Correlation

waheeb Algabri

In this blog we will explore the concept of correlation, how to calculate correlations in R, and how correlations can be used to provide insight about the relationship between two columns of data.

We work on Teca regression data (tecaRegressionData.rds). We will use this data to provide insight to our business question, which is, "How are quarterly sales affected by quarter of the year, region, and by product category (parent name)?"

Column Name	Definition
site_name	The name of the site where the data was collected.
quarter	The quarter in which the data was collected.
quarterNoYear	The quarter without the year component.
lat	Latitude coordinates of the site location.
long	Longitude coordinates of the site location.
totalRevenue	Total revenue generated by the site.
Pop_py1	Population data for the site from the previous year.
Fuel_py1	Fuel sales data for the site from the previous year.
Juicetonics_py1	Sales data for juice and tonics for the site from the previous year.
ColdDispensedBeverage_py1	Sales data for cold dispensed beverages for the site from the previous year.
OffInvoiceCigs_py1	Sales data for cigarettes through off-invoice channels for the site from the previous year.
Lottery_py1	Sales data for lottery products for the site from the previous year.
Other_py1	Other sales data for the site from the previous year.

This data is based on the teca dataset that you may have used before. The original teca data is very granular and each row in that dataset represents a line item for a purchase at one of about 150 gas stations and convenience stores in the central United States.

The data that we're using aggregates the data by store for each quarter of 2019. Thus, every store should have four rows of data that pertains to one row for each quarter.

Correlation can be used along with scatter plots to investigate two-way relationships between quarterly sales and the other variables. Scatter plots help us to *see* a relationship whereas correlations allow us to *quantify* a relationship.

Because we want to better understand how quarterly sales is affected by the other variables, quarterly sales is known as our dependent variable. In other words, we expect that the values for quarterly sales depend on the values of the other variables. In contrast, the other variables are known as independent variables because we expect that their values are determined independently of the other variables. These independent variables are also known as predictor variables, or explanatory variables because they are being used to explain and predict the value of quarterly sales.

Preliminaries

If you haven't already done so, then install the tidyverse collection of packages and the corrplot package.

You only need to install these packages once on the machine that you're using. If you have not already done so, then you can do so by uncommenting the code chunk below and running it. If you *have* already done so, then you should *not* run the next code chunk.

```
# install.packages('tidyverse')
# install.packages('corrplot')
```

Load the tidyverse collection of packages, as well as the corrplot package.

```
library(tidyverse)
library(corrplot)
```

Make sure that you have also downloaded the tecaRegressionData.rds file into the same folder in which this file is saved. Use the next code chunk to read in the data and load it as a dataframe object.

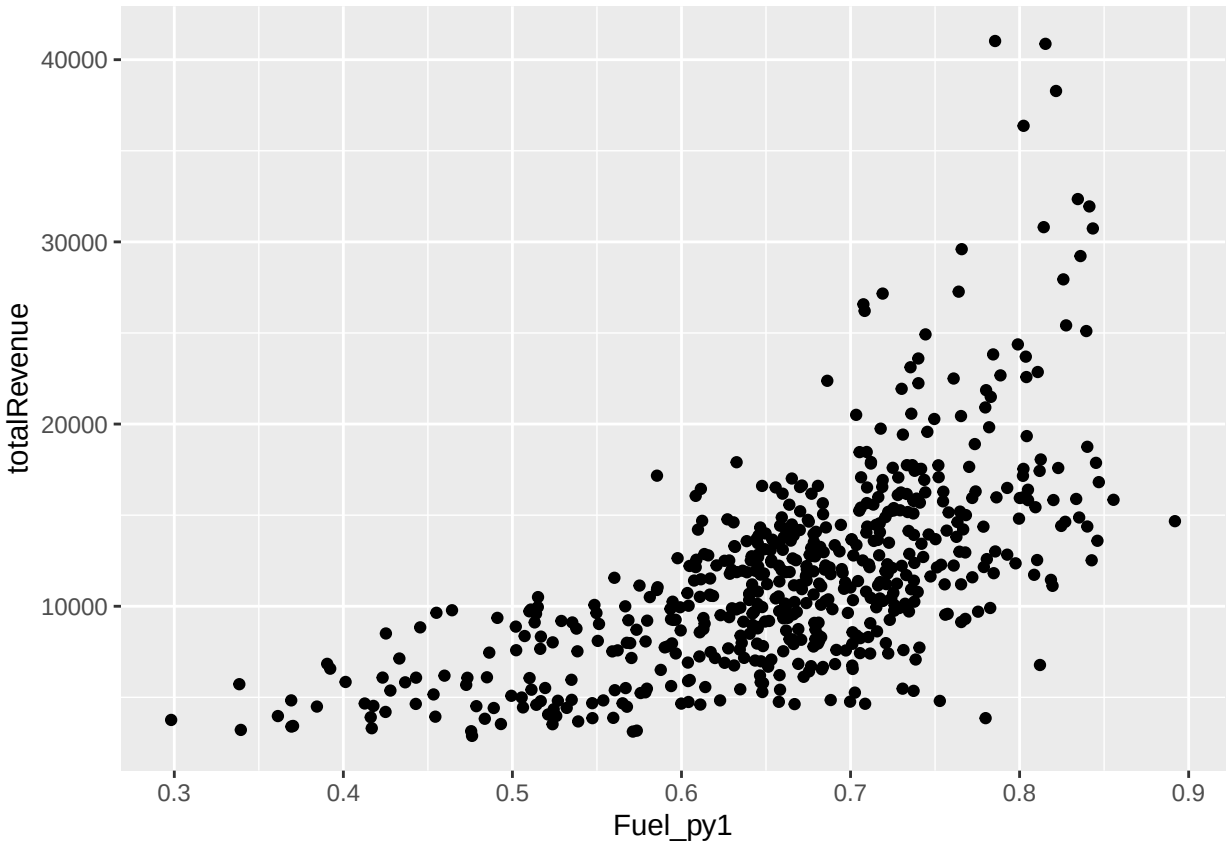
```
trd <- readRDS('tecaRegressionData.rds')
```

Visualizing Two-Way Relationships

Let's start our investigation of how variables are related by using visualizations. Because we are interested in predicting quarterly sales performance, let's focus our attention on exploring the relationship between totalRevenue and some of the other variables in this dataset. Specifically, let's investigate the relationship of totalRevenue with the percentage of sales that came from the Fuel category during the same quarter in the previous year, Fuel_py1, by creating a scatter plot.

Typically, the dependent variable values are plotted on the y-axis and the predictor variable is plotted on the x-axis.

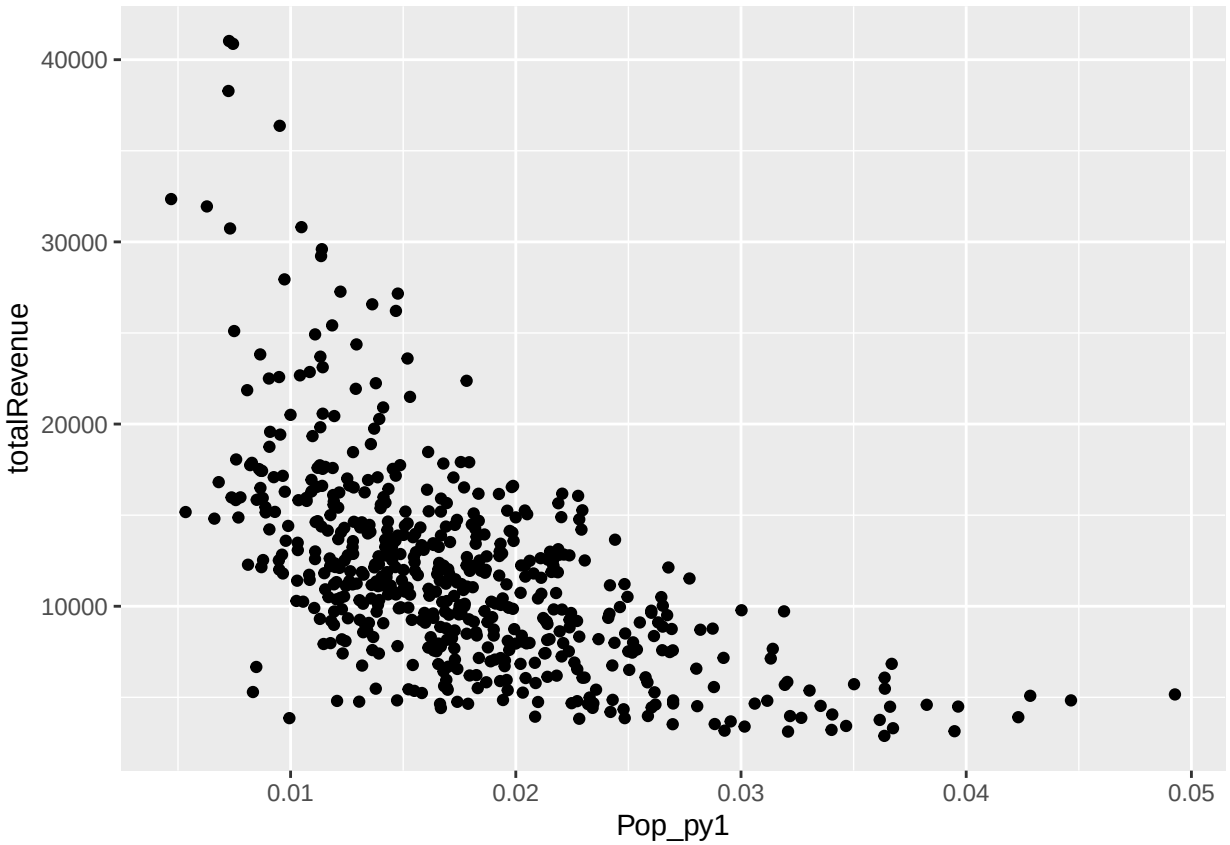
```
ggplot(trd, aes(x = Fuel_py1, y = totalRevenue)) +
  geom_point()
```



What can we learn from this plot? Well, it seems that the total revenue is often higher if the percentage of sales from Fuel during the previous year is also higher.

Let's compare this relationship to the relationship between the total quarterly sales, totalRevenue, and the percentage of total quarterly revenue last year that came from Pop products, Pop_py1.

```
ggplot(trd, aes(x = Pop_py1, y = totalRevenue)) +
  geom_point()
```



This plot indicates that the relationship between Pop sales from the previous year has a negative relationship with total quarterly revenue such that as Pop_py1 increases, totalRevenue often decreases. Wouldn't it be great if we could quantify the direction and strength of these relationships so that we can communicate them to others? We can to some extent. This is the purpose of a correlation coefficient.

What is Correlation?

Correlation is simply the extent to which two variables are linearly related to each other. A correlation coefficient is the numeric value that we use to quantify those relationships. Correlation coefficients can range from -1 at the low end to 1 at the high end. * A negative correlation coefficient means that as one variable increases the other variable decreases. As the correlation coefficient gets closer to -1 that relationship gets stronger and more certain. * A correlation of 0 means that there is no linear relationship. It's actually rare to have a correlation of exactly zero due to random movements in the same direction.
* A positive correlation means that as one variable increases the other variable also increases. As the correlation coefficient get closer to 1, that relationship gets stronger and more certain.

Thus, a correlation coefficient indicates both the direction and strength of a linear relationship. Let's calculate the correlation between totalRevenue and Fuel_py1:

```
cor(trd[,c('totalRevenue', 'Fuel_py1')])
```

```
##              totalRevenue  Fuel_py1
## totalRevenue    1.0000000 0.6320499
## Fuel_py1        0.6320499 1.0000000
```

Notice that it actually prints out a correlation matrix, which contains a lot of unneccesary information when

you are looking at just two variables. The correlation between the two variables is .63. However, a matrix like this is useful if you're looking at the correlation between three or more variables.

Correlation Matrix

Let's also include Pop_py1 to see a more useful correlation matrix:

```
cor(trd[,c('totalRevenue', 'Fuel_py1', 'Pop_py1')])
```

```
##              totalRevenue   Fuel_py1    Pop_py1
## totalRevenue    1.0000000  0.6320499 -0.5845053
## Fuel_py1        0.6320499  1.0000000 -0.7924551
## Pop_py1        -0.5845053 -0.7924551  1.0000000
```

You can still see the correlation between totalRevenue and Fuel_py1, but now you can also see the correlation between totalRevenue and Pop_py1 and Fuel_py1 and Pop_py1. Notice that the correlation between totalRevenue and Pop_py1 is -.58, as was illustrated by the scatter plot.

It's also worth noting that exploring the correlations between the independent variables can be useful, as well. In this case, they are negatively correlated.

If correlation is based on a two-way relationship, won't we have a lot of correlations to analyze in our dataset? Yes! As the number of of variables in a dataset increases, the number of correlation to analyze increases by even more. Specifically:
* If you have **three** variables in a dataset, you can have **three** possible correlations to analyze.
* If you have **four** variables in a dataset, you can have **six** possible correlations to analyze.
* If you have **five** variables in a dataset, you can have **ten** possible correlations to analyze.

One way to explore all of these correlations is by creating a correlation matrix for all of the numeric variables in a dataframe. Here is how we can do that.

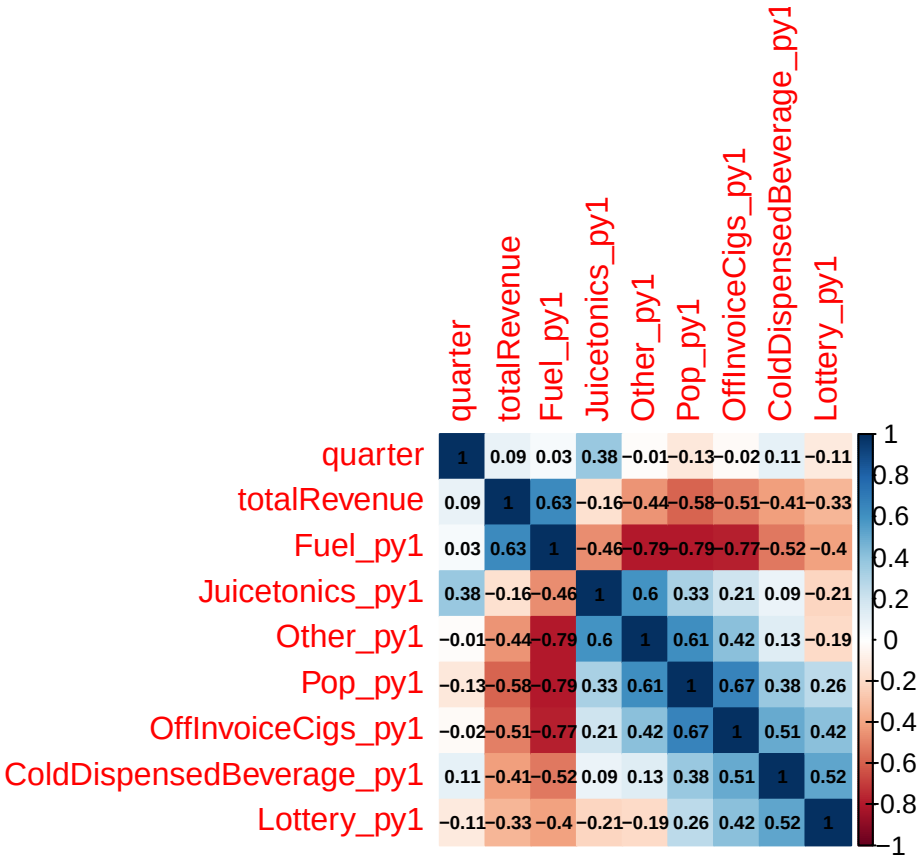
```
ctrd <- cor(trd %>% select(where(is.numeric)))
ctrd
```

```
##                               quarter totalRevenue    Pop_py1    Fuel_py1
## quarter                    1.00000000   0.09160391 -0.1291364  0.03436791
## totalRevenue               0.09160391   1.00000000 -0.5845053  0.63204988
## Pop_py1                   -0.12913641  -0.58450532  1.0000000 -0.79245512
## Fuel_py1                   0.03436791   0.63204988 -0.7924551  1.00000000
## Juicetonics_py1            0.37857783  -0.16471007  0.3293326 -0.46443825
## ColdDispensedBeverage_py1  0.10563972  -0.41017711  0.3766656 -0.52099363
## OffInvoiceCigs_py1        -0.02101153  -0.50987871  0.6677604 -0.77283609
## Lottery_py1               -0.11094902  -0.33229129  0.2634759 -0.40300405
## Other_py1                 -0.01137992  -0.44321963  0.6139875 -0.79223244
##                           Juicetonics_py1 ColdDispensedBeverage_py1
## quarter                        0.37857783                0.10563972
## totalRevenue                  -0.16471007               -0.41017711
## Pop_py1                        0.32933256                0.37666560
## Fuel_py1                      -0.46443825               -0.52099363
## Juicetonics_py1                1.00000000                0.08753644
## ColdDispensedBeverage_py1      0.08753644                1.00000000
## OffInvoiceCigs_py1             0.20827782                0.50519310
## Lottery_py1                   -0.21020173                0.52272476
```

```
## Other_py1                       0.59848129                0.13487043
##                              OffInvoiceCigs_py1 Lottery_py1   Other_py1
## quarter                             -0.02101153  -0.1109490 -0.01137992
## totalRevenue                        -0.50987871  -0.3322913 -0.44321963
## Pop_py1                              0.66776039   0.2634759  0.61398747
## Fuel_py1                            -0.77283609  -0.4030041 -0.79223244
## Juicetonics_py1                      0.20827782  -0.2102017  0.59848129
## ColdDispensedBeverage_py1            0.50519310   0.5227248  0.13487043
## OffInvoiceCigs_py1                   1.00000000   0.4174073  0.41777106
## Lottery_py1                          0.41740726   1.0000000 -0.19466991
## Other_py1                            0.41777106  -0.1946699  1.00000000
```

That's too much information to process, though. Let's make it easier to see patterns by using colors, shapes, and groups.

```
corrplot(ctrd
         , method = 'color' # I also like pie and ellipse
         , order = 'hclust' # Orders the variables so that ones that behave similarly are placed next t
         , addCoef.col = 'black'
         , number.cex = .6 # Lower values decrease the size of the numbers in the cells
         )
```



Now that we have created a nicely looking correlation plot, let's pause and consider what patterns stand out to us.

First, you'll notice that going down the diagonal there is a value of 1 for every cell. This is to be expected because it means that the variable is perfectly correlated with itself.

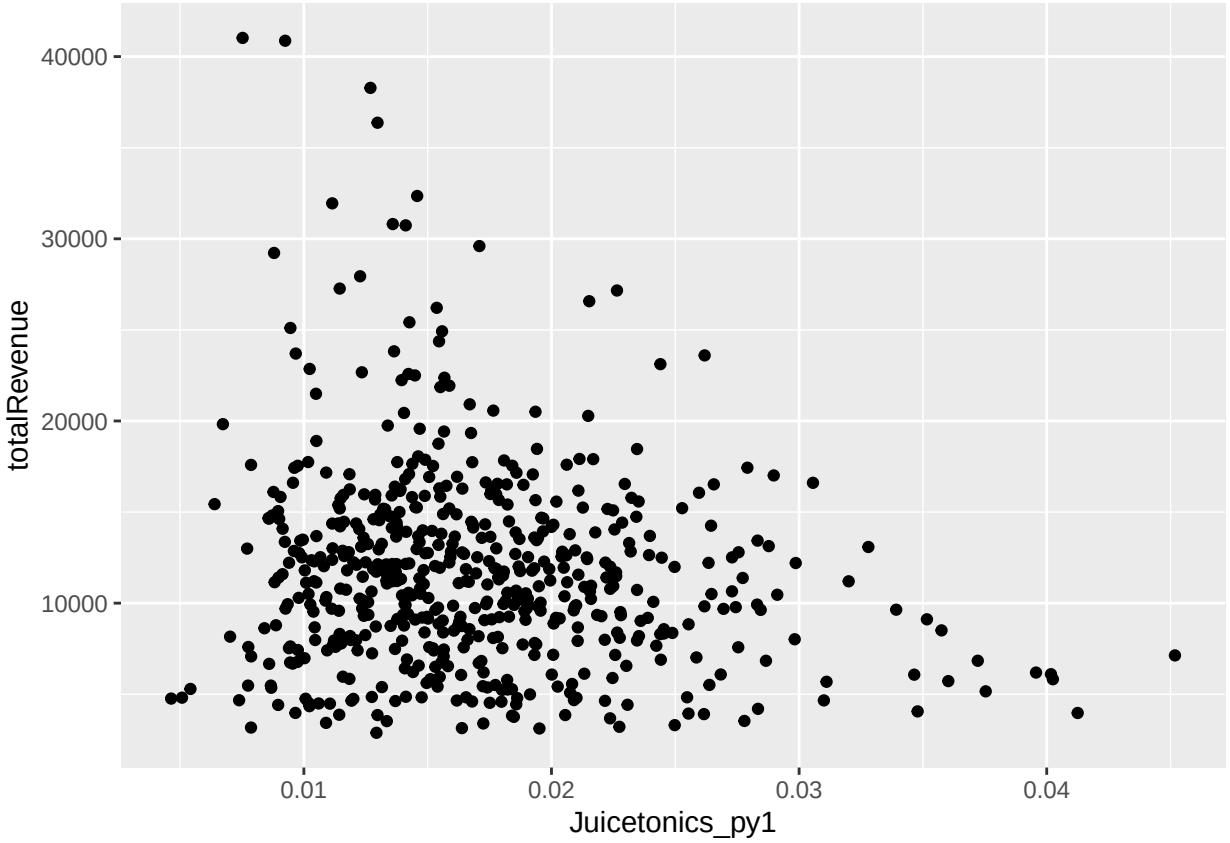
Perhaps you'll also notice that after ordering the variables using hierarchical clustering, it's easier to see which variables behave similarly. For instance, Pop_py1 and OffInvoiceCigs_py1 seem to have the same pattern of positive and negative correlations with the other variables, and a strong positive correlation with each other.

One way to prioritize your investigation is to focus on the column or row of the dependent variable, totalRevenue. By looking at the totalRevenue column we can see that Fuel_py1 is the variable that has the strongest positive correlation of 0.63, while Pop_py1 has the strongest negative correlation of -0.58. Thus, these correlation coefficients quantify the relationship that was displayed in the scatter plots.

Scatterplot of totalRevenue and Juicetonics_py1

Let's look at one more scatterplot for a variable that is not strongly correlated with totalRevenue, Juicetonics_py1. Based on the correlation matrix above, these variables have a correlation of -.16.

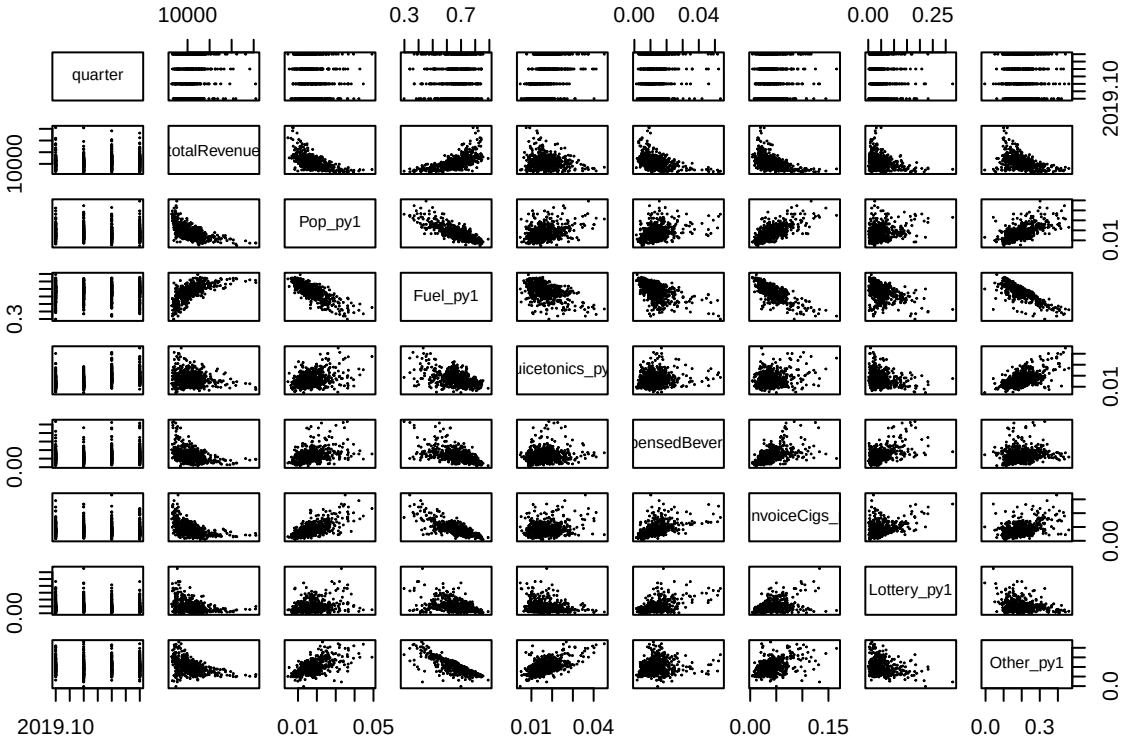
```
ggplot(trd, aes(x = Juicetonics_py1, y = totalRevenue)) +
  geom_point()
```



Notice that the grouping still has a downward slope, but the points are not as tightly clustered together as they are with Pop_py1 and totalRevenue.

To get a quick visualization of all the pairwise scatterplots, you can creat a pairplot.

```
pairs(trd %>% select(where(is.numeric)), cex = .1) # See the help documentation for adding in coefficie
```



Sometimes correlation coefficients are strongly influenced by an outlier. These scatterplots can help provide an idea of whether that is happening. If it is, then it may be worth removing the outliers.

Concluding Comments

Just as descriptive statistics are complemented by visualizing the data as a histogram, when analyzing correlation it's helpful to look at both numeric values and visualizations. Scatter plots are commonly used to visualize correlations.