# BAN503 - Module 1 Assignment 3

## Silas, Kimberly

#install.packages("tidyverse")  
library(tidyverse)

## -- Attaching packages ------------------------------------ tidyverse 1.2.1 --

## v ggplot2 3.1.0 v purrr 0.3.2   
## v tibble 2.1.1 v dplyr 0.8.0.1  
## v tidyr 0.8.3 v stringr 1.4.0   
## v readr 1.3.1 v forcats 0.4.0

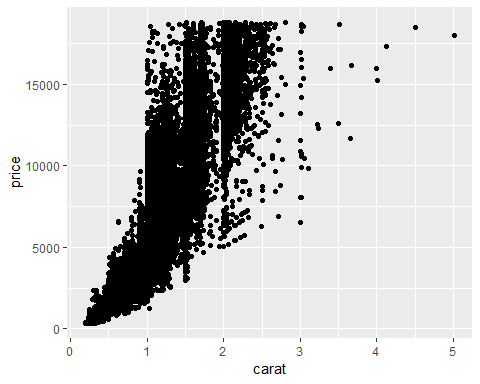
## -- Conflicts --------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

diamondsdata = diamonds  
diamonds

## # A tibble: 53,940 x 10  
## carat cut color clarity depth table price x y z  
## <dbl> <ord> <ord> <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl>  
## 1 0.23 Ideal E SI2 61.5 55 326 3.95 3.98 2.43  
## 2 0.21 Premium E SI1 59.8 61 326 3.89 3.84 2.31  
## 3 0.23 Good E VS1 56.9 65 327 4.05 4.07 2.31  
## 4 0.290 Premium I VS2 62.4 58 334 4.2 4.23 2.63  
## 5 0.31 Good J SI2 63.3 58 335 4.34 4.35 2.75  
## 6 0.24 Very Good J VVS2 62.8 57 336 3.94 3.96 2.48  
## 7 0.24 Very Good I VVS1 62.3 57 336 3.95 3.98 2.47  
## 8 0.26 Very Good H SI1 61.9 55 337 4.07 4.11 2.53  
## 9 0.22 Fair E VS2 65.1 61 337 3.87 3.78 2.49  
## 10 0.23 Very Good H VS1 59.4 61 338 4 4.05 2.39  
## # ... with 53,930 more rows

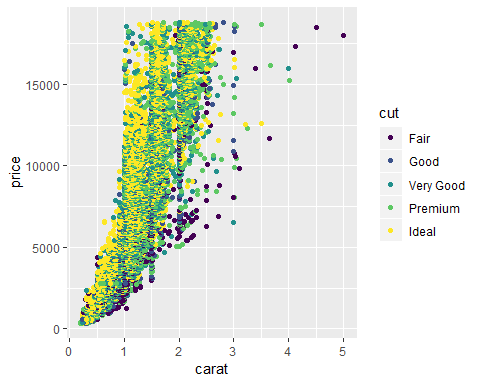
there are 10 variables and 53940 observations

library(tidyverse)  
ggplot(diamonds, aes(x=carat,y=price))+  
 geom\_point()

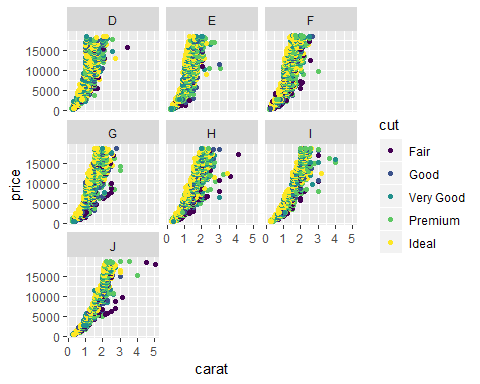


as the carat size increases the plots become more sparse as they are rare

library(tidyverse)  
ggplot(diamonds, aes(x=carat,y=price, color = cut))+  
 geom\_point()

 Adding in the cut shows the larger the carat the quality of the cut generally decreases. Even the most expensive diamond being a 5 carat only has a cut of fair, while the cheapest has a premium cut but is less than a carat.

library(tidyverse)  
ggplot(diamonds, aes(x=carat,y=price, color = cut))+  
 geom\_point()+  
 facet\_wrap(~ color)

 The highest carat with the best cut appears to be a color J. The colors E,F and G all have 3 as the highest carat. Price is not directly related to just one feature.

library(tidyverse)  
library(readr)  
inventory <- read\_csv("InventoryData.csv")

## Parsed with column specification:  
## cols(  
## `Item SKU` = col\_character(),  
## Store = col\_character(),  
## Supplier = col\_character(),  
## `Cost per Unit ($)` = col\_double(),  
## `On Hand` = col\_double(),  
## `Annual Demand` = col\_double()  
## )

inventoryA <- filter(inventory,Supplier=="A")

There are 3695 rows in this new data frame

inventoryA = mutate(inventoryA, OnHandRatio = `On Hand` / `Annual Demand`)  
inventoryA

## # A tibble: 3,695 x 7  
## `Item SKU` Store Supplier `Cost per Unit ~ `On Hand` `Annual Demand`  
## <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 0100 0034~ A 125. 159 1693  
## 2 011 0201~ A 12.3 173 1695  
## 3 0113 0317~ A 208. 166 1496  
## 4 0113 0802~ A 187. 157 1654  
## 5 0122 0034~ A 68.5 34 290  
## 6 0122 0201~ A 120. 77 680  
## 7 0122 0317~ A 56.6 133 1239  
## 8 013 0034~ A 19.1 28 277  
## 9 013 0201~ A 22.7 103 962  
## 10 013 0317~ A 1.13 29 297  
## # ... with 3,685 more rows, and 1 more variable: OnHandRatio <dbl>

The code takes the observation in the on hand column and divides it by the annual demand column creating a new column called OnHandRatio

library(dplyr)  
inventoryA[sapply(inventoryA, is.character)] <-lapply(inventoryA[sapply(inventoryA, is.character)], as.numeric)

## Warning in lapply(inventoryA[sapply(inventoryA, is.character)],  
## as.numeric): NAs introduced by coercion

avg\_cost = mutate(inventoryA, SKUAvgCost = `Cost per Unit ($)` / `Item SKU`)  
avg\_cost <- select(avg\_cost,"Item SKU", "SKUAvgCost")  
avg\_cost

## # A tibble: 3,695 x 2  
## `Item SKU` SKUAvgCost  
## <dbl> <dbl>  
## 1 100 1.25   
## 2 11 1.12   
## 3 113 1.84   
## 4 113 1.65   
## 5 122 0.561   
## 6 122 0.984   
## 7 122 0.464   
## 8 13 1.47   
## 9 13 1.75   
## 10 13 0.0869  
## # ... with 3,685 more rows

The most challenging concept to me was the that R studio is case sensitive. I currently spend the majority of my day working in a SQL environment that is not case sensitive and I find myself being somewhat lazy in that area and let the intelisense complete most of my table and column names. I have caught myself a few times here today and in my last class having to change the cases of column names in order to get the code to run.