Final Project

Brennan Black

4/12/2021

```
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   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

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

```
library(RMySQL)
```

```
## Loading required package: DBI
```

```
library(dplyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(gridExtra)
```

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

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

```
library(knitr)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(rsample)
library(recipes)
```

```
##
## Attaching package: 'recipes'
```

```
## The following object is masked from 'package:stringr':
##
##     fixed
```

```
## The following object is masked from 'package:stats':
##
##     step
```

```
library(kknn)
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 0.1.2 --
```

```
## v broom     0.7.4     v parsnip   0.1.5
## v dials     0.0.9     v tune      0.1.3
## v infer     0.5.4     v workflows 0.2.2
## v modeldata 0.1.0     v yardstick 0.0.8
```

```
## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x scales::discard()    masks purrr::discard()
## x dplyr::filter()      masks stats::filter()
## x recipes::fixed()     masks stringr::fixed()
## x dplyr::lag()         masks stats::lag()
## x yardstick::spec()    masks readr::spec()
## x recipes::step()      masks stats::step()
```

```
options(scipen=999)
```

Load Data

```
#db = dbConnect(MySQL(),
               #user = 'ofx_user',
               #password = 'TestyTester#2021',
               #dbname = 'OFX',
               #host = 'ballenger.wlu.edu')

#knitr::opts_knit$set(sql.max.print = -1)
```

Creating CSV Files

```
#cat <- dbSendQuery(db, 'SELECT * FROM category')
#category <- fetch(cat, n = -1)
#write.csv(category, "category_data.csv")

#pro <- dbSendQuery(db, 'SELECT * FROM product')
#product <- fetch(pro, n = -1)
#write.csv(product, "product_data.csv")

#ord_pro <- dbSendQuery(db, 'SELECT * FROM order_product')
#order_product <- fetch(ord_pro, n = -1)
#write.csv(order_product, "order_product_data.csv")
```

```
#cat <- dbSendQuery(db, 'SELECT * FROM category')
#category <- fetch(cat, n = -1)
#write.csv(category, "category_data.csv")

#buy <- dbSendQuery(db, 'SELECT * FROM buyer')
#buyer <- fetch(buy, n = -1)
#write.csv(buyer, "buyer_data.csv")

#loc <- dbSendQuery(db, 'SELECT * FROM location')
#location <- fetch(loc, n = -1)
#write.csv(location, "location_data.csv")

#ord <- dbSendQuery(db, 'SELECT * FROM orders')
#orders <- fetch(ord, n = -1)
#write.csv(orders, "orders_data.csv")
```

Creating Data Frames

```
category_data <- read.csv("data/category_data.csv")

product_data <- read.csv("data/product_data.csv")

order_product_data <- read.csv("data/order_product_data.csv")

category_data <- read.csv("data/category_data.csv")

buyer_data <- read.csv("data/buyer_data.csv")

location_data <- read.csv("data/location_data.csv")

orders_data <- read.csv("data/orders_data.csv")
```

Insight 1

Gross Profit Per Unit for Technology Subcategories

```
Tech_GP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Technology") %>%
  ggplot(mapping = aes(x = Sub_Category, y = Gross_Profit_Per_Unit)) +
  geom_boxplot() +
  labs(y = "GP per unit")
```

Gross Profit Per Unit for Office Supplies Subcategories

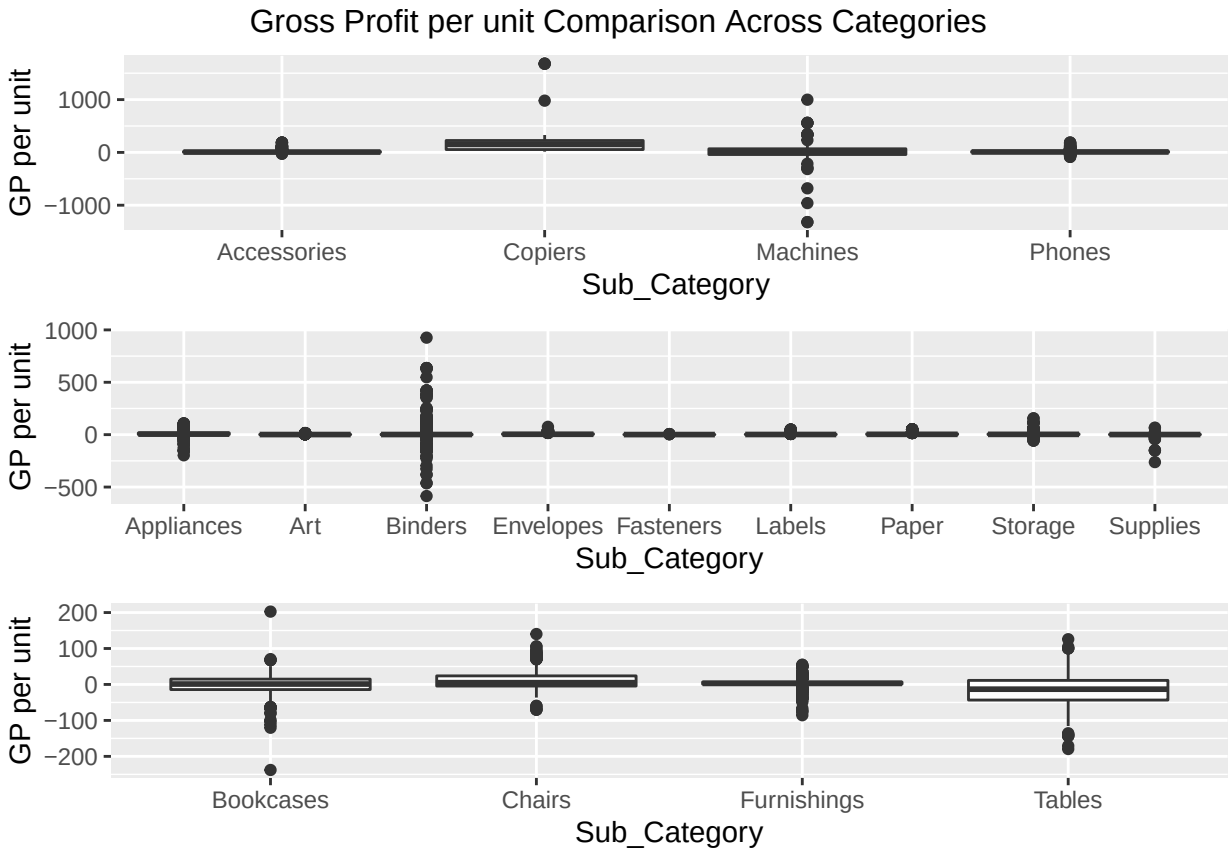
```
Office_Supplies_GP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Office Supplies") %>%
  ggplot(mapping = aes(x = Sub_Category, y = Gross_Profit_Per_Unit)) +
  geom_boxplot() +
  labs(y = "GP per unit")
```

Gross Profit Per Unit for Furniture Subcategories

```
Furniture_GP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Furniture") %>%
  ggplot(mapping = aes(x = Sub_Category, y = Gross_Profit_Per_Unit)) +
  geom_boxplot() +
  labs(y = "GP per unit")
```

Subcategory Gross Profit Comparison

```
grid.arrange(Tech_GP,
             Office_Supplies_GP,
             Furniture_GP, nrow = 3,
             top = "Gross Profit per unit Comparison Across Categories")
```



The plot above shows the gross profits per unit for each subcategory within each category. The plots show that there are a lot of products within the Machines, Binders, Supplies, Bookcases, and Tables subcategories that have a negative gross profit. We will investigate further by looking at total profit.

Insight 2

Technology Products Total Profits

```
Tech_TP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  group_by(Sub_Category) %>%
  filter(Category == "Technology") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit)) %>%
  ggplot(mapping = aes(x = Sub_Category, y = Total_Profit)) +
  geom_col()
```

Office Supply Products Total Profits

```
Office_Supplies_TP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  group_by(Sub_Category) %>%
```

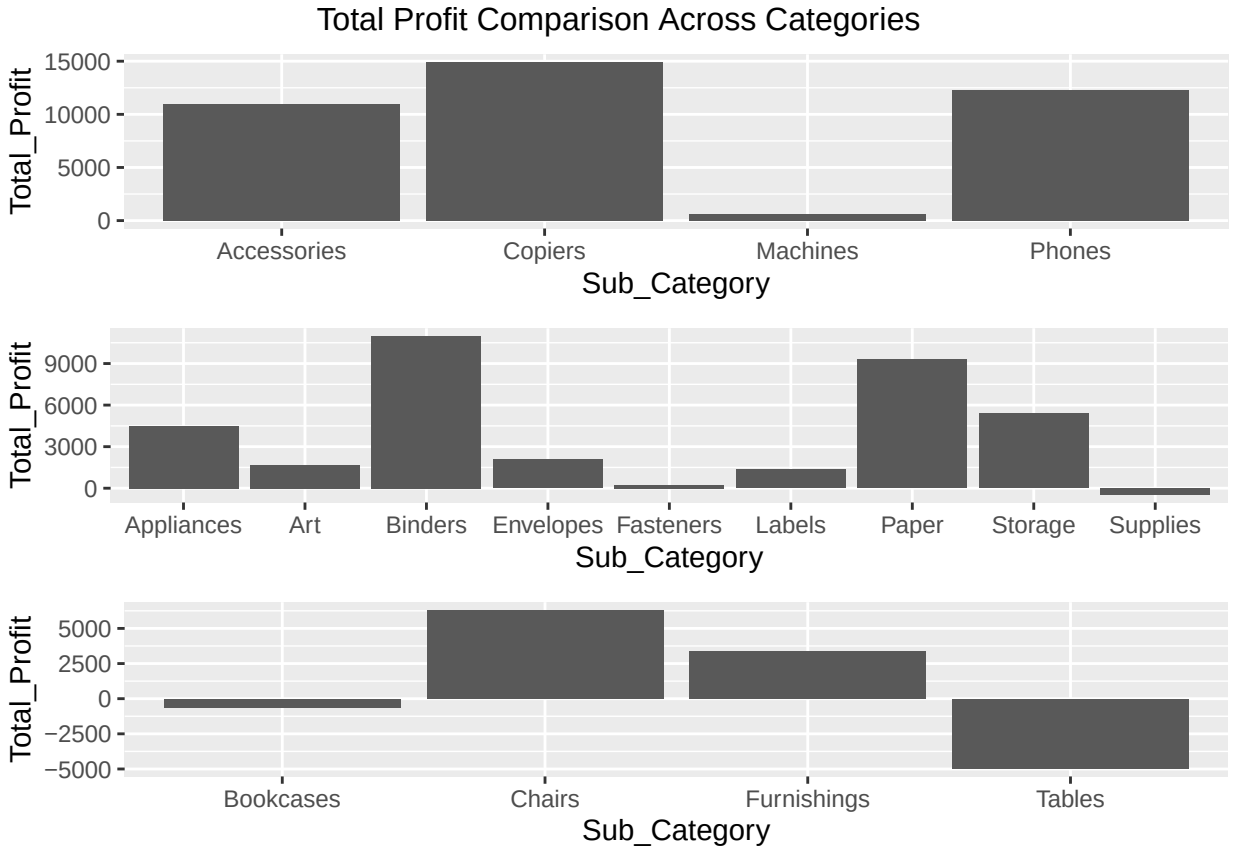
```
  filter(Category == "Office Supplies") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit)) %>%
  ggplot(mapping = aes(x = Sub_Category, y = Total_Profit)) +
  geom_col()
```

Furniture Product Categories Total Profits

```
Furniture_TP <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  group_by(Sub_Category) %>%
  filter(Category == "Furniture") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit)) %>%
  ggplot(mapping = aes(x = Sub_Category, y = Total_Profit)) +
  geom_col()
```

Subcategory Total Profit Comparison

```
grid.arrange(Tech_TP,
             Office_Supplies_TP,
             Furniture_TP,
             nrow = 3,
             top = "Total Profit Comparison Across Categories")
```



The plot shows that the Machines, Fasteners, Supplies, Bookcases, and Tables subcategories are all struggling in terms of profit. We will investigate these subcategories further, except for Fastners. This is because the first plot shows that all of the products in Fasteners are close to the mean of Gross Profit per unit. My objective is to tell the company to take out specific products that are dragging down profits; since all of the Fastener products are close to the mean, this getting rid of products will not be effective in improving profitability.

Insight 3

Average Machine Profits

```
machines_profit <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  summarise(Sub_Category_Profit = sum(Gross_Profit_Per_Unit)/length(Gross_Profit_Per_Unit))
```

Average Supplies Profits

```
supplies_profit <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  summarise(Sub_Category_Profit = sum(Gross_Profit_Per_Unit)/length(Gross_Profit_Per_Unit))
```

Average Bookcase Profits

```
bookcase_profit <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  summarise(Sub_Category_Profit = sum(Gross_Profit_Per_Unit)/length(Gross_Profit_Per_Unit))
```

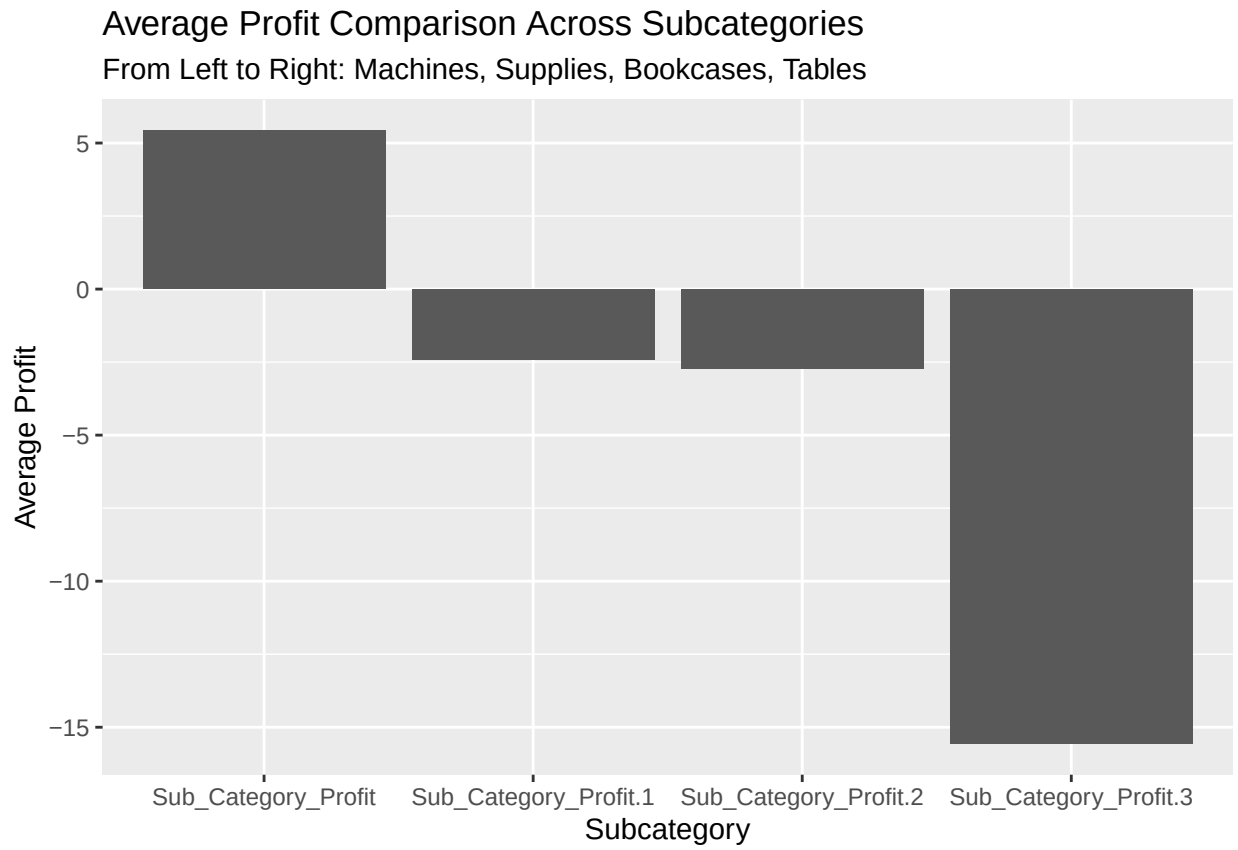
Average Table Profits

```
table_profit <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  summarise(Sub_Category_Profit = sum(Gross_Profit_Per_Unit)/length(Gross_Profit_Per_Unit))

Profit_Comparison <- data.frame(machines_profit, supplies_profit, bookcase_profit, table_profit)
```

```
Profit_Comparison_Longer <- Profit_Comparison %>%
  pivot_longer(1:4, names_to = "Sub_Category", values_to = "Average_Profit")
```

Profit Comparison

```
ggplot(data = Profit_Comparison_Longer, aes(x = Sub_Category, y = Average_Profit)) +
  geom_bar(stat = "identity") +
  labs(title = "Average Profit Comparison Across Subcategories",
       subtitle = "From Left to Right: Machines, Supplies, Bookcases, Tables",
       x = "Subcategory",
       y = "Average Profit")
```



This plot shows us the average profits for our selected subcategories. The Tables subcategory is struggling the most, which means that there are likely some poorly performing products that would improve profitability if they were cut. The same notion goes for the other subcategories as well but to a lesser degree.

Insight 4

Machine Subcategory Gross Margin

```
Machine_Gross_Margin <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
```

```
  inner_join(orders_data, by = "Order_ID") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  mutate(Gross_Margin = (Gross_Profit_Per_Unit/Unit_Price)) %>%
  distinct(Product_Name, Quantity, Gross_Margin)
```

```
Machine_Gross_Margin %>%
  filter(Gross_Margin < 0) %>%
  filter(Quantity > 6) %>%
  arrange(Gross_Margin)
```

```
##                                                Product_Name Quantity
## 1 Epson TM-T88V Direct Thermal Printer - Monochrome - Desktop        7
## 2                  Texas Instrument TI-15 Fraction Calculator        7
## 3                          Cisco 9971 IP Video Phone Charcoal        9
## 4                   Lexmark MX611dhe Monochrome Laser Printer        8
## 5         Xerox WorkCentre 6505DN Laser Multifunction Printer        7
##   Gross_Margin
## 1   -1.4333280
## 2   -0.8000000
## 3   -0.8000000
## 4   -0.1666673
## 5   -0.1000000
```

These are the products within the Machine subcategory that have high quantity and low profitability. These products should be cut.

Supplies Subcategory Gross Margin

```
Supplies_Gross_Margin <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  mutate(Gross_Margin = (Gross_Profit_Per_Unit/Unit_Price)) %>%
  distinct(Product_Name, Quantity, Gross_Margin)
```

```
Supplies_Gross_Margin %>%
  filter(Gross_Margin < 0) %>%
  filter(Quantity > 6) %>%
  arrange(Gross_Margin)
```

```
##                                       Product_Name Quantity Gross_Margin
## 1 Serrated Blade or Curved Handle Hand Letter Openers        7   -0.2375000
## 2  Martin Yale Chadless Opener Electric Letter Opener        7   -0.2250003
## 3                         Stiletto Hand Letter Openers        9   -0.2125000
```

These are the products within the Supplies subcategory that have high quantity and low profitability. These products should be cut.

Bookcase Subcategory Gross Margin

```
Bookcase_Gross_Margin <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  mutate(Gross_Margin = (Gross_Profit_Per_Unit/Unit_Price)) %>%
  distinct(Product_Name, Quantity, Gross_Margin)
```

```
Bookcase_Gross_Margin %>%
  filter(Gross_Margin < 0) %>%
  filter(Quantity > 6) %>%
  arrange(Gross_Margin)
```

```
##                                                 Product_Name Quantity
## 1                      O'Sullivan 2-Shelf Heavy-Duty Bookcases        7
## 2                                  Bestar Classic Bookcase        7
## 3                                Hon 4-Shelf Metal Bookcases        8
## 4       Atlantic Metals Mobile 4-Shelf Bookcases, Custom Colors        7
## 5                                  Bestar Classic Bookcase        7
## 6  Riverside Palais Royal Lawyers Bookcase, Royale Cherry Finish        7
## 7               O'Sullivan Living Dimensions 5-Shelf Bookcases        9
## 8         O'Sullivan Plantations 2-Door Library in Landvery Oak        7
## 9           Sauder Mission Library with Doors, Fruitwood Finish        7
## 10                     Sauder Camden County Collection Library       10
## 11                       Sauder Cornerstone Collection Library        8
## 12             Bush Birmingham Collection Bookcase, Dark Cherry        9
## 13                         DMI Eclipse Executive Suite Bookcases       10
##    Gross_Margin
## 1    -1.8000000
## 2    -1.7000000
## 3    -1.5000000
## 4    -1.3333333
## 5    -0.6200000
## 6    -0.5400009
## 7    -0.3235303
## 8    -0.2352948
## 9    -0.1911765
## 10   -0.1764706
## 11   -0.1625000
## 12   -0.1428571
## 13   -0.0125000
```

These are the products within the Bookcases subcategory that have high quantity and low profitability. These products should be cut.

Table Subcategory Gross Margin

```
Tables_Gross_Margin <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
```

```
  inner_join(orders_data, by = "Order_ID") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  mutate(Gross_Margin = (Gross_Profit_Per_Unit/Unit_Price)) %>%
  distinct(Product_Name, Quantity, Gross_Margin)
```

```
Tables_Gross_Margin %>%
  filter(Gross_Margin < 0) %>%
  filter(Quantity > 6) %>%
  arrange(Gross_Margin)
```

```
##                                                                  Product_Name
## 1              BoxOffice By Design Rectangular and Half-Moon Meeting Room Tables
## 2                                          Hon Rectangular Conference Tables
## 3                                               Balt Solid Wood Round Tables
## 4                                             Hon 94000 Series Round Tables
## 5                                                KI Adjustable-Height Table
## 6                                                     BPI Conference Tables
## 7                         Chromcraft Bull-Nose Wood Oval Conference Tables & Bases
## 8                         Chromcraft Bull-Nose Wood Oval Conference Tables & Bases
## 9                                       Iceberg OfficeWorks 42" Round Tables
## 10                                                    KI Conference Tables
## 11                                    Bevis Oval Conference Table, Walnut
## 12                          Bush Advantage Collection Racetrack Conference Table
## 13                                               KI Adjustable-Height Table
## 14                                                    BPI Conference Tables
## 15                                      Chromcraft Round Conference Tables
## 16   Riverside Furniture Oval Coffee Table, Oval End Table, End Table with Drawer
## 17   Riverside Furniture Oval Coffee Table, Oval End Table, End Table with Drawer
## 18                                  Bretford Rectangular Conference Table Tops
## 19                                     Bevis Round Bullnose 29" High Table Top
## 20                                               KI Adjustable-Height Table
## 21                                                    KI Conference Tables
## 22                          Office Impressions End Table, 20-1/2"H x 24"W x 20"D
## 23                                               Laminate Occasional Tables
## 24                                    Bevis Round Conference Table Top, X-Base
## 25         Barricks 18" x 48" Non-Folding Utility Table with Bottom Storage Shelf
## 26                                                    KI Conference Tables
## 27 Lesro Sheffield Collection Coffee Table, End Table, Center Table, Corner Table
## 28                                               Laminate Occasional Tables
##    Quantity Gross_Margin
## 1         7  -0.74000000
## 2         7  -0.66000000
## 3         7  -0.51666704
## 4         8  -0.51666667
## 5         9  -0.48000000
## 6         7  -0.45454545
## 7        13  -0.43333394
## 8         7  -0.43333394
## 9         9  -0.41666667
## 10        9  -0.40000000
## 11        8  -0.38333333
## 12        7  -0.36666667
## 13        7  -0.34545455
```

```
## 14       10  -0.33333333
## 15        8  -0.31666667
## 16        9  -0.30000000
## 17       12  -0.30000000
## 18        7  -0.28333333
## 19        7  -0.26666667
## 20        9  -0.23333333
## 21        7  -0.20000000
## 22        7  -0.20000000
## 23        8  -0.18571429
## 24        7  -0.11250000
## 25        7  -0.05714286
## 26       10  -0.05000000
## 27        7  -0.03750000
## 28        7  -0.03750000
```

These are the products within the Tables subcategory that have high quantity and low profitability. These products should be cut.

Insight 5

Total Machine Profits - Consumers

```
machines_profit_consumer <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Consumer") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

machines_profit_consumer
```

```
##   Total_Profit
## 1    -574.6595
```

Total Machine Profits - Corporate

```
machines_profit_corporate <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Corporate") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

machines_profit_corporate
```

```
##   Total_Profit
## 1     -93.3357
```

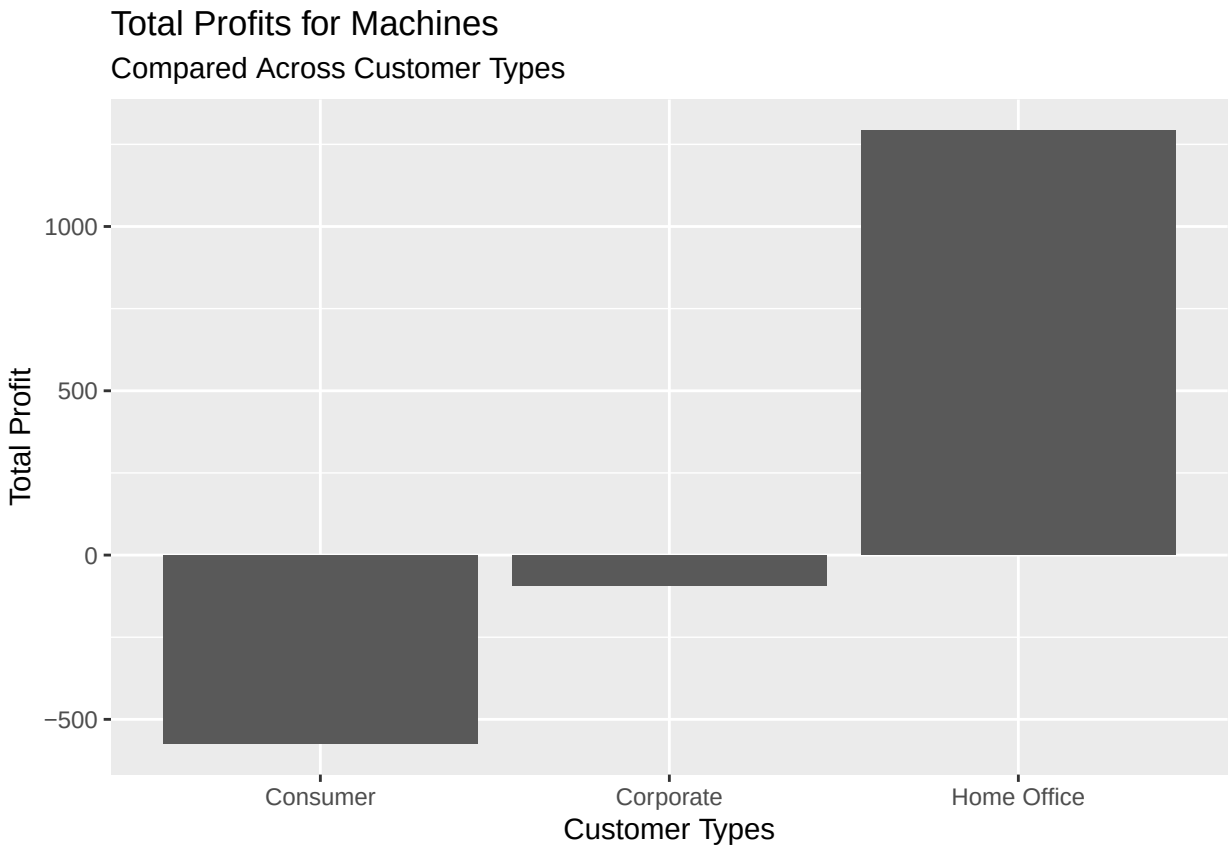
Total Machine Profits - Home Office

```
machines_profit_home_office <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Home Office") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

machines_profit_home_office
```

```
##   Total_Profit
## 1     1293.019
```

Machines Profit Comparison

```
machine_customer_type <- c("Consumer", "Corporate", "Home Office")

machine_total_profit <- c(-574.6595, -93.3357,
1293.019)

Profit_Comparison_Machines <- data.frame(machine_customer_type, machine_total_profit)

ggplot(data = Profit_Comparison_Machines, mapping = aes(x = machine_customer_type, y = machine_total_pr
  geom_col() +
  labs(title = "Total Profits for Machines",
       subtitle = "Compared Across Customer Types",
       x ="Customer Types",
       y = "Total Profit")
```



This plot breaks down total profit by the different customer segments. For the Machine subcategory, the Consumer and Corporate groups are not profitable. This means that some products should be cut from these segments.

Insight 6

Total Supplies Profits - Consumers

```
supplies_profit_consumer <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Consumer") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

supplies_profit_consumer
```

```
##   Total_Profit
## 1    -580.2834
```

Total Supplies Profits - Corporate

```
supplies_profit_corporate <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Corporate") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

supplies_profit_corporate
```

```
##   Total_Profit
## 1      85.0763
```

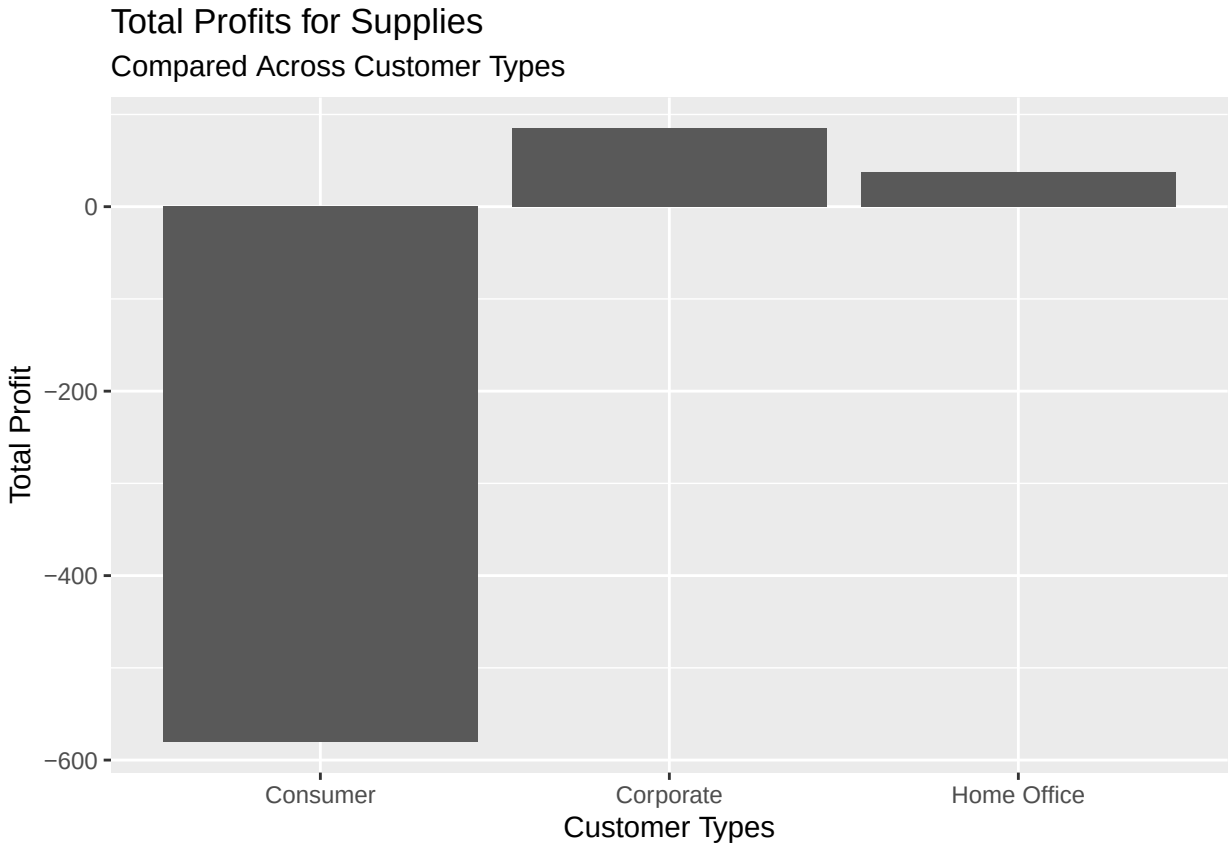
Total Supplies Profits - Home Office

```
supplies_profit_home_office <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Home Office") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

supplies_profit_home_office
```

```
##   Total_Profit
## 1      37.7482
```

Supplies Profit Comparison

```
supplies_customer_type <- c("Consumer", "Corporate", "Home Office")

supplies_total_profit <- c(-580.2834, 85.0763, 37.7482)

Profit_Comparison_Supplies <- data.frame(supplies_customer_type, supplies_total_profit)

ggplot(data = Profit_Comparison_Supplies, mapping = aes(x = supplies_customer_type, y = supplies_total_
  geom_col() +
  labs(title = "Total Profits for Supplies",
       subtitle = "Compared Across Customer Types",
       x ="Customer Types",
       y = "Total Profit")
```



This plot breaks down total profit by the different customer segments. For the Supplies subcategory, the Consume group is not profitable. This means that some products should be cut from this segment.

Insight 7

Total Bookcase Profits - Consumers

```
bookcase_profit_consumer <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Consumer") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

bookcase_profit_consumer
```

```
##   Total_Profit
## 1    -1048.531
```

Total Bookcase Profits - Corporate

```
bookcase_profit_corporate <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Corporate") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

bookcase_profit_corporate
```

```
##   Total_Profit
## 1     231.9954
```

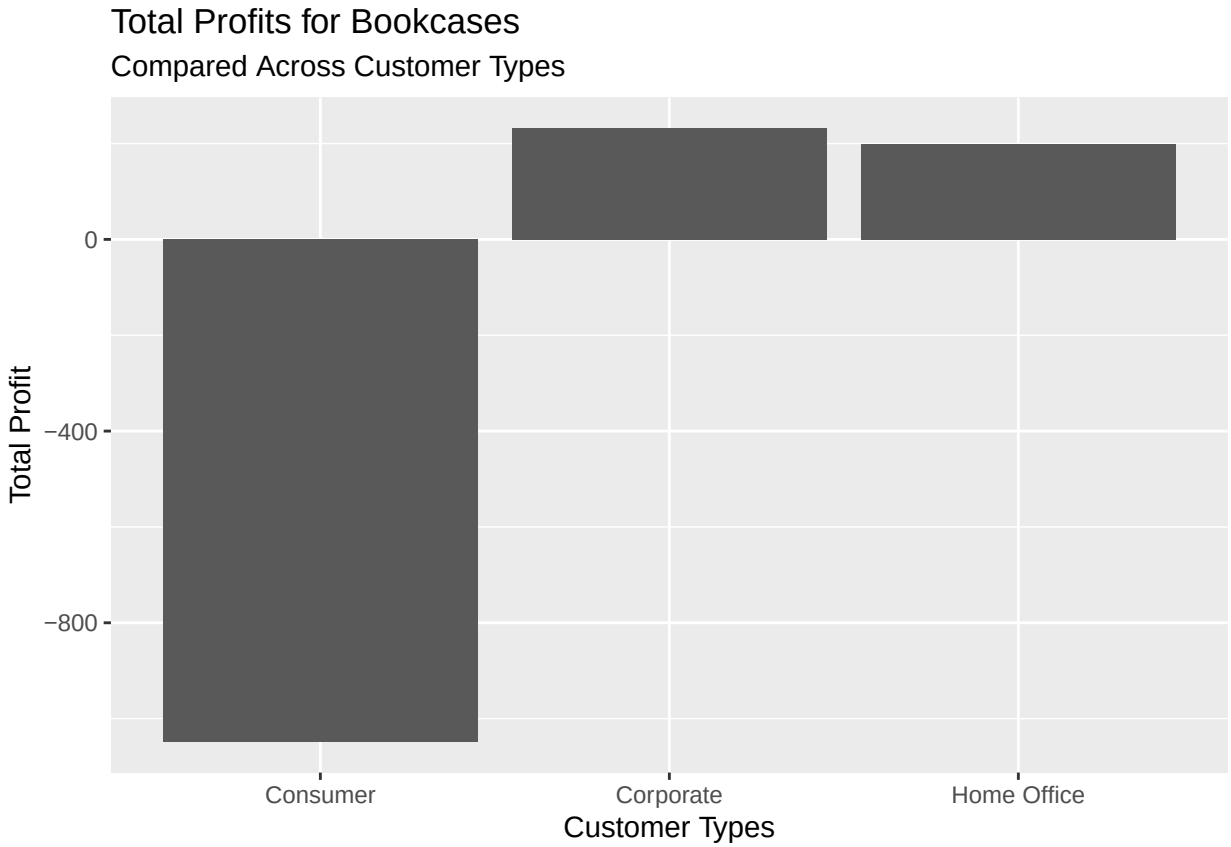
Total Bookcase Profits - Home Office

```
bookcase_profit_home_office <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Home Office") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

bookcase_profit_home_office
```

```
##   Total_Profit
## 1     199.2443
```

Bookcases Profit Comparison

```
bookcase_customer_type <- c("Consumer", "Corporate", "Home Office")

bookcase_total_profit <- c(-1048.531, 231.9954, 199.2443)

Profit_Comparison_Bookcases <- data.frame(bookcase_customer_type, bookcase_total_profit)

ggplot(data = Profit_Comparison_Bookcases, mapping = aes(x = bookcase_customer_type, y = bookcase_total_
  geom_col() +
  labs(title = "Total Profits for Bookcases",
       subtitle = "Compared Across Customer Types",
       x ="Customer Types",
       y = "Total Profit")
```



This plot breaks down total profit by the different customer segments. For the Bookcases subcategory, the Consumer group is not profitable. This means that some products should be cut from this segment.

Insight 8

Total Table Profits - Consumers

```
table_profit_consumer <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Consumer") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

table_profit_consumer
```

```
##   Total_Profit
## 1    -2759.934
```

Total Table Profits - Corporate

```
table_profit_corporate <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Corporate") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

table_profit_corporate
```

```
##   Total_Profit
## 1    -1411.077
```

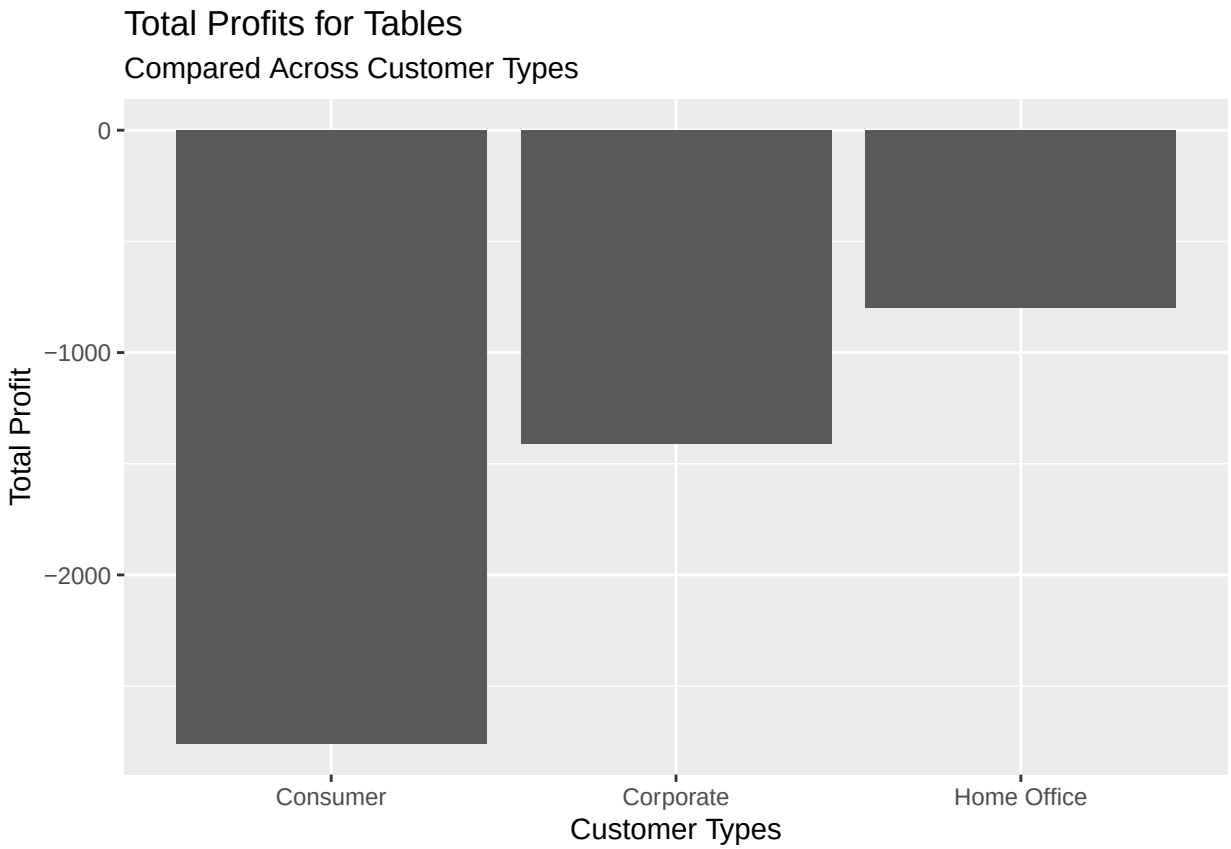
Total Table Profits - Home Office

```
table_profit_home_office <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Home Office") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  summarize(Total_Profit = sum(Gross_Profit_Per_Unit))

table_profit_home_office
```

```
##   Total_Profit
## 1    -797.3933
```

Tables Profit Comparison

```
table_customer_type <- c("Consumer", "Corporate", "Home Office")

table_total_profit <- c(-2759.934, -1411.077, -797.3933)

Profit_Comparison_Tables <- data.frame(table_customer_type, table_total_profit)

ggplot(data = Profit_Comparison_Tables, mapping = aes(x = table_customer_type, y = table_total_profit))
  geom_col() +
  labs(title = "Total Profits for Tables",
       subtitle = "Compared Across Customer Types",
       x ="Customer Types",
       y = "Total Profit")
```



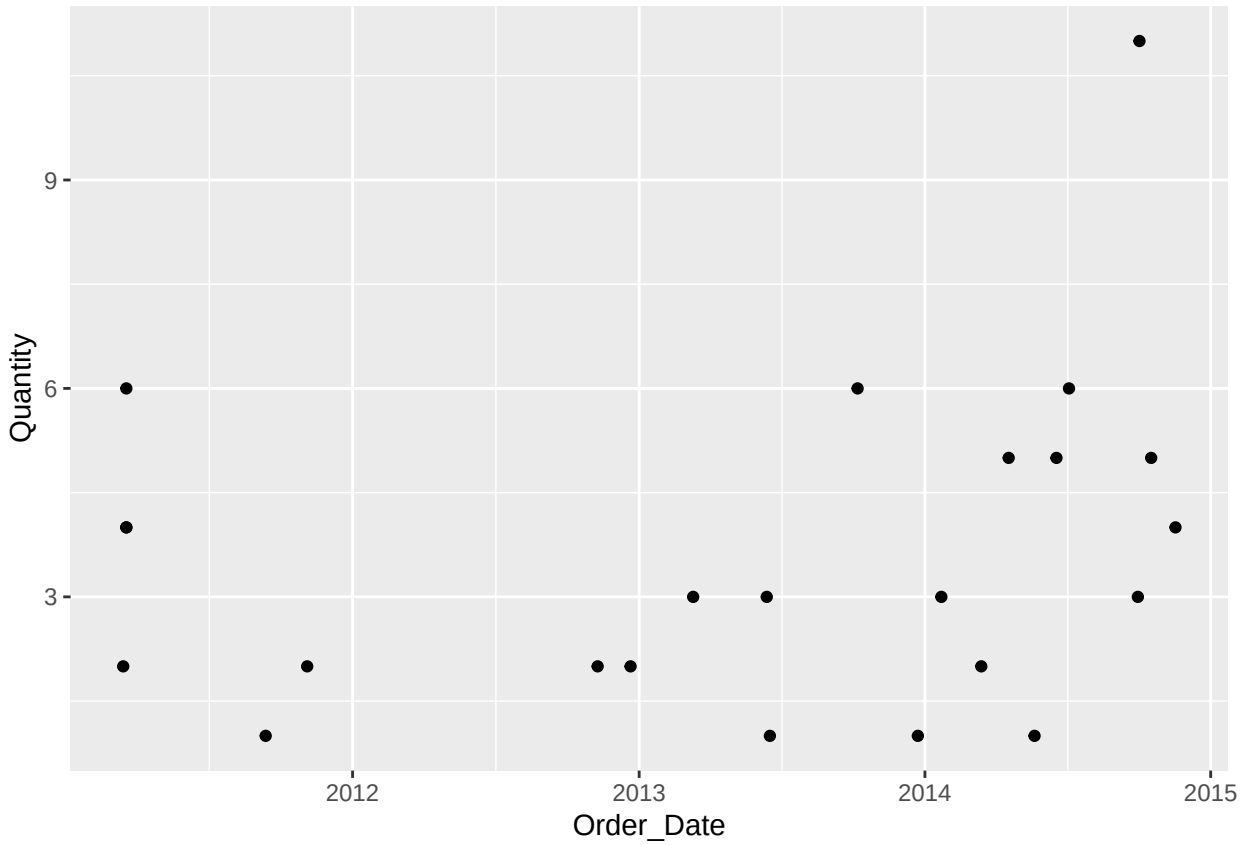
This plot breaks down total profit by the different customer segments. For the Supplies subcategory, none of the segments are profitable. This means that some products need to be cut from each of the segments, especially the Consumer segment.

Insight 9

```
orders_data <- orders_data %>%
  mutate(Order_Date = date(Order_Date))
```

Quantity of Machine Sales Over Time

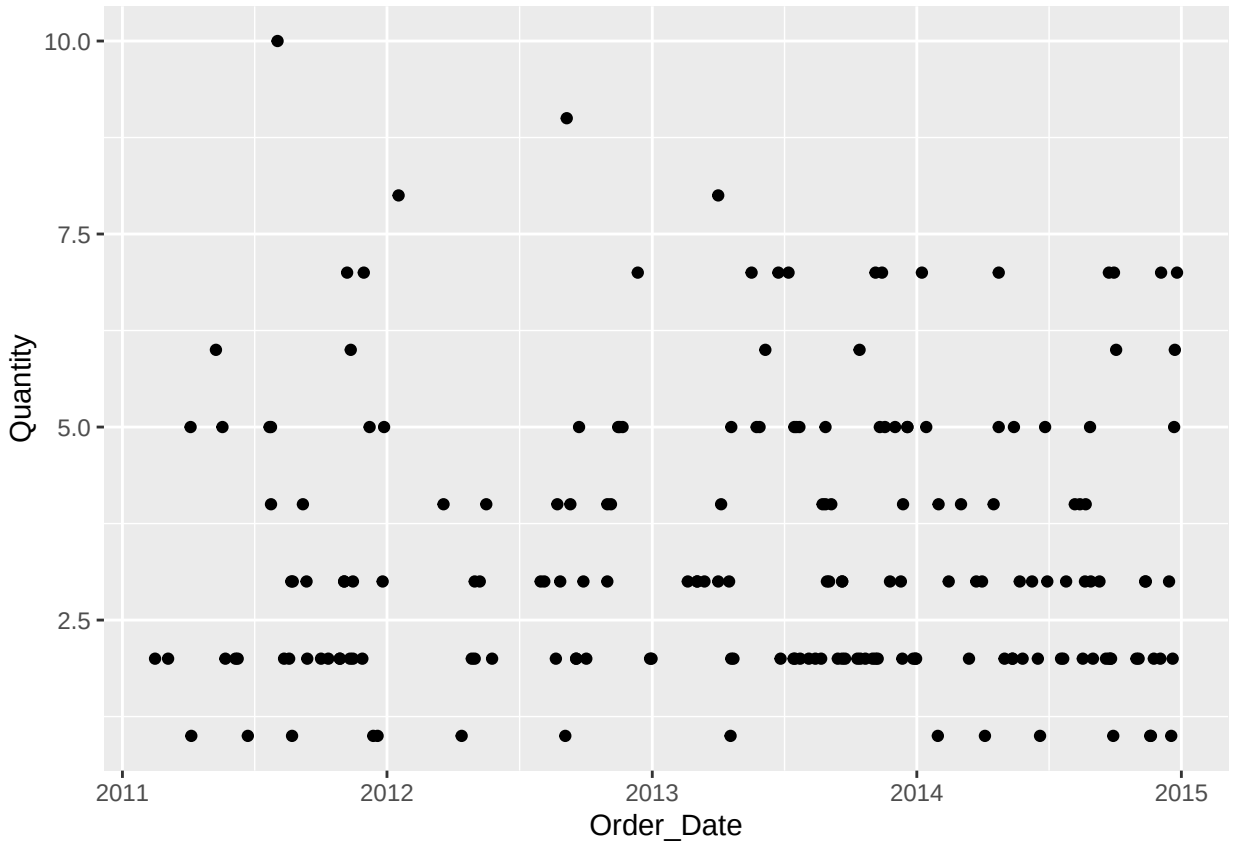
```
product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Type == "Home Office") %>%
  filter(Category == "Technology") %>%
  filter(Sub_Category == "Machines") %>%
  ggplot(aes(x = Order_Date, y = Quantity)) +
  geom_point()
```



This plot shows that, for the Machine subcategory, the average quantity per order stayed the same over the years but the number of orders increased in 2014 and 2015.

Quantity of Supplies Sales Over Time

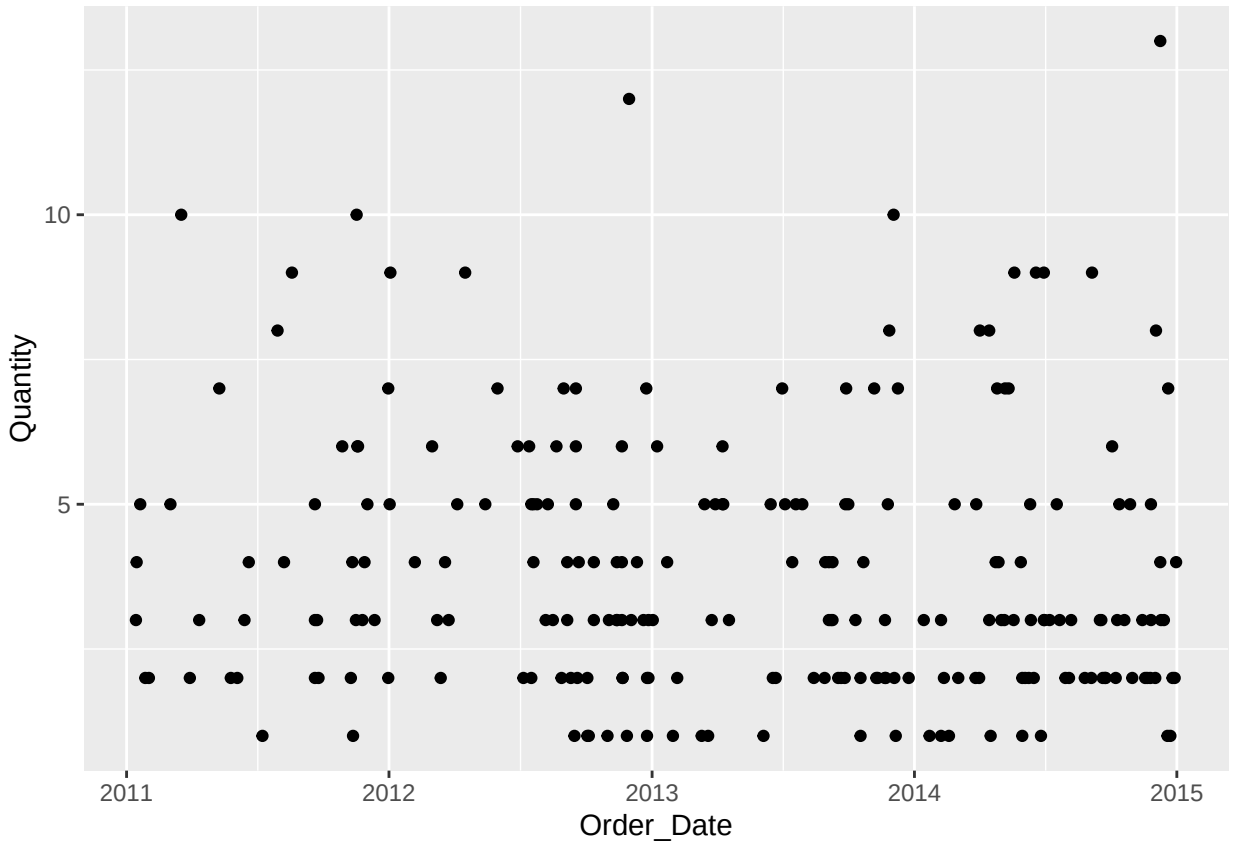
```
product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Category == "Office Supplies") %>%
  filter(Sub_Category == "Supplies") %>%
  ggplot(aes(x = Order_Date, y = Quantity)) +
  geom_point()
```



This plot shows that, for the Supplies subcategory, the average quantity per order stayed the same over the years but the number of orders increased in 2014 and 2015.

Quantity of Bookcases Sales Over Time

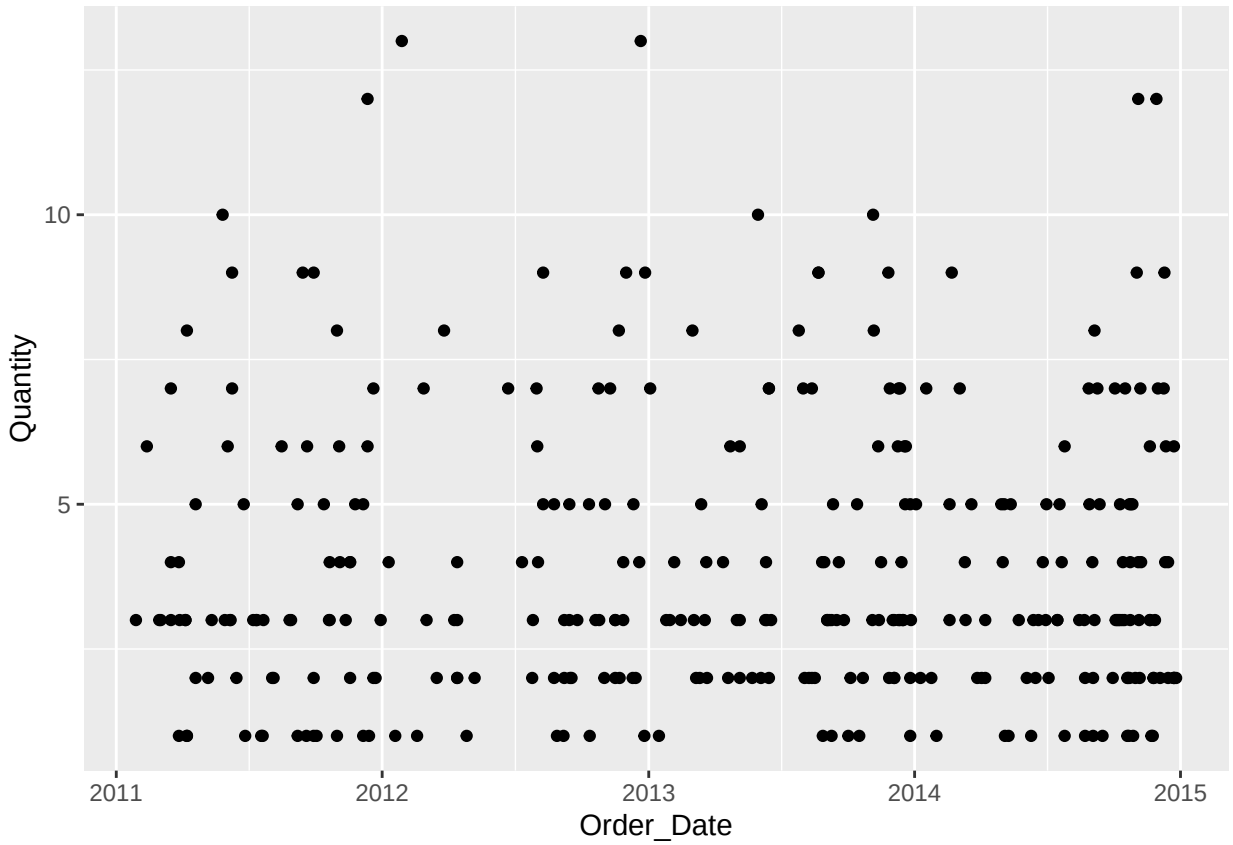
```
product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Bookcases") %>%
  ggplot(aes(x = Order_Date, y = Quantity)) +
  geom_point()
```



This plot shows that, for the Bookcases subcategory, the average quantity per order stayed the same over the years but the number of orders increased in 2014 and 2015.

Quantity of Table Sales Over Time

```
product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(category_data, by = "Sub_Category") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  filter(Category == "Furniture") %>%
  filter(Sub_Category == "Tables") %>%
  ggplot(aes(x = Order_Date, y = Quantity)) +
  geom_point()
```

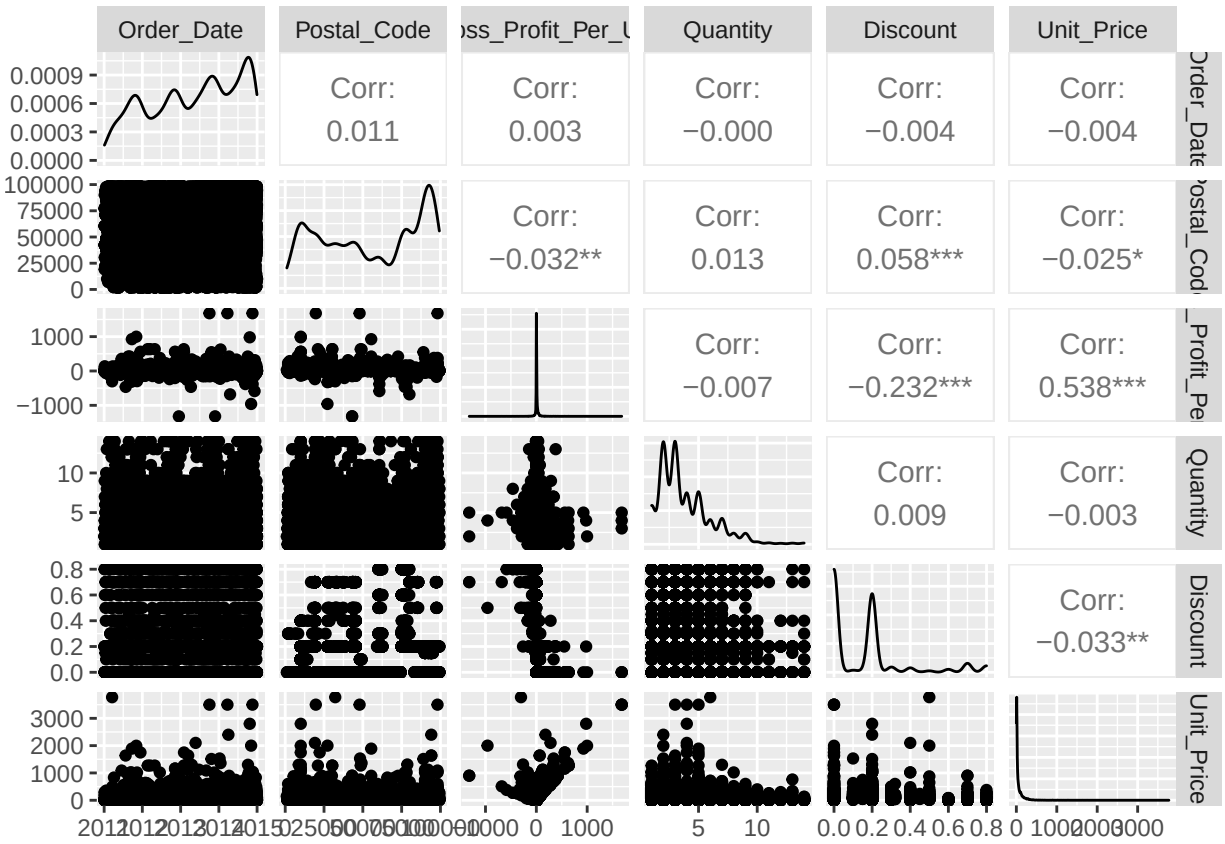


This plot shows that, for the Tables subcategory, the average quantity per order stayed the same over the years but the number of orders increased in 2014 and 2015.

Insight 10

```
ofx_correlation <- product_data %>%
  inner_join(order_product_data, by = "Product_ID") %>%
  inner_join(orders_data, by = "Order_ID") %>%
  inner_join(buyer_data, by = "Buyer_ID") %>%
  select(Order_Date, Postal_Code, Gross_Profit_Per_Unit, Quantity, Discount, Unit_Price)
```

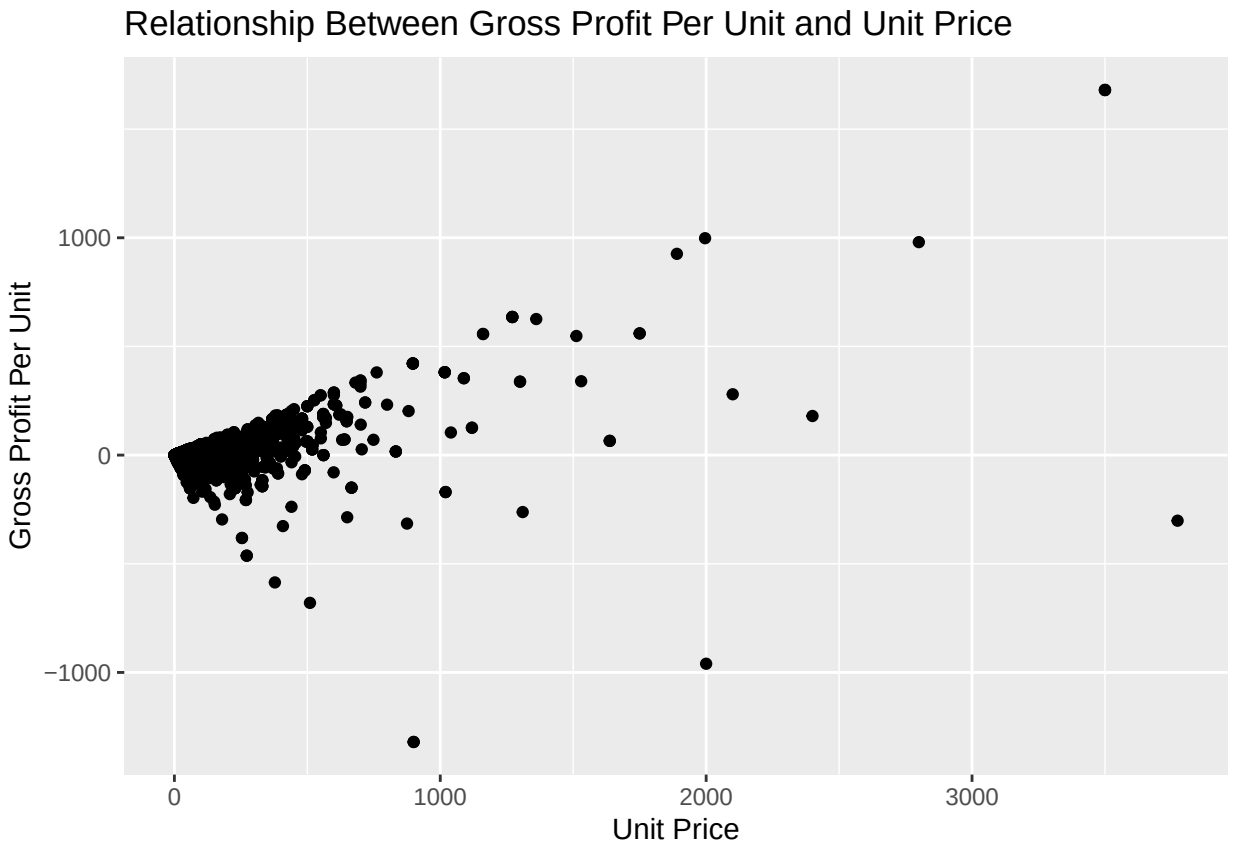
```
ofx_correlation %>%
  ggpairs()
```



The correlation matrix shows that most of the vectors are not correlated with each other. The one correlation that is somewhat strong is between Unit_Price and Gross_Profit_Per_Unit.

Insight 11

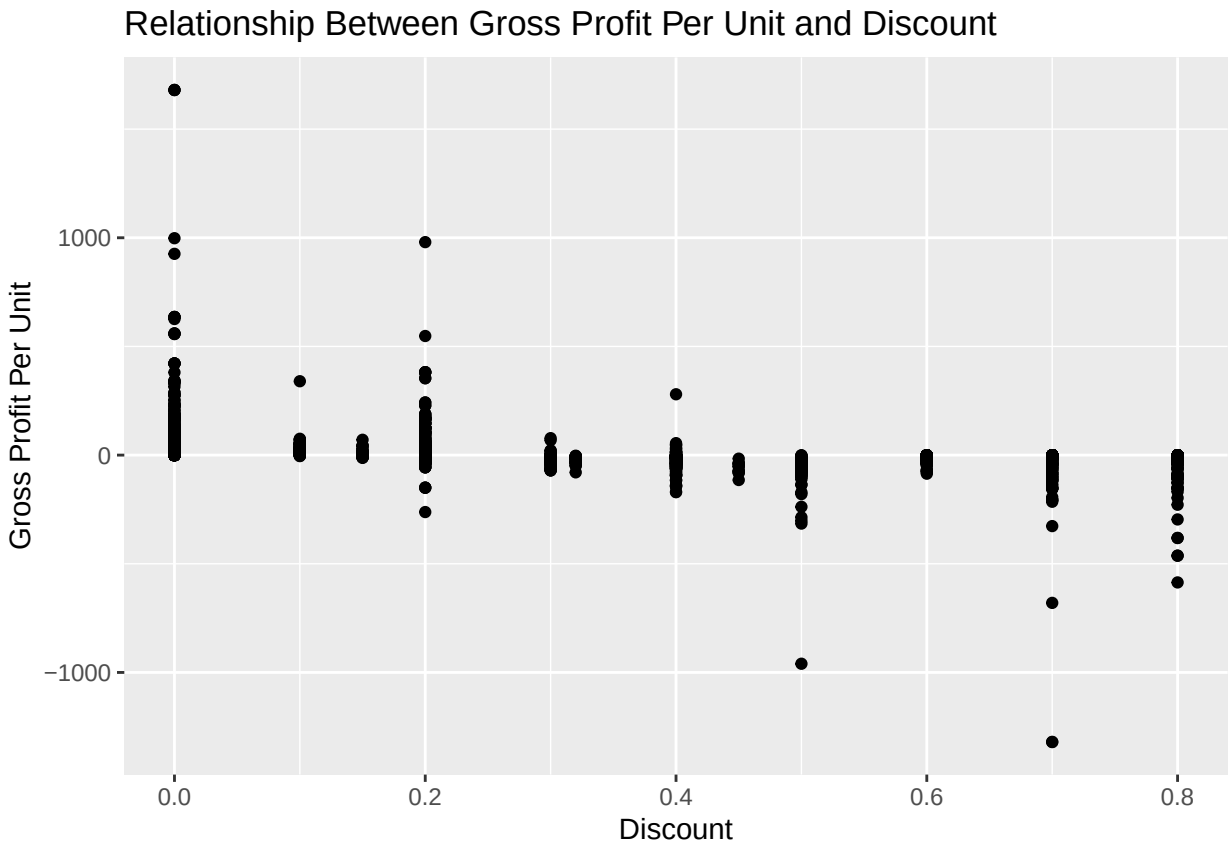
```
order_product_data %>%
  ggplot(aes(x = Unit_Price, y = Gross_Profit_Per_Unit)) +
  geom_point() +
  labs(title = "Relationship Between Gross Profit Per Unit and Unit Price",
       y = "Gross Profit Per Unit",
       x = "Unit Price")
```



This point plot explores the relationship between Gross Profit Per Unit and Unit Price. It is not very conclusive but there is some relationship between high price and high profit.

Insight 12

```
order_product_data %>%
  ggplot(aes(x = Discount, y = Gross_Profit_Per_Unit)) +
  geom_point() +
  labs(title = "Relationship Between Gross Profit Per Unit and Discount",
       y = "Gross Profit Per Unit",
       x = "Discount")
```



This point plot explores the relationship between Gross Profit Per Unit and Discount. It is not conclusive, which means that there is no relationship between high discount and high profit.

Part 2

Data Wrangling

```
bike_data <- read.csv("data/bike_share_hourly.csv")
```

```
bike_data <- bike_data %>%
  rename(date_day = dteday,
         year = yr,
         month = mnth,
         hour = hr,
         working_day = workingday,
         weather = weathersit,
         total_riders = cnt)
```

```
bike_data %>%
  summarise(across(everything(), ~ sum(is.na(.)))) %>%
  kable()
```

instant	date_day	season	year	month	hour	holiday	weekday	working_day	weather	temp	atemp	hum	windspeed	casual	registered	total_riders
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

There are no NA's in this data frame.

```
bike_data <- bike_data %>%
  mutate(season = factor(season, labels = c("Winter", "Spring", "Summer", "Fall"))) %>%
  mutate(holiday = factor(holiday, labels = c("no", "yes"))) %>%
  mutate(working_day = factor(working_day, labels = c("no", "yes"))) %>%
  mutate(year = factor(year, labels = c("2011", "2012"))) %>%
  mutate(weather = factor(weather)) %>%
  mutate(weekday = factor(weekday)) %>%
  mutate(month = factor(month))
```

```
bike_data<- bike_data %>%
  mutate(time_of_day = case_when(
    hour >= 0 & hour <= 6 ~ 1,
    hour >= 7 & hour <= 12 ~ 2,
    hour >= 13 & hour <= 18 ~ 3,
    hour >= 19 & hour <= 23 ~ 4
    ))
```

```
bike_data<- bike_data %>%
  mutate(time_of_day = factor(time_of_day))
```

Question 1

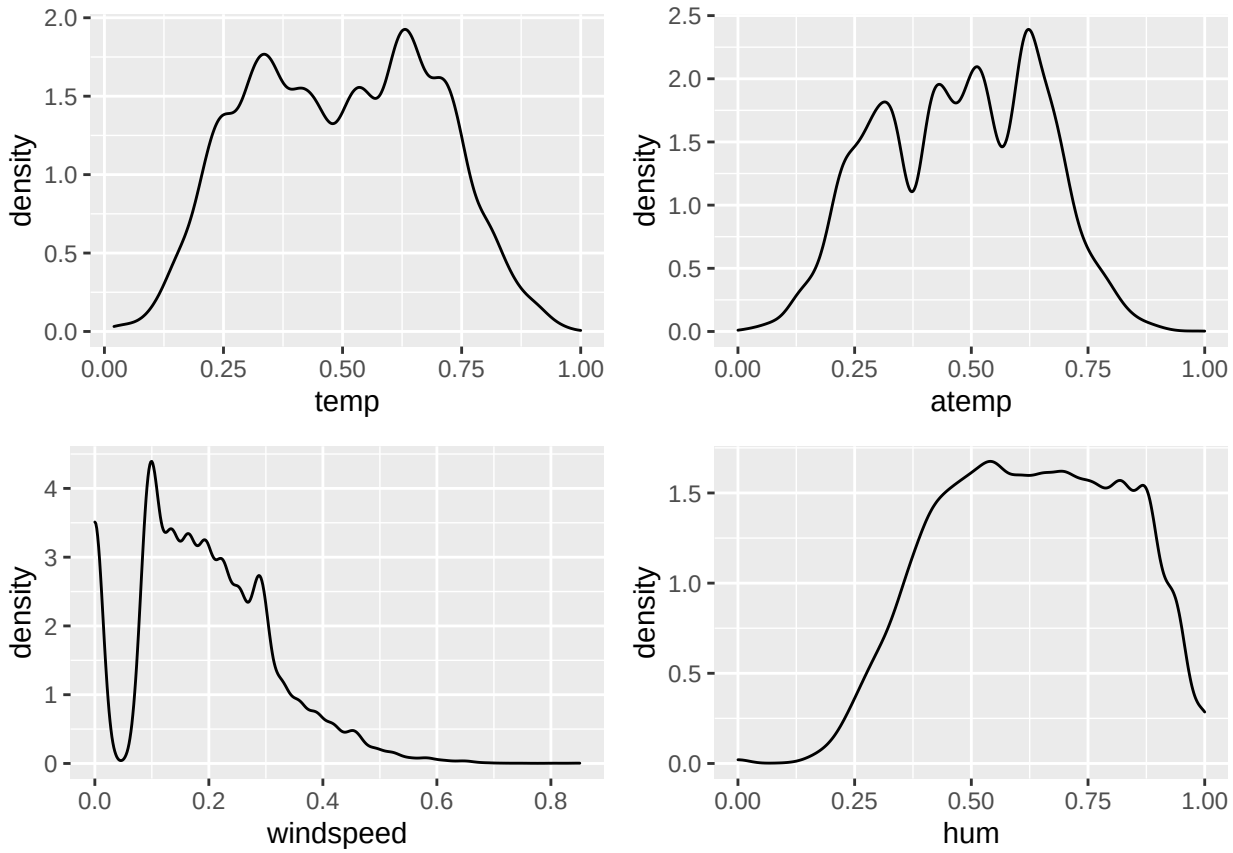
```
temp_density <- bike_data %>%
  ggplot(aes(temp)) +
  geom_density()
```

```
atemp_density <- bike_data %>%
  ggplot(aes(atemp)) +
  geom_density()
```

```
humidity_density <- bike_data %>%
  ggplot(aes(hum)) +
  geom_density()
```

```
windspeed_density <- bike_data %>%
  ggplot(aes(windspeed)) +
  geom_density()
```

```
grid.arrange(temp_density, atemp_density, windspeed_density, humidity_density, ncol = 2, nrow = 2)
```



The variables are suitable for the kNN classification algorithm because their values are all between 0 and 1. This means that I do not need to normalize or standardize the variables.

Question 2

```
glimpse(bike_data)
```

```
## Rows: 17,379
## Columns: 18
## $ instant      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17~
## $ date_day     <chr> "2011-01-01", "2011-01-01", "2011-01-01", "2011-01-01", "~
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ year         <fct> 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 201~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ hour         <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,~
## $ holiday      <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weekday      <fct> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weather      <fct> 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, ~
## $ temp         <dbl> 0.24, 0.22, 0.22, 0.24, 0.24, 0.24, 0.22, 0.20, 0.24, 0.3~
## $ atemp        <dbl> 0.2879, 0.2727, 0.2727, 0.2879, 0.2879, 0.2576, 0.2727, 0~
## $ hum          <dbl> 0.81, 0.80, 0.80, 0.75, 0.75, 0.75, 0.80, 0.86, 0.75, 0.7~
## $ windspeed    <dbl> 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0896, 0.0000, 0~
## $ casual       <int> 3, 8, 5, 3, 0, 0, 2, 1, 1, 8, 12, 26, 29, 47, 35, 40, 41,~
## $ registered   <int> 13, 32, 27, 10, 1, 1, 0, 2, 7, 6, 24, 30, 55, 47, 71, 70,~
## $ total_riders <int> 16, 40, 32, 13, 1, 1, 2, 3, 8, 14, 36, 56, 84, 94, 106, 1~
```

```
## $ time_of_day  <fct> 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, ~
```

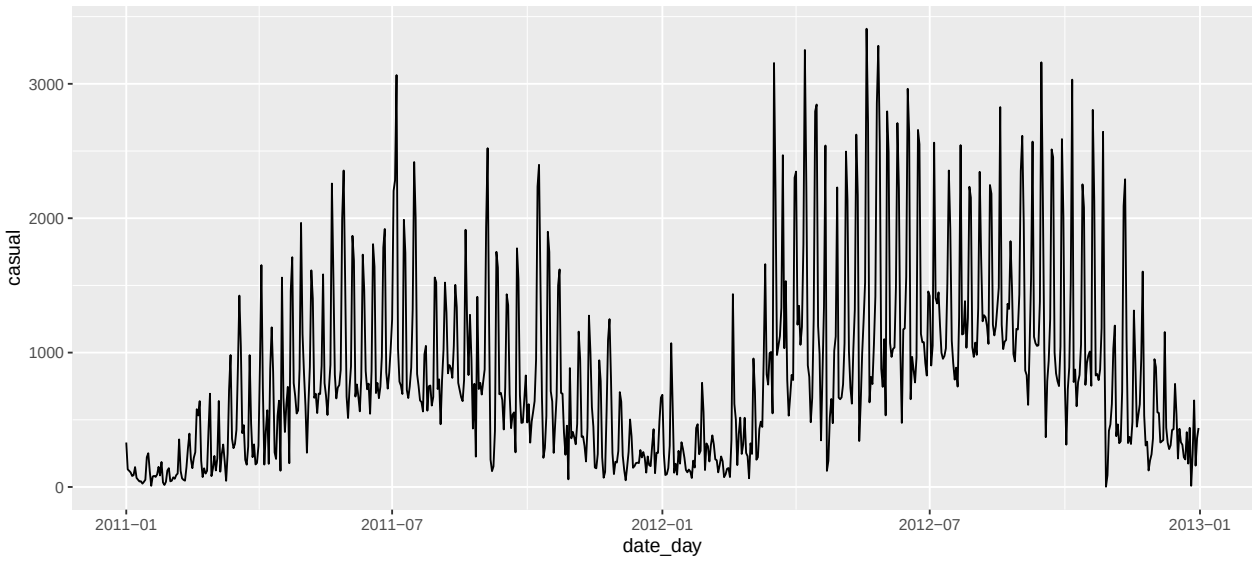
```
bike_data <- bike_data %>%
  mutate(date_day = date(date_day)) %>%
  mutate(total_riders = as.numeric(total_riders))
```

```
glimpse(bike_data)
```

```
## Rows: 17,379
## Columns: 18
## $ instant      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17~
## $ date_day     <date> 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-~
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ year         <fct> 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 201~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ hour         <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,~
## $ holiday      <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weekday      <fct> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weather      <fct> 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, ~
## $ temp         <dbl> 0.24, 0.22, 0.22, 0.24, 0.24, 0.24, 0.22, 0.20, 0.24, 0.3~
## $ atemp        <dbl> 0.2879, 0.2727, 0.2727, 0.2879, 0.2879, 0.2576, 0.2727, 0~
## $ hum          <dbl> 0.81, 0.80, 0.80, 0.75, 0.75, 0.75, 0.80, 0.86, 0.75, 0.7~
## $ windspeed    <dbl> 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0896, 0.0000, 0~
## $ casual       <int> 3, 8, 5, 3, 0, 0, 2, 1, 1, 8, 12, 26, 29, 47, 35, 40, 41,~
## $ registered   <int> 13, 32, 27, 10, 1, 1, 0, 2, 7, 6, 24, 30, 55, 47, 71, 70,~
## $ total_riders <dbl> 16, 40, 32, 13, 1, 1, 2, 3, 8, 14, 36, 56, 84, 94, 106, 1~
## $ time_of_day  <fct> 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, ~
```

Casual

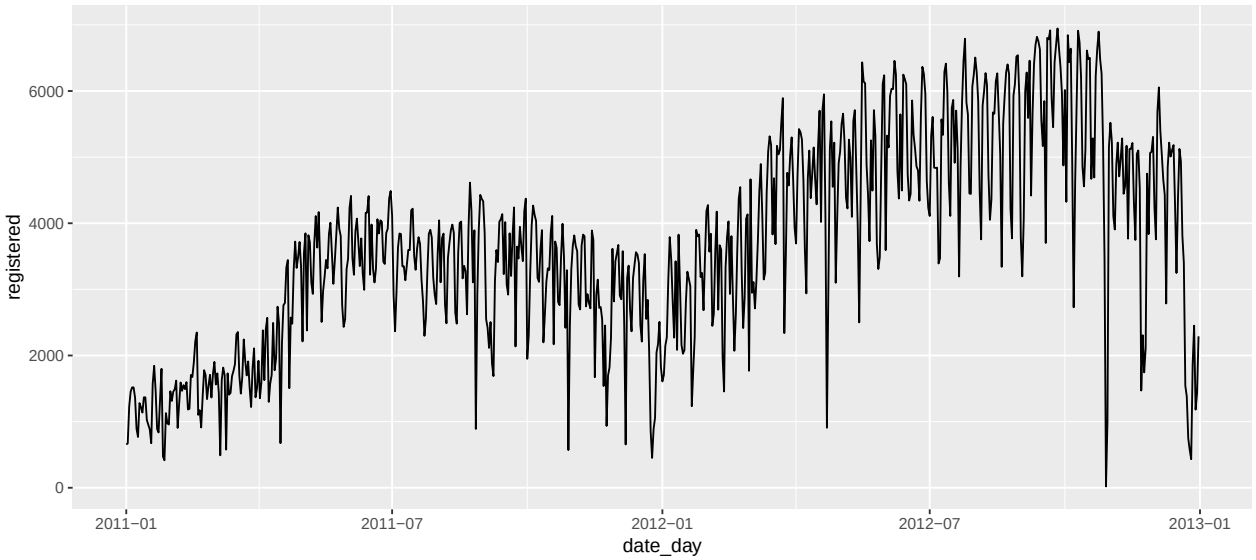
```
casual_riders <- bike_data %>%
  select(casual, date_day) %>%
  group_by(date_day) %>%
  summarise(casual = sum(casual)) %>%
  ungroup() %>%
  ggplot(aes(x = date_day, y = casual)) +
  geom_line()

casual_riders
```



Casual ridership increased from 2011 to 2012 but not substantially.

Registered

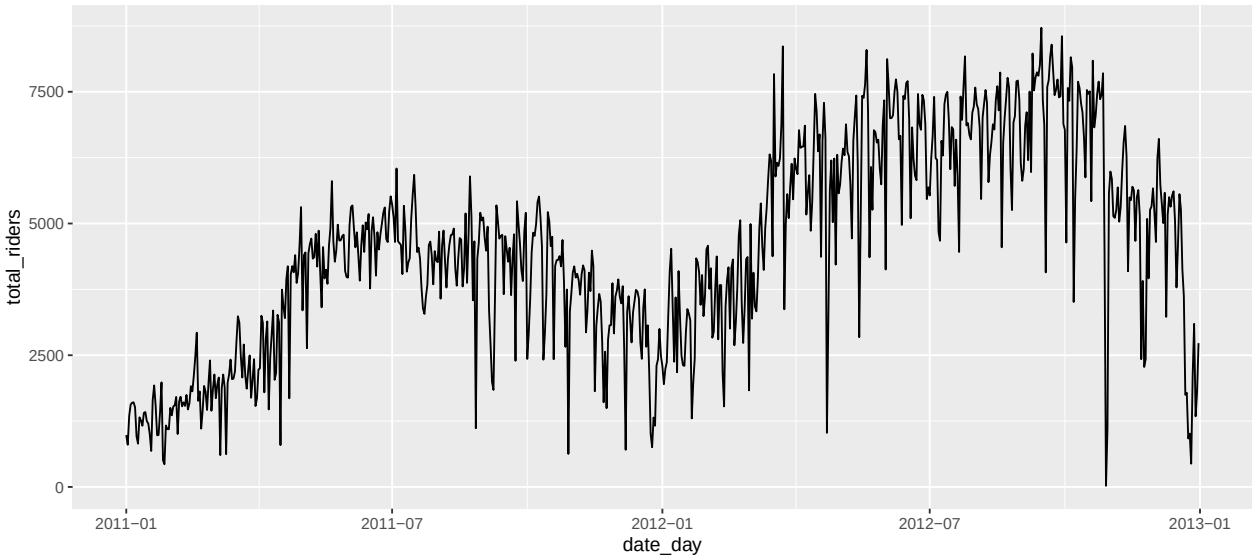
```
registered_riders <- bike_data %>%
  select(registered, date_day) %>%
  group_by(date_day) %>%
  summarise(registered = sum(registered)) %>%
  ungroup() %>%
  ggplot(aes(x = date_day, y = registered)) +
  geom_line()

registered_riders
```



Registered ridership increased substantially from 2011 to 2012.

Total Riders

```
total_riders_line <- bike_data %>%
  select(total_riders, date_day) %>%
  group_by(date_day) %>%
  summarise(total_riders = sum(total_riders)) %>%
  ungroup() %>%
  ggplot(aes(x = date_day, y = total_riders)) +
  geom_line()

total_riders_line
```



Registered ridership increased substantially from 2011 to 2012.

The Model - 2011 and 2012 Data

```
kn_bike_data <- bike_data %>%
  select(-instant, -date_day, -year, -hour, -registered, -casual)
```

I dropped the date_day variable since it will not be used as a predictor and also because we have the season and month columns.

In regard to the fact that ridership grew substantially in 2012 and that management thinks this will be the case going forward, I will make a model containing data from both 2011 and 2012 and another model with just 2012 data.

Split the data

```
set.seed(2021)

bike_split <- initial_split(kn_bike_data, prop = .75)

bike_split_train <- training(bike_split)
```

```
bike_split_test <- testing(bike_split)

bike_split_train %>%
  glimpse()
```

```
## Rows: 13,035
## Columns: 12
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ holiday      <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weekday      <fct> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 0, 0, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weather      <fct> 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 2, 2, 2, 2, ~
## $ temp         <dbl> 0.22, 0.22, 0.24, 0.24, 0.24, 0.22, 0.20, 0.24, 0.38, 0.4~
## $ atemp        <dbl> 0.2727, 0.2727, 0.2879, 0.2879, 0.2576, 0.2727, 0.2576, 0~
## $ hum          <dbl> 0.80, 0.80, 0.75, 0.75, 0.75, 0.80, 0.86, 0.75, 0.76, 0.7~
## $ windspeed    <dbl> 0.0000, 0.0000, 0.0000, 0.0000, 0.0896, 0.0000, 0.0000, 0~
## $ total_riders <dbl> 40, 32, 13, 1, 1, 2, 3, 8, 36, 94, 106, 110, 67, 35, 37, ~
## $ time_of_day  <fct> 1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 1, 1, ~
```

```
bike_split_test %>%
  glimpse()
```

```
## Rows: 4,344
## Columns: 12
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ holiday      <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weekday      <fct> 6, 6, 6, 6, 6, 6, 6, 0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, yes, ~
## $ weather      <fct> 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ temp         <dbl> 0.24, 0.32, 0.36, 0.42, 0.42, 0.40, 0.40, 0.46, 0.46, 0.3~
## $ atemp        <dbl> 0.2879, 0.3485, 0.3333, 0.4242, 0.4242, 0.4091, 0.4091, 0~
## $ hum          <dbl> 0.81, 0.76, 0.81, 0.77, 0.82, 0.87, 0.87, 0.94, 0.94, 0.8~
## $ windspeed    <dbl> 0.0000, 0.0000, 0.2836, 0.2836, 0.2985, 0.2537, 0.1940, 0~
## $ total_riders <dbl> 16, 14, 56, 84, 93, 36, 34, 6, 3, 53, 76, 30, 31, 5, 1, 6~
## $ time_of_day  <fct> 1, 2, 2, 2, 3, 4, 4, 1, 1, 2, 3, 4, 4, 1, 1, 2, 2, 2, 2, ~
```

Prep the data

```
bike_recipe <- recipe(total_riders ~ ., data = bike_split_train) %>%
  step_dummy(holiday, working_day, one_hot = FALSE) %>%
  step_dummy(season, weekday, weather, time_of_day, month, one_hot = TRUE) %>%
  prep()
```

Apply recipe to training data

```
bike_juiced <- juice(bike_recipe)
```

Bake the test data

```
# Apply the same recipe to the test data
bike_split_test_baked <- bake(bike_recipe, new_data = bike_split_test)
```

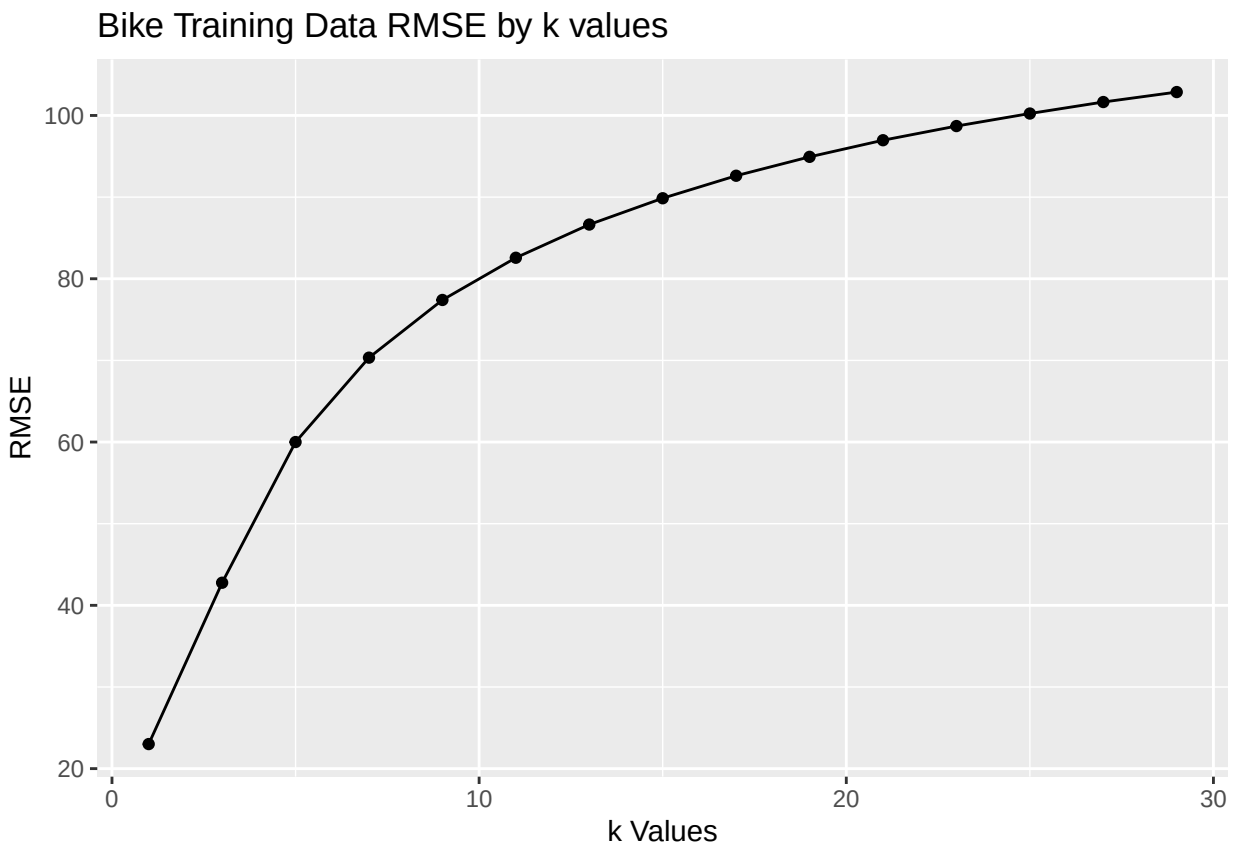
Test multiple values of k

```
# Optimal K value usually found is the square root of N, where N is the total number of samples
possible_optimal_k  <- sqrt(nrow(bike_data))

# Set initial value of i

begin <- 1

for (i in seq(begin, 30, 2)) {

# Make a knn spec
knn_spec <- nearest_neighbor(neighbor = i) %>%
  set_engine("kknn") %>%
  set_mode("regression")

# Use the knn spec to fit the prepared training data
knn_fit <- knn_spec %>%
  fit(total_riders ~., data = bike_juiced)

# Evaluate the model with the training data
train_knn_fit <-  knn_fit %>%
    predict(bike_juiced) %>%
    bind_cols(bike_juiced) %>%
    metrics(truth = total_riders, estimate = .pred) %>%
    mutate(k = i)

# Evaluate the model with the testing data
test_knn_fit <- knn_fit %>%
    predict(bike_split_test_baked) %>%
    bind_cols(bike_split_test_baked) %>%
    metrics(truth = total_riders, estimate = .pred) %>%
    mutate(k = i)

# Create summary performance evaluation dataframes
if (i == begin) {
summary_train_knn_fit <- train_knn_fit
summary_test_knn_fit <-  test_knn_fit
} else {
summary_train_knn_fit <- bind_rows(summary_train_knn_fit, train_knn_fit)
summary_test_knn_fit <- bind_rows(summary_test_knn_fit, test_knn_fit)

}

}

# Output summary performance dataframesrm()
```

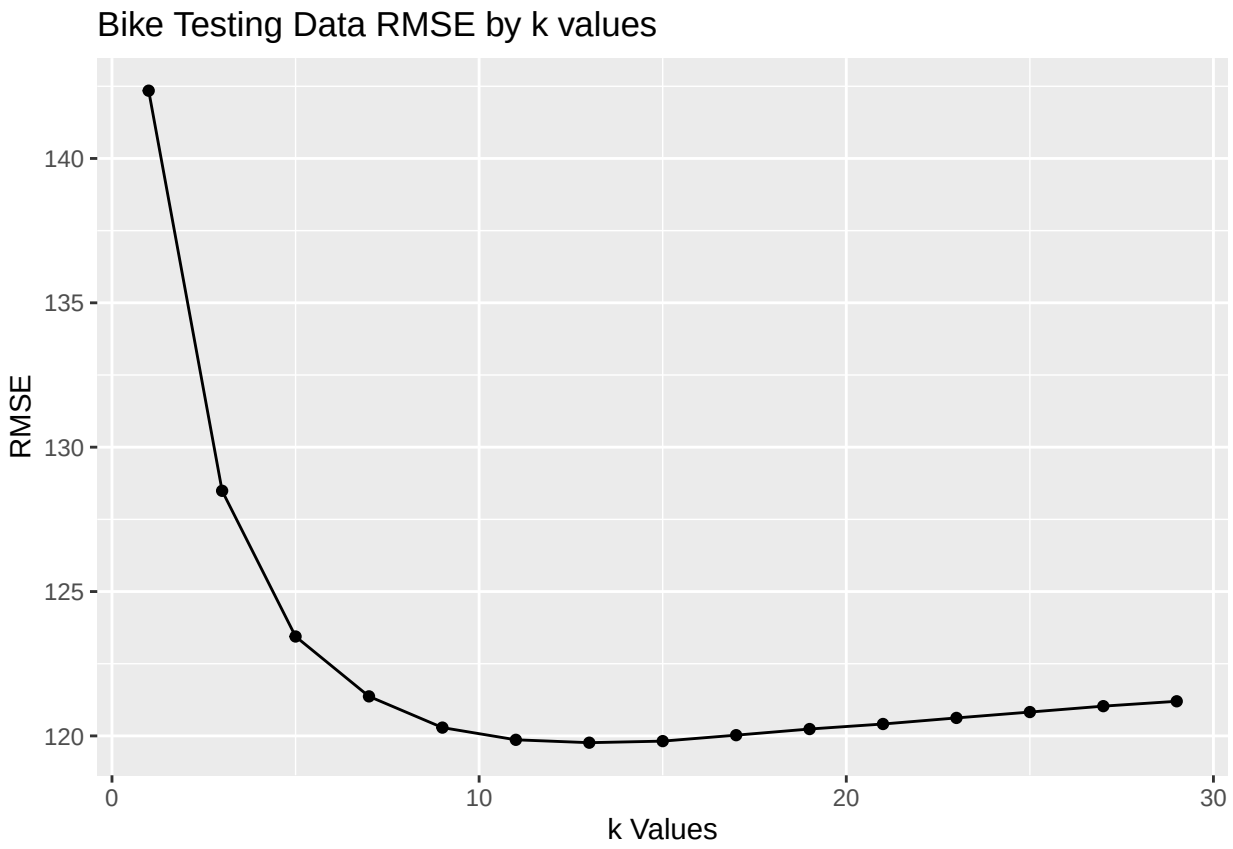
```
summary_train_knn_fit %>%
  filter(.metric %in% c("rmse", "rsq"))
```

```
## # A tibble: 30 x 4
##    .metric .estimator .estimate     k
##    <chr>   <chr>          <dbl> <dbl>
##  1 rmse    standard      23.0       1
##  2 rsq     standard       0.984     1
##  3 rmse    standard      42.8       3
##  4 rsq     standard       0.947     3
##  5 rmse    standard      60.0       5
##  6 rsq     standard       0.895     5
##  7 rmse    standard      70.3       7
##  8 rsq     standard       0.855     7
##  9 rmse    standard      77.4       9
## 10 rsq     standard       0.824     9
## # ... with 20 more rows
```

```
summary_train_knn_fit %>%
  filter(.metric == "rmse") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Bike Training Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```



```
summary_test_knn_fit %>%
  filter(.metric == "rmse") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Bike Testing Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```

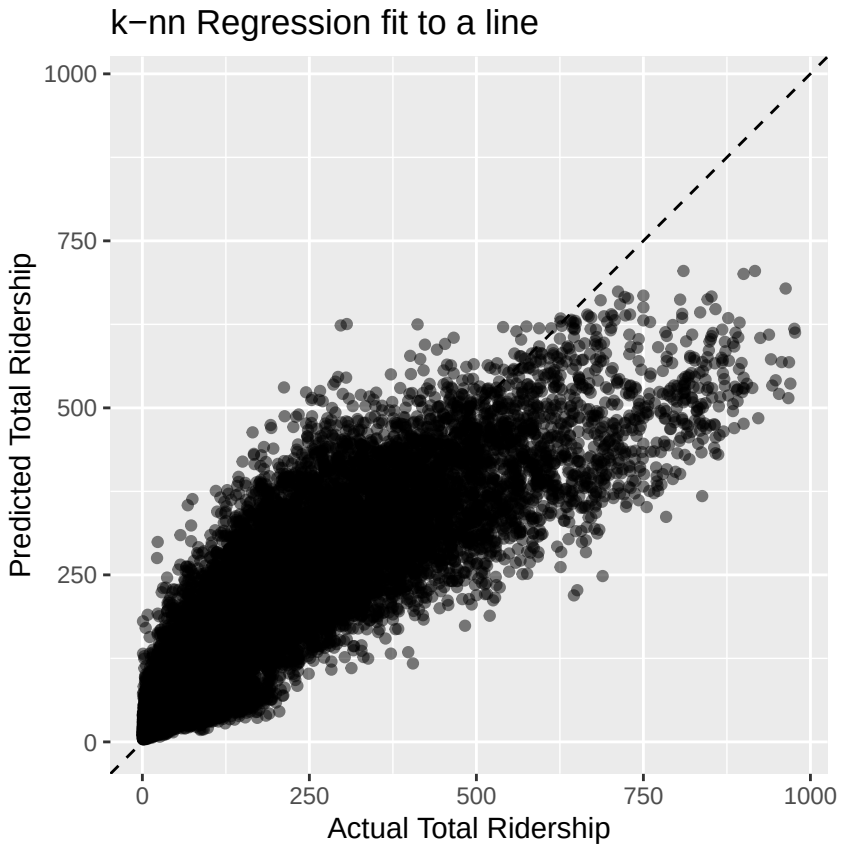


This plot shows that there is an RMSE elbow at the k value of 15, which means that 15 is the number of neighbors that we should use.

#Final Model

```
# Apply the same recipe to the final combined dataset
bike_baked <- bake(bike_recipe, new_data = bike_data)

# knn spec for optimal value of k
knn_spec <- nearest_neighbor(neighbor = 15) %>%
  set_engine("kknn") %>%
  set_mode("regression")

# Use the optimal knn spec to fit the prepared combined dataframe
final_knn_fit <- knn_spec %>%
  fit(total_riders ~., data = bike_baked)

# Regression fit to a line
final_knn_fit %>%
```

```
  predict(bike_baked) %>%
  bind_cols(bike_baked) %>%
  select(total_riders, .pred) %>%
  ggplot(aes(x = total_riders, y = .pred)) +
  # Create a diagonal line:
  geom_abline(lty = 2) +
  geom_point(alpha = 0.5) +
  labs(title = "k-nn Regression fit to a line",
       y = "Predicted Total Ridership",
       x = "Actual Total Ridership") +
  # Scale and size the x- and y-axis uniformly:
  coord_obs_pred()
```



This regression line is somewhat hard to read but there is a decent relationship between the two variables.

Predict

```
# Create a dataframe for the new data

new_bike <- tibble(season_Winter = c(1,0,0,0,0,0),
                   season_Spring = c(0,1,1,1,0,0),
                   season_Summer = c(0,0,0,0,1,0),
                   season_Fall = c(0,0,0,0,0,1),
                   month_X1 = c(0,0,0,0,0,0),
                   month_X2 = c(2,0,0,0,0,0),
                   month_X3 = c(0,3,0,0,0,0),
```

```
                    month_X4 = c(0,0,0,0,0,0),
                    month_X5 = c(0,0,5,0,0,0),
                    month_X6 = c(0,0,0,6,0,0),
                    month_X7 = c(0,0,0,0,0,0),
                    month_X8 = c(0,0,0,0,0,0),
                    month_X9 = c(0,0,0,0,9,0),
                    month_X10 = c(0,0,0,0,0,10),
                    month_X11 = c(0,0,0,0,0,0),
                    month_X12 = c(0,0,0,0,0,0),
                    time_of_day_X1 = c(1,0,0,0,0,0),
                    time_of_day_X2 = c(0,0,1,0,0,1),
                    time_of_day_X3 = c(0,1,0,1,0,0),
                    time_of_day_X4 = c(0,0,0,0,1,0),
                    holiday_yes = c(0,0,1,0,0,0),
                    holiday_no = c(1,1,0,1,1,1),
                    weekday_X0 = c(0,0,0,0,0,0),
                    weekday_X1 = c(0,0,1,1,0,0),
                    weekday_X2 = c(0,0,0,0,0,0),
                    weekday_X3 = c(0,1,0,0,0,0),
                    weekday_X4 = c(0,0,0,0,0,0),
                    weekday_X5 = c(0,0,0,0,0,1),
                    weekday_X6 = c(1,0,0,0,1,0),
                    working_day_yes = c(0,1,0,1,0,1),
                    working_day_no = c(1,0,1,0,1,0),
                    weather_X1 = c(1,0,0,1,0,0),
                    weather_X2 = c(0,1,1,0,0,1),
                    weather_X3 = c(0,0,0,0,1,0),
                    weather_X4 = c(0,0,0,0,0,0),
                    temp = c(5,13,24,22,26,16),
                    atemp = c(3,13,24,24,30,17),
                    hum = c(60,76,87,45,67,85),
                    windspeed = c(0,15,7,31,17,9))
```

```
# Predict classification of new data
predict(final_knn_fit, new_data = new_bike)
```

```
## # A tibble: 6 x 1
##   .pred
##   <dbl>
## 1  52.6
## 2  33.7
## 3  48.0
## 4 163.
## 5 156.
## 6  53.6
```

The Model - 2012 Data

```
bike_data_2012 <- bike_data %>%
  filter(year == 2012)
```

```
kn_bike_data_2012 <- bike_data_2012 %>%
  select(-instant, -date_day, -year, -hour, -registered, -casual)
```

Split the data

```
set.seed(2021)

bike_split_2012 <- initial_split(kn_bike_data_2012, prop = .75)

bike_split_train_2012 <- training(bike_split_2012)

bike_split_test_2012 <- testing(bike_split_2012)

bike_split_train_2012 %>%
  glimpse()
```

```
## Rows: 6,551
## Columns: 12
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ holiday      <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weekday      <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, n~
## $ weather      <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 2, 2, 1, 1, ~
## $ temp         <dbl> 0.36, 0.32, 0.30, 0.28, 0.28, 0.26, 0.26, 0.26, 0.40, 0.4~
## $ atemp        <dbl> 0.3485, 0.3485, 0.3333, 0.3030, 0.2879, 0.2727, 0.2576, 0~
## $ hum          <dbl> 0.66, 0.76, 0.81, 0.81, 0.81, 0.93, 0.93, 0.87, 0.62, 0.5~
## $ windspeed    <dbl> 0.1343, 0.0000, 0.0000, 0.0896, 0.1045, 0.1343, 0.1642, 0~
## $ total_riders <dbl> 93, 75, 52, 8, 5, 2, 7, 14, 201, 223, 267, 265, 215, 111,~
## $ time_of_day  <fct> 1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, ~
```

```
bike_split_test_2012 %>%
  glimpse()
```

```
## Rows: 2,183
## Columns: 12
## $ season       <fct> Winter, Winter, Winter, Winter, Winter, Winter, Winter, W~
## $ month        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ holiday      <fct> no, no, no, no, no, yes, yes, yes, yes, yes, yes, yes, ye~
## $ weekday      <fct> 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, ~
## $ working_day  <fct> no, no, no, no, no, no, no, no, no, no, no, no, no, no, yes, ~
## $ weather      <fct> 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, ~
## $ temp         <dbl> 0.36, 0.26, 0.30, 0.34, 0.42, 0.36, 0.28, 0.24, 0.26, 0.2~
## $ atemp        <dbl> 0.3788, 0.2727, 0.3182, 0.3333, 0.4242, 0.3182, 0.2576, 0~
## $ hum          <dbl> 0.66, 0.93, 0.81, 0.76, 0.67, 0.34, 0.45, 0.35, 0.35, 0.3~
## $ windspeed    <dbl> 0.0000, 0.1045, 0.1045, 0.1343, 0.3881, 0.4478, 0.3284, 0~
## $ total_riders <dbl> 48, 40, 70, 138, 105, 7, 4, 68, 109, 154, 93, 66, 34, 3, ~
## $ time_of_day  <fct> 1, 2, 2, 2, 4, 1, 1, 2, 2, 3, 4, 4, 4, 1, 1, 1, 2, 1, 1, ~
```

Prep the data

```
bike_recipe_2012 <- recipe(total_riders ~ ., data = bike_split_train_2012) %>%
  step_dummy(holiday, working_day, one_hot = FALSE) %>%
  step_dummy(season, weekday, weather, time_of_day, month, one_hot = TRUE) %>%
  prep()
```

Apply recipe to training data

```
bike_juiced_2012 <- juice(bike_recipe_2012)
```

Bake the test data

```
# Apply the same recipe to the test data
bike_split_test_baked_2012 <- bake(bike_recipe_2012, new_data = bike_split_test_2012)
```

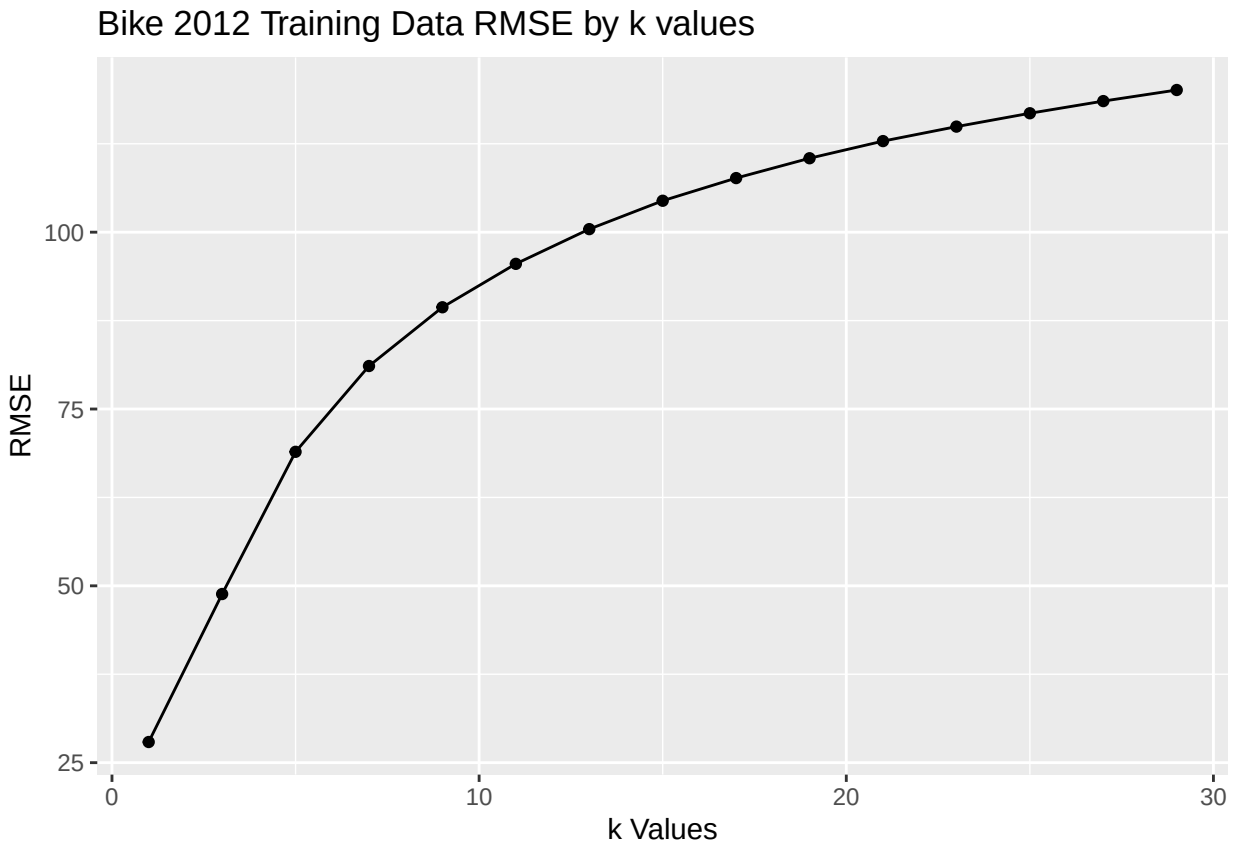
Test multiple values of k

```
# Optimal K value usually found is the square root of N, where N is the total number of samples
possible_optimal_k_2012  <- sqrt(nrow(bike_data_2012))

# Set initial value of i

begin <- 1

for (i in seq(begin, 30, 2)) {

# Make a knn spec
knn_spec_2012 <- nearest_neighbor(neighbor = i) %>%
  set_engine("kknn") %>%
  set_mode("regression")

# Use the knn spec to fit the prepared training data
knn_fit_2012 <- knn_spec_2012 %>%
  fit(total_riders ~., data = bike_juiced_2012)

# Evaluate the model with the training data
train_knn_fit_2012 <-  knn_fit_2012 %>%
    predict(bike_juiced_2012) %>%
    bind_cols(bike_juiced_2012) %>%
    metrics(truth = total_riders, estimate = .pred) %>%
    mutate(k = i)

# Evaluate the model with the testing data
test_knn_fit_2012 <- knn_fit_2012 %>%
    predict(bike_split_test_baked_2012) %>%
    bind_cols(bike_split_test_baked_2012) %>%
    metrics(truth = total_riders, estimate = .pred) %>%
    mutate(k = i)

# Create summary performance evaluation dataframes
if (i == begin) {
summary_train_knn_fit_2012 <- train_knn_fit_2012
summary_test_knn_fit_2012 <-  test_knn_fit_2012
} else {
summary_train_knn_fit_2012 <- bind_rows(summary_train_knn_fit_2012, train_knn_fit_2012)
summary_test_knn_fit_2012 <- bind_rows(summary_test_knn_fit_2012, test_knn_fit_2012)
```

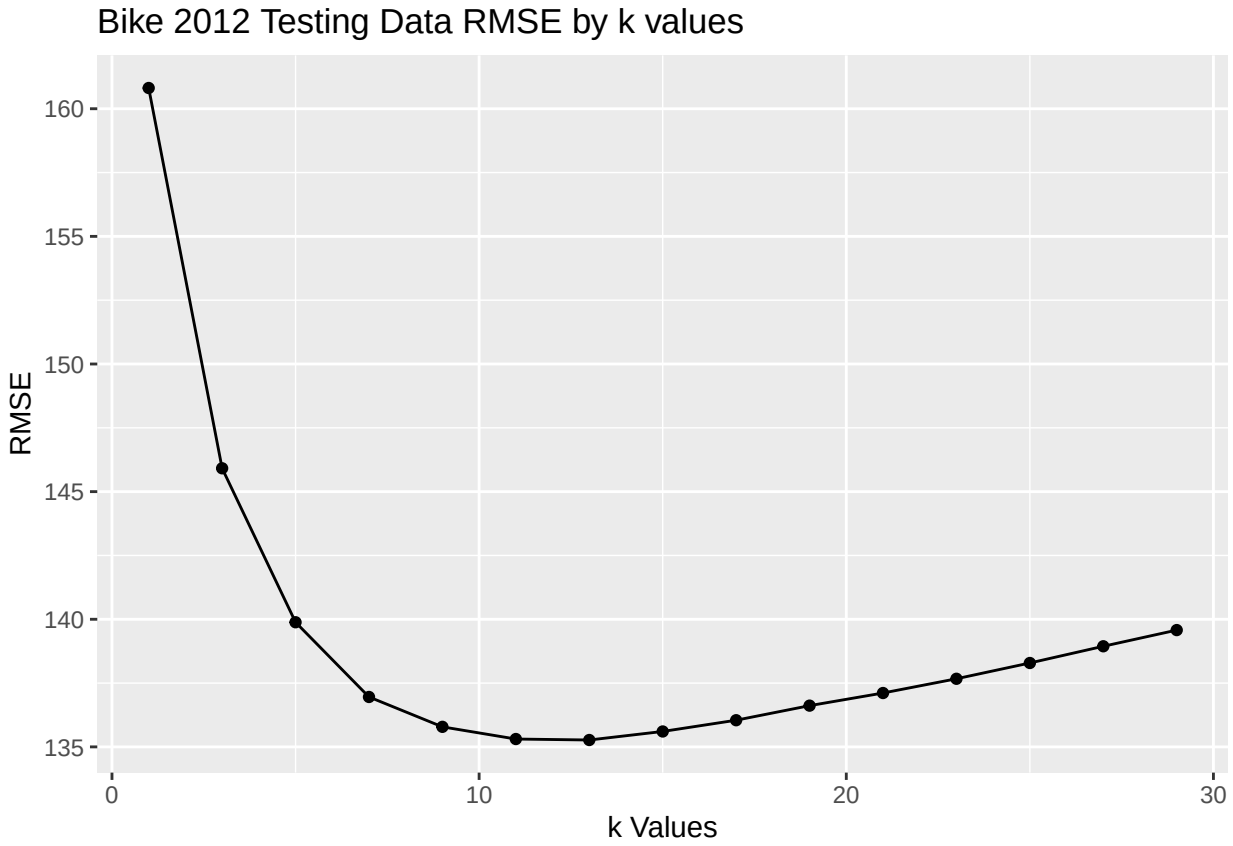
```
}

}

# Output summary performance dataframesrm()
summary_train_knn_fit_2012 %>%
  filter(.metric %in% c("rmse", "rsq"))
```

```
## # A tibble: 30 x 4
##    .metric .estimator .estimate     k
##    <chr>   <chr>          <dbl> <dbl>
##  1 rmse    standard      27.9       1
##  2 rsq     standard       0.982     1
##  3 rmse    standard      48.8       3
##  4 rsq     standard       0.947     3
##  5 rmse    standard      68.9       5
##  6 rsq     standard       0.895     5
##  7 rmse    standard      81.1       7
##  8 rsq     standard       0.854     7
##  9 rmse    standard      89.4       9
## 10 rsq     standard       0.822     9
## # ... with 20 more rows
```

```
summary_train_knn_fit_2012 %>%
  filter(.metric == "rmse") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Bike 2012 Training Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```




```
summary_test_knn_fit_2012 %>%
  filter(.metric == "rmse") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Bike 2012 Testing Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```

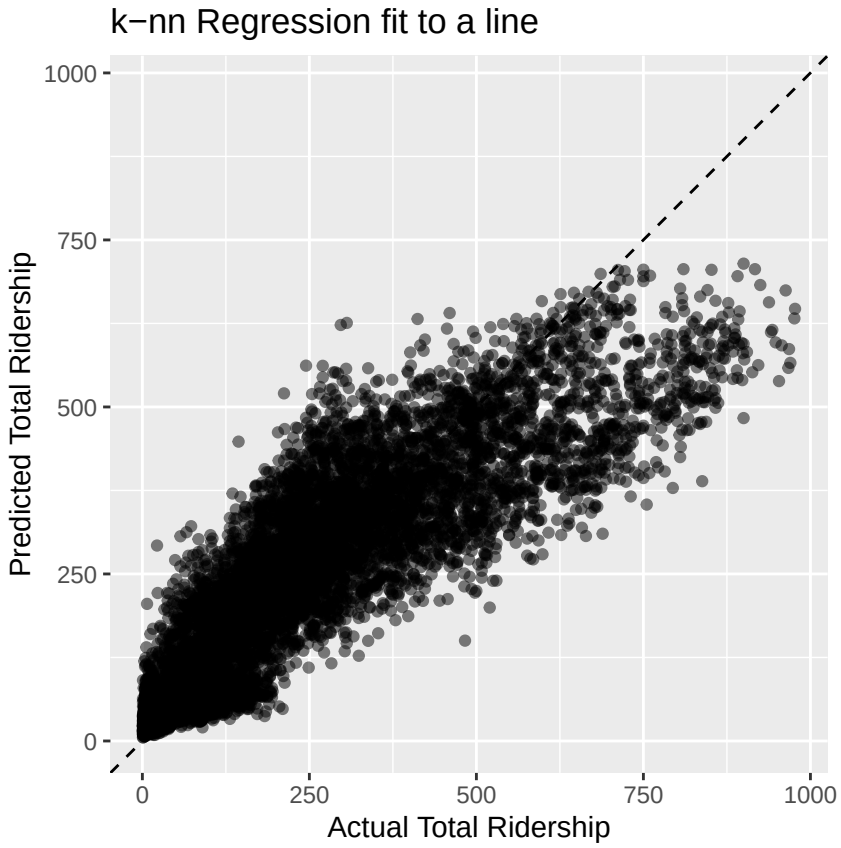


This plot shows that there is an RMSE elbow at the k value of 13, which means that 13 is the number of neighbors that we should use.

#Final Model

```
# Apply the same recipe to the final combined dataset
bike_baked_2012 <- bake(bike_recipe_2012, new_data = bike_data_2012)

# knn spec for optimal value of k
knn_spec_2012 <- nearest_neighbor(neighbor = 13) %>%
  set_engine("kknn") %>%
  set_mode("regression")

# Use the optimal knn spec to fit the prepared combined dataframe
final_knn_fit_2012 <- knn_spec_2012 %>%
  fit(total_riders ~., data = bike_baked_2012)

# Regression fit to a line
final_knn_fit_2012 %>%
  predict(bike_baked_2012) %>%
  bind_cols(bike_baked_2012) %>%
  select(total_riders, .pred) %>%
  ggplot(aes(x = total_riders, y = .pred)) +
  # Create a diagonal line:
  geom_abline(lty = 2) +
  geom_point(alpha = 0.5) +
  labs(title = "k-nn Regression fit to a line",
       y = "Predicted Total Ridership",
```

```
      x = "Actual Total Ridership") +
  # Scale and size the x- and y-axis uniformly:
  coord_obs_pred()
```



The regression shows a somewhat strong relationship between the two variables. It is stronger than the regression for the data frame that includes 2011 data.

Predict

```
# Create a dataframe for the new data
new_bike_2012 <- tibble(season_Winter = c(1,0,0,0,0,0),
                        season_Spring = c(0,1,1,1,0,0),
                        season_Summer = c(0,0,0,0,1,0),
                        season_Fall = c(0,0,0,0,0,1),
                        month_X1 = c(0,0,0,0,0,0),
                        month_X2 = c(2,0,0,0,0,0),
                        month_X3 = c(0,3,0,0,0,0),
                        month_X4 = c(0,0,0,0,0,0),
                        month_X5 = c(0,0,5,0,0,0),
                        month_X6 = c(0,0,0,6,0,0),
                        month_X7 = c(0,0,0,0,0,0),
                        month_X8 = c(0,0,0,0,0,0),
                        month_X9 = c(0,0,0,0,9,0),
                        month_X10 = c(0,0,0,0,0,10),
                        month_X11 = c(0,0,0,0,0,0),
                        month_X12 = c(0,0,0,0,0,0),
```

```
                    time_of_day_X1 = c(1,0,0,0,0,0),
                    time_of_day_X2 = c(0,0,1,0,0,1),
                    time_of_day_X3 = c(0,1,0,1,0,0),
                    time_of_day_X4 = c(0,0,0,0,1,0),
                    holiday_yes = c(0,0,1,0,0,0),
                    holiday_no = c(1,1,0,1,1,1),
                    weekday_X0 = c(0,0,0,0,0,0),
                    weekday_X1 = c(0,0,1,1,0,0),
                    weekday_X2 = c(0,0,0,0,0,0),
                    weekday_X3 = c(0,1,0,0,0,0),
                    weekday_X4 = c(0,0,0,0,0,0),
                    weekday_X5 = c(0,0,0,0,0,1),
                    weekday_X6 = c(1,0,0,0,1,0),
                    working_day_yes = c(0,1,0,1,0,1),
                    working_day_no = c(1,0,1,0,1,0),
                    weather_X1 = c(1,0,0,1,0,0),
                    weather_X2 = c(0,1,1,0,0,1),
                    weather_X3 = c(0,0,0,0,1,0),
                    weather_X4 = c(0,0,0,0,0,0),
                    temp = c(5,13,24,22,26,16),
                    atemp = c(3,13,24,24,30,17),
                    hum = c(60,76,87,45,67,85),
                    windspeed = c(0,15,7,31,17,9))
```

```
# Predict classification of new data
predict(final_knn_fit_2012, new_data = new_bike_2012)
```

```
## # A tibble: 6 x 1
##   .pred
##   <dbl>
## 1  86.6
## 2 185.
## 3  61.6
## 4 273.
## 5 221.
## 6 139.
```

Ultimately, I would say that the 2012 model has the better results because the regression line is stronger and management thinks that the trend seen in 2012 will continue.

Part 3

Data Wrangling

```
auto_data <- read.csv("data/auto_segments.csv")
```

```
auto_data_clean <- auto_data %>%
  drop_na() %>%
  rename_with(~ tolower(gsub(".", "_", .x, fixed = TRUE))) %>%
  rename(category = var_1)
```

```
glimpse(auto_data_clean)
```

```
## Rows: 9,228
## Columns: 11
## $ id               <int> 462809, 466315, 461735, 461319, 460156, 464347, 465015~
## $ gender           <chr> "Male", "Female", "Male", "Male", "Male", "Female", "F~
## $ ever_married     <chr> "No", "Yes", "Yes", "Yes", "No", "No", "Yes", "Yes", "~
## $ age              <int> 22, 67, 67, 56, 32, 33, 61, 55, 26, 19, 58, 41, 32, 31~
## $ graduated        <chr> "No", "Yes", "Yes", "No", "Yes", "Yes", "Yes", "Yes", ~
## $ profession       <chr> "Healthcare", "Engineer", "Lawyer", "Artist", "Healthc~
## $ work_experience <int> 1, 1, 0, 0, 1, 1, 0, 1, 1, 4, 0, 1, 9, 1, 1, 0, 12, 3,~
## $ spending_score   <chr> "Low", "Low", "High", "Average", "Low", "Low", "Low", ~
## $ family_size      <int> 4, 1, 2, 2, 3, 3, 3, 4, 3, 4, 1, 2, 5, 6, 4, 1, 1, 4, ~
## $ category         <chr> "Cat_4", "Cat_6", "Cat_6", "Cat_6", "Cat_6", "Cat_6", ~
## $ segmentation     <chr> "D", "B", "B", "C", "C", "D", "D", "C", "A", "D", "B",~
```

```
auto_data_clean <- auto_data_clean %>%
  mutate(gender = factor(gender)) %>%
  mutate(ever_married = factor(ever_married)) %>%
  mutate(graduated = factor(graduated)) %>%
  mutate(profession = factor(profession)) %>%
  mutate(work_experience = factor(work_experience)) %>%
  mutate(spending_score = factor(spending_score)) %>%
  mutate(family_size = factor(family_size)) %>%
  mutate(category = factor(category))
```

```
library(gmodels)

CrossTable(auto_data_clean$segmentation, auto_data_clean$category)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  9228
##
##
##                              | auto_data_clean$category
## auto_data_clean$segmentation |           |     Cat_1 |     Cat_2 |     Cat_3 |     Cat_4 |     Cat_5
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##                            A |        19 |        37 |       111 |       268 |       374 |        24
##                              |     0.190 |     0.069 |     3.194 |     4.584 |     7.158 |     0.287
##                              |     0.008 |     0.015 |     0.046 |     0.111 |     0.154 |     0.010
##                              |     0.237 |     0.274 |     0.222 |     0.299 |     0.301 |     0.235
##                              |     0.002 |     0.004 |     0.012 |     0.029 |     0.041 |     0.003
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##                            B |        16 |        26 |       120 |       200 |       277 |        22
##                              |     0.275 |     0.742 |     0.292 |     0.090 |     0.125 |     0.068
##                              |     0.008 |     0.012 |     0.057 |     0.095 |     0.132 |     0.010
```

```
##                              |     0.200 |     0.193 |     0.240 |     0.223 |     0.223 |     0.216
##                              |     0.002 |     0.003 |     0.013 |     0.022 |     0.030 |     0.002
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##                            C |        21 |        27 |       112 |       158 |       156 |        19
##                              |     0.238 |     0.742 |     0.330 |    13.522 |    63.988 |     1.069
##                              |     0.010 |     0.012 |     0.051 |     0.073 |     0.072 |     0.009
##                              |     0.263 |     0.200 |     0.224 |     0.176 |     0.126 |     0.186
##                              |     0.002 |     0.003 |     0.012 |     0.017 |     0.017 |     0.002
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##                            D |        24 |        45 |       158 |       270 |       434 |        37
##                              |     0.205 |     1.766 |     3.209 |     2.536 |    26.347 |     2.969
##                              |     0.010 |     0.018 |     0.063 |     0.107 |     0.172 |     0.015
##                              |     0.300 |     0.333 |     0.315 |     0.301 |     0.350 |     0.363
##                              |     0.003 |     0.005 |     0.017 |     0.029 |     0.047 |     0.004
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##                 Column Total |        80 |       135 |       501 |       896 |      1241 |       102
##                              |     0.009 |     0.015 |     0.054 |     0.097 |     0.134 |     0.011
## -----------------------------|-----------|-----------|-----------|-----------|-----------|----------
##
##
```

The Model

```
auto <- auto_data_clean %>%
  select(-id, -age)
```

Split the data

```
set.seed(2021)

auto_split <- initial_split(auto, prop = .75)

auto_train <- training(auto_split)

auto_test <- testing(auto_split)

auto_train %>%
  glimpse()
```

```
## Rows: 6,921
## Columns: 9
## $ gender          <fct> Female, Male, Male, Male, Female, Female, Female, Fema~
## $ ever_married    <fct> Yes, Yes, Yes, No, No, Yes, Yes, No, Yes, No, No, Yes,~
## $ graduated       <fct> Yes, Yes, No, Yes, Yes, Yes, Yes, Yes, No, No, No, Yes~
## $ profession      <fct> Engineer, Lawyer, Artist, Healthcare, Healthcare, Engi~
## $ work_experience <fct> 1, 0, 0, 1, 1, 0, 1, 1, 0, 9, 1, 1, 0, 12, 3, 13, 1, 9~
## $ spending_score  <fct> Low, High, Average, Low, Low, Low, Average, Low, Low, ~
## $ family_size     <fct> 1, 2, 2, 3, 3, 3, 4, 3, 1, 5, 6, 4, 1, 1, 4, 2, 3, 8, ~
## $ category        <fct> Cat_6, Cat_6, Cat_6, Cat_6, Cat_6, Cat_7, Cat_6, Cat_6~
## $ segmentation    <chr> "B", "B", "C", "C", "D", "D", "C", "A", "B", "D", "B",~
```

```
auto_test %>%
  glimpse()
```

```
## Rows: 2,307
## Columns: 9
## $ gender          <fct> Male, Male, Female, Female, Female, Male, Female, Fema~
## $ ever_married    <fct> No, No, No, No, , No, Yes, Yes, Yes, Yes, Yes, Yes, No~
## $ graduated       <fct> No, No, No, Yes, No, No, No, Yes, Yes, No, Yes, No, , ~
## $ profession      <fct> Healthcare, Healthcare, Healthcare, Artist, Executive,~
## $ work_experience <fct> 1, 4, 1, 5, 1, 1, 1, 8, 0, 1, 1, 1, 8, 9, 8, 8, 7, 0, ~
## $ spending_score  <fct> Low, Low, Low, Low, Average, Low, Average, Low, Averag~
## $ family_size     <fct> 4, 4, 2, 2, 3, 3, 5, 2, 4, 2, 1, 1, 2, 2, 4, 1, 5, 3, ~
## $ category        <fct> Cat_4, Cat_4, Cat_1, Cat_6, Cat_3, Cat_2, Cat_3, Cat_6~
## $ segmentation    <chr> "D", "D", "C", "B", "B", "D", "D", "A", "B", "B", "A",~
```

Prep the data

```
auto_recipe <- recipe(segmentation ~ ., data = auto_train) %>%
  step_dummy(gender, ever_married, graduated, one_hot = FALSE) %>%
  step_dummy(profession, work_experience, spending_score, family_size, category, one_hot = TRUE) %>%
  prep()
```

Apply recipe to training data

```
auto_juiced <- juice(auto_recipe)
```

Bake the test data

```
# Apply the same recipe to the test data
auto_test_baked <- bake(auto_recipe, new_data = auto_test)
```

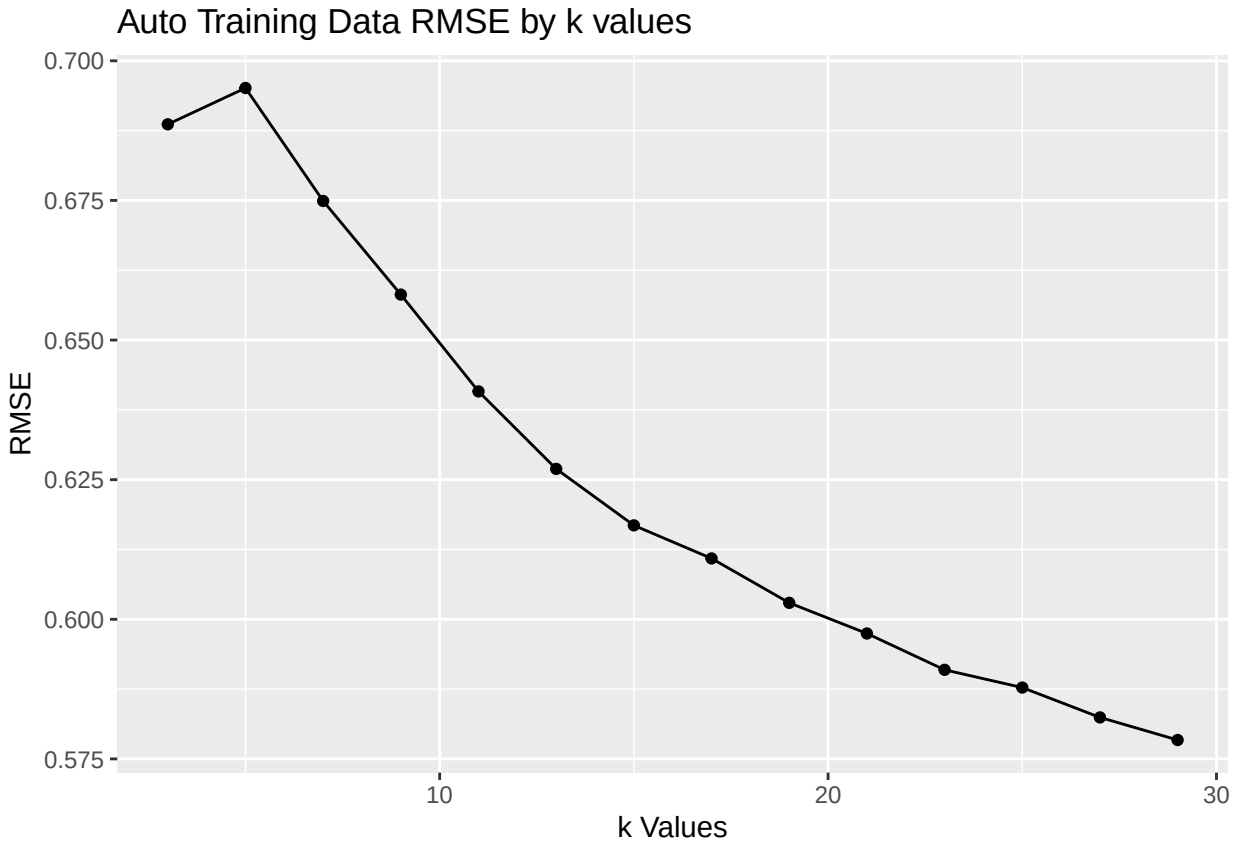
Test multiple values of k

```
# Optimal K value usually found is the square root of N, where N is the total number of samples
possible_optimal_k  <- sqrt(nrow(auto))

# Set initial value of i

begin <- 3

for (i in seq(begin, 30, 2)) {

# Make a knn spec
auto_knn_spec <- nearest_neighbor(neighbor = i) %>%
  set_engine("kknn") %>%
  set_mode("classification")

# Use the knn spec to fit the prepared training data
```

```
auto_knn_fit <- auto_knn_spec %>%
  fit(segmentation ~., data = auto_juiced)

# Evaluate the model with the training data
auto_train_knn_fit <-  auto_knn_fit %>%
    predict(auto_juiced) %>%
    bind_cols(auto_juiced) %>%
    metrics(truth = segmentation, estimate = .pred_class) %>%
    mutate(k = i)

# Evaluate the model with the testing data
auto_test_knn_fit <- auto_knn_fit %>%
    predict(auto_test_baked) %>%
    bind_cols(auto_test_baked) %>%
    metrics(truth = segmentation, estimate = .pred_class) %>%
    mutate(k = i)

# Create summary performance evaluation dataframes
if (i == begin) {
auto_summary_train_knn_fit <- auto_train_knn_fit
auto_summary_test_knn_fit <-  auto_test_knn_fit
} else {
auto_summary_train_knn_fit <- bind_rows(auto_summary_train_knn_fit, auto_train_knn_fit)
auto_summary_test_knn_fit <- bind_rows(auto_summary_test_knn_fit, auto_test_knn_fit)
}

}

# Output summary performance dataframesrm()
auto_summary_train_knn_fit
```

```
## # A tibble: 28 x 4
##    .metric  .estimator .estimate     k
##    <chr>    <chr>          <dbl> <dbl>
##  1 accuracy multiclass     0.689     3
##  2 kap      multiclass     0.585     3
##  3 accuracy multiclass     0.695     5
##  4 kap      multiclass     0.593     5
##  5 accuracy multiclass     0.675     7
##  6 kap      multiclass     0.566     7
##  7 accuracy multiclass     0.658     9
##  8 kap      multiclass     0.544     9
##  9 accuracy multiclass     0.641    11
## 10 kap      multiclass     0.520    11
## # ... with 18 more rows
```

```
auto_summary_train_knn_fit %>%
  filter(.metric == "accuracy") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Auto Training Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```




```
auto_summary_test_knn_fit %>%
  filter(.metric == "accuracy") %>%
  ggplot(aes(x = k, y = .estimate)) +
  geom_point() +
  geom_line() +
  labs(title = "Auto Testing Data RMSE by k values",
       x = "k Values",
       y = "RMSE")
```

This plot is somewhat confusing to read because it changes so much. I am going to go with a k value of 15 because that is where the second elbow is. I think the first elbow is no representative because the rest plot is a different trend.

#Final Model

```
# Apply the same recipe to the final combined dataset
auto_baked <- bake(auto_recipe, new_data = auto)

# knn spec for optimal value of k
auto_knn_spec <- nearest_neighbor(neighbor = 15) %>%
  set_engine("kknn") %>%
  set_mode("classification")

# Use the optimal knn spec to fit the prepared combined dataframe
auto_final_knn_fit <- auto_knn_spec %>%
  fit(segmentation ~., data = auto_baked)
```

Predict

```
# Create a dataframe for the new data
new_auto <- tibble(gender_Female = c(1,0,1,0,0),
                   gender_Male = c(0,1,0,1,1),
                   ever_married_X = c(0,0,0,0,0),
                   ever_married_No = c(0,0,0,0,0),
                   ever_married_Yes = c(1,1,1,1,1),
                   graduated_X = c(0,0,0,0,0),
```

```
                    graduated_No = c(0,0,0,0,0),
                    graduated_Yes = c(1,1,1,1,1),
                    profession_X = c(0,0,0,0,0),
                    profession_Artist = c(0,0,1,0,0),
                    profession_Doctor = c(0,0,0,1,1),
                    profession_Engineer = c(1,0,0,6,0),
                    profession_Entertainment = c(0,0,0,0,0),
                    profession_Executive = c(0,0,0,0,0),
                    profession_Healthcare = c(0,1,0,0,0),
                    profession_Homemaker = c(0,0,0,0,0),
                    profession_Lawyer = c(0,0,0,0,0),
                    profession_Marketing = c(0,0,0,0,0),
                    work_experience_X0 = c(1,0,0,1,0),
                    work_experience_X1 = c(0,0,1,0,0),
                    work_experience_X2 = c(0,0,0,0,0),
                    work_experience_X3 = c(0,0,0,0,0),
                    work_experience_X4 = c(0,0,0,0,0),
                    work_experience_X5 = c(0,0,0,0,5),
                    work_experience_X6 = c(0,0,0,0,0),
                    work_experience_X7 = c(0,0,0,0,0),
                    work_experience_X8 = c(0,1,0,0,0),
                    work_experience_X9 = c(0,0,0,0,0),
                    work_experience_X10 = c(0,0,0,0,0),
                    work_experience_X11 = c(0,0,0,0,0),
                    work_experience_X12 = c(0,0,0,0,0),
                    work_experience_X13 = c(0,0,0,0,0),
                    work_experience_X14 = c(0,0,0,0,0),
                    spending_score_Average = c(0,1,1,0,0),
                    spending_score_High = c(0,0,0,1,0),
                    spending_score_Low = c(1,0,0,0,1),
                    family_size_X1 = c(1,0,0,0,0),
                    family_size_X2 = c(0,0,0,0,0),
                    family_size_X3 = c(0,0,1,0,1),
                    family_size_X4 = c(0,1,0,0,0),
                    family_size_X5 = c(0,0,0,1,0),
                    family_size_X6 = c(0,0,0,0,0),
                    family_size_X7 = c(0,0,0,0,0),
                    family_size_X8 = c(0,0,0,0,0),
                    family_size_X9 = c(0,0,0,0,0),
                    category_X = c(0,0,0,0,0),
                    category_Cat_1 = c(0,0,0,0,0),
                    category_Cat_2 = c(0,0,0,0,0),
                    category_Cat_3 = c(0,0,0,0,0),
                    category_Cat_4 = c(0,0,0,1,0),
                    category_Cat_5 = c(0,0,0,0,0),
                    category_Cat_6 = c(1,1,1,0,1),
                    category_Cat_7 = c(0,0,0,0,0))
```

Predict classification of new data

```
predict(auto_final_knn_fit, new_data = new_auto)
```

```
## # A tibble: 5 x 1
##   .pred_class
##   <fct>
## 1 D
## 2 C
## 3 C
## 4 B
## 5 A
```

Project Log

1). Date: 4/12/2021 Problem: I was having trouble figuring out what to do for part one of the project. Solution: I met with Professor Ballenger, who gave me some ideas.

2). Date: 4/15/2021 Problem: I wanted to make the line graphs for part 2 to worked. Solution: I met with Professor Ballenger, who assured me that they were already correct.

3). Date: 4/16/2021 Problem: I was having trouble getting the tibbles to work with the final predictions Solution: I met with Professor Ballenger, who told me that I needed to factor the month vector and create dummy variables for each month.

Honor Pledge

On my honor, I have neither given nor received any unacknowledged aid on this assignment.

-Brennan Black