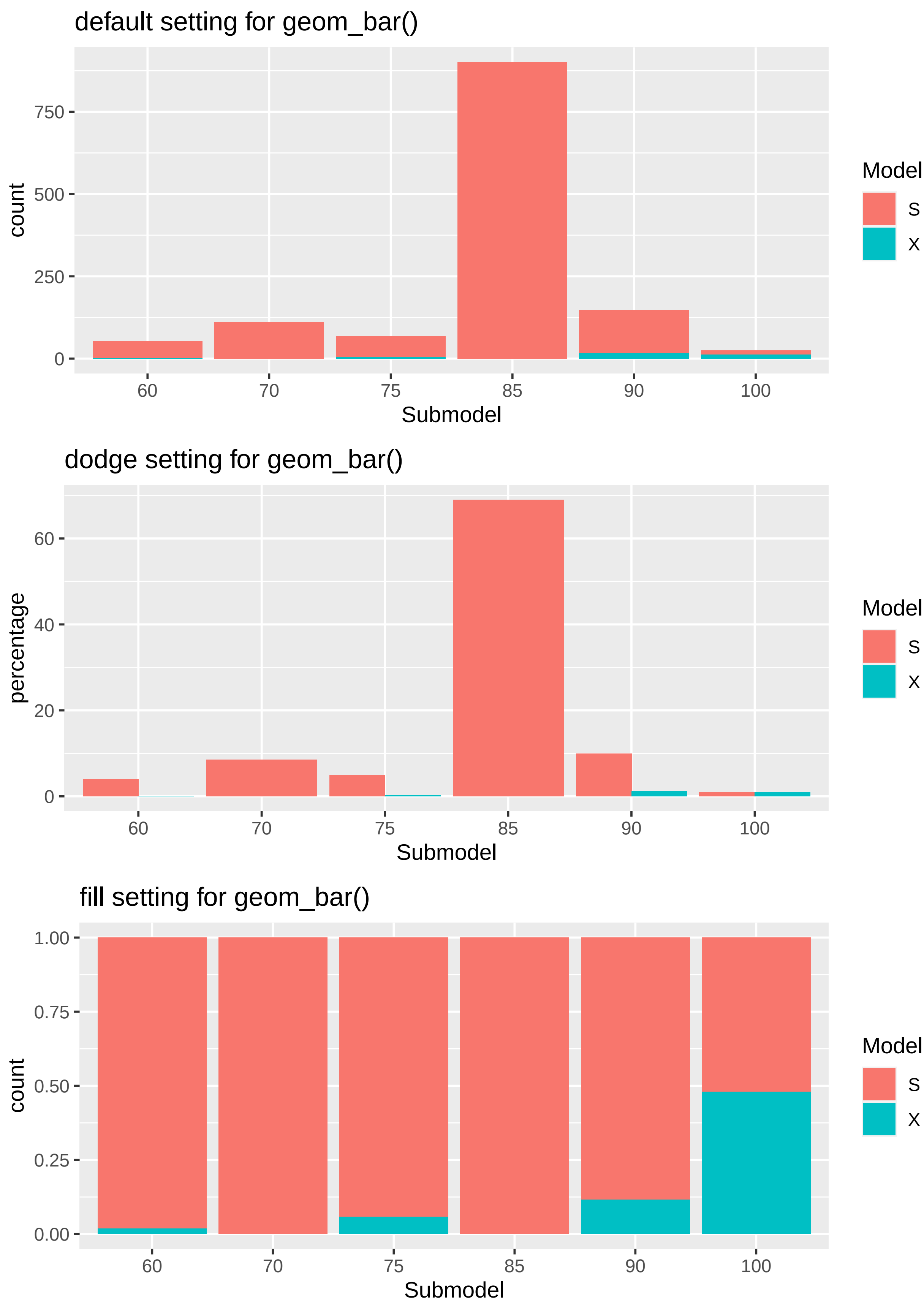


Figure 8: different representations of the same data using geom_bar()

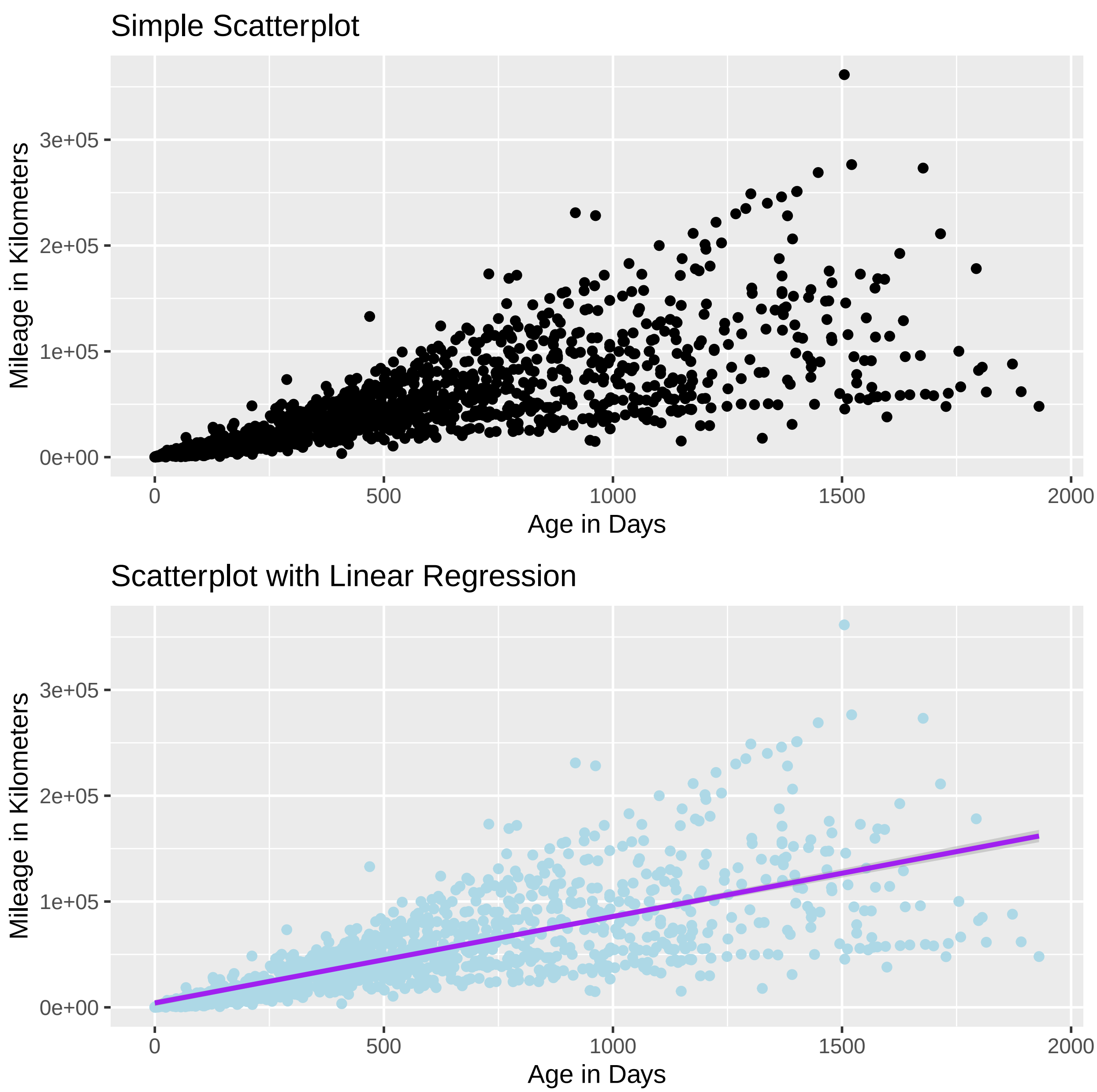


Figure 9: demonstrates the ability to alter color schemes of the lines of best fit and the scatterplots from which that linear regression is derived

```
g2<- ggplot(data = Tesla, aes(x = Location, fill=Model))+
  geom_bar(position = 'dodge')
g3<- ggplot(data = Tesla, aes(x = Location, fill=Model))+
  geom_bar(position = 'fill')
multiplot(g1,g2,g3,cols=1)
```

Question 7

Do different continents have different daily driving patterns (MileagePerDay)?

```
g1<- ggplot(data = Tesla, aes(x = Location, fill=MileagePerDay))+
  geom_bar()
g2<- ggplot(data = Tesla, aes(x = Location, fill=MileagePerDay))+
  geom_bar(position = 'dodge')
g3<- ggplot(data = Tesla, aes(x = Location, fill=MileagePerDay))+
  geom_bar(position = 'fill')
multiplot(g1,g2,g3,cols=1)
```

Question 8

Which Location tends to have older cars?

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
g1<- ggplot(data = Tesla)+
  geom_density(mapping = aes(x=AgeInDays),size=2)+
  facet_grid(Location~.)
g1
```

```
df <- data.frame(Location = Tesla$Location,
                 AgeInDays= Tesla$AgeInDays)
df1<- aggregate(df,by=list(df$Location),FUN=mean,na.rm=TRUE)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

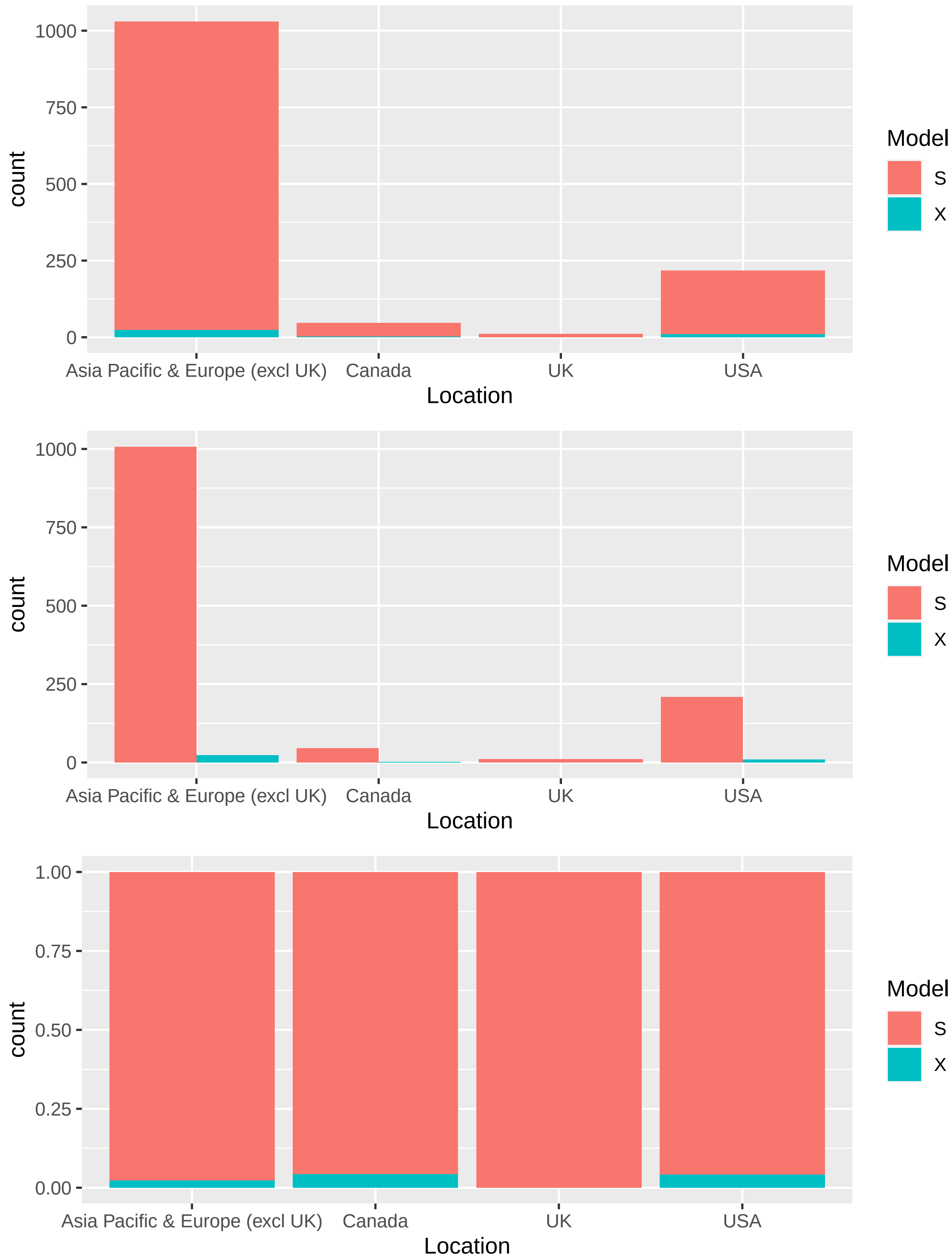


Figure 10: is the answer to question 6

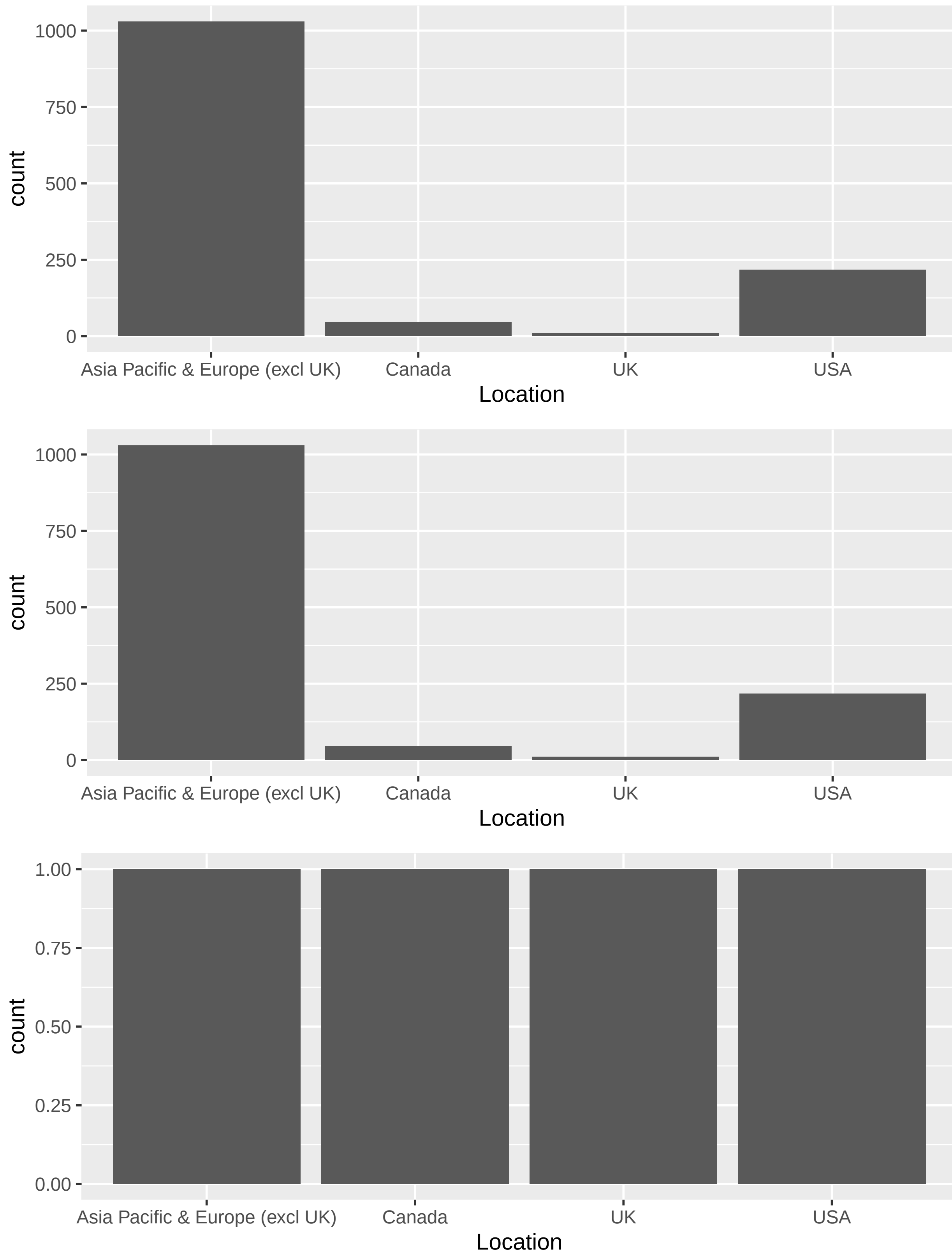


Figure 11: is the answer to question 7

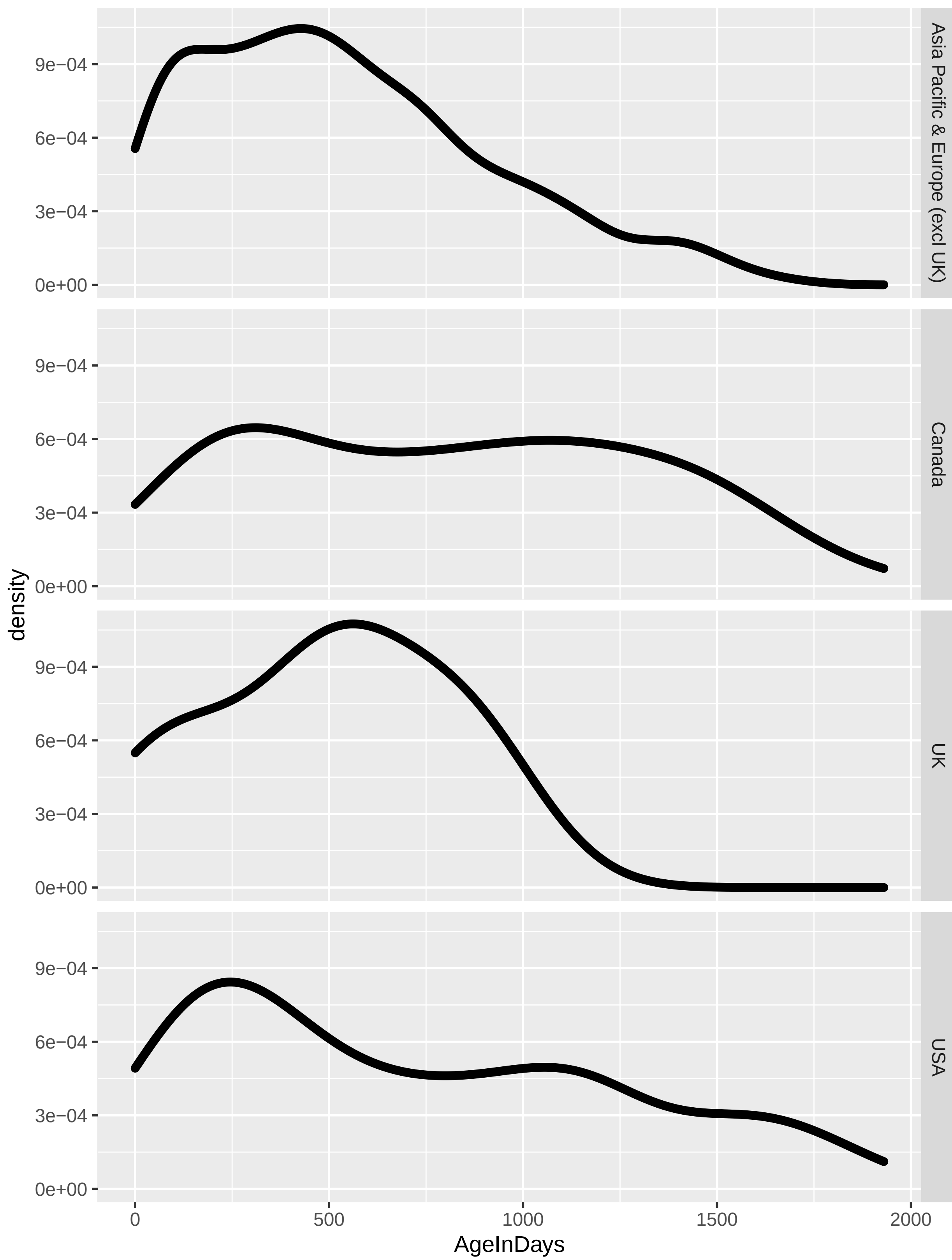


Figure 12: Shows the density function of each location. The center of mass of the density function indicates the indicates average age of each car.

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

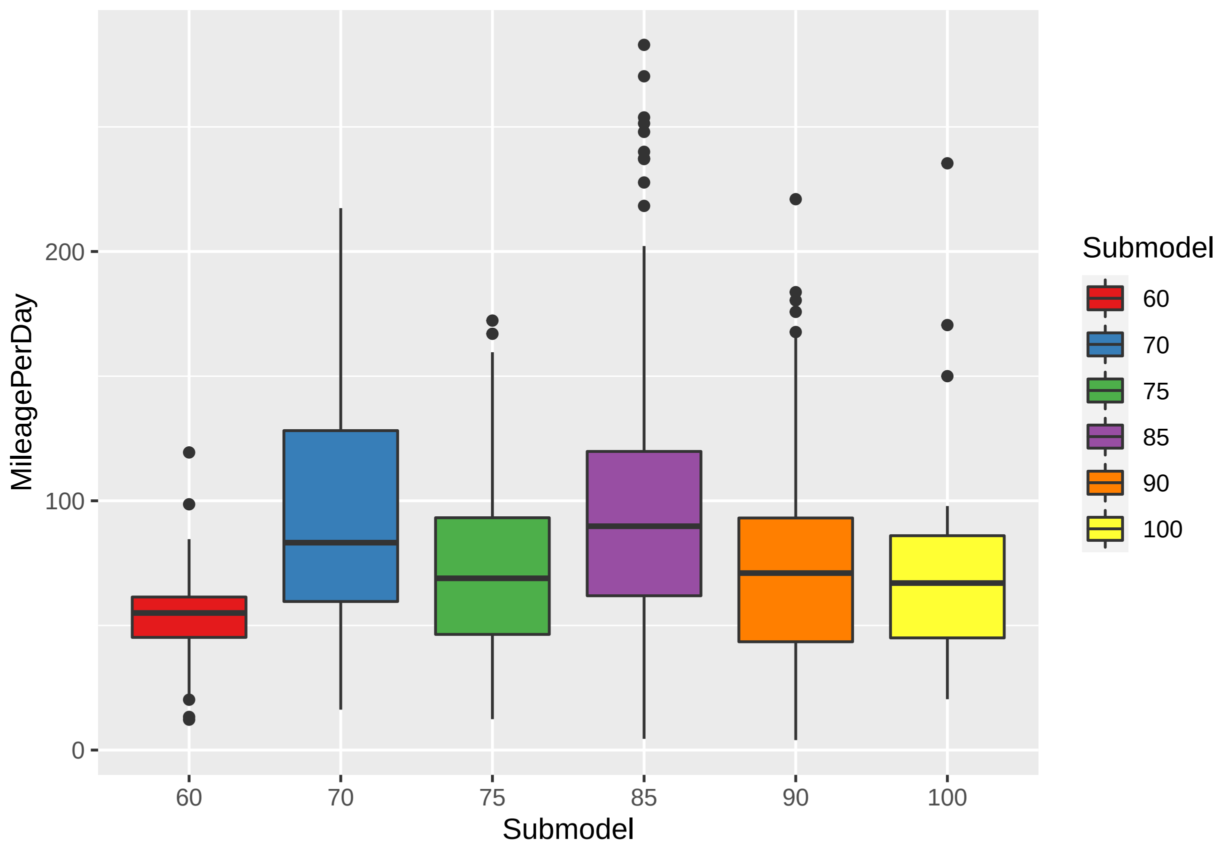
```
df1 <- data.frame(Location = df1$Group.1,
                  AgeInDays=df1$AgeInDays)

#I couldn't figure out how to get this to work!
#df2 <- aggregate(df,by=list(df$Location), FUN=median,na.rm=TRUE)

knitr::kable(df1)
```

Location	AgeInDays
Asia Pacific & Europe (excl UK)	543.1117
Canada	810.2766
UK	496.7273
USA	720.9220

Question 9

Which Submodels tend to be driven on the shortest daily drives?

```
library("viridis")
g1 <- ggplot(data = Tesla, aes(x=Submodel, y=MileagePerDay, fill=Submodel))+
  geom_boxplot()+
  scale_fill_brewer(palette= "Set1")
g1
```




```
df<- data.frame(Location = Tesla$Location,
           MileagePerDay=Tesla$MileagePerDay)
df1 <- aggregate(df,by=list(df$Location),FUN=mean,na.rm=TRUE)
df1<- data.frame(Location = df1$Group.1,
                 MileagePerDay= df1$MileagePerDay)
knitr::kable(df1)
```

Location	MileagePerDay
Asia Pacific & Europe (excl UK)	97.23495
Canada	73.43830
UK	65.17273
USA	53.54266