Regression Data

waheeb Algabri

In this lesson you will be introduced to the data that will be used to answer this question which is, "How are quarterly sales affected by quarter of the year, region, and by product category (parent name)?"

Preliminaries

If you haven't already done so, then install the tidyverse collection of packages. There are eight packages in this collection:
1. dplyr - for dataframe manipulation
2. tidyr - for reshaping data
3. ggplot2 - for visualizations
4. readr - for reading and writing data
5. stringr - for working with character strings
6. forcats - for working with factors
7. tibble - an "improved" alternative to dataframes
8. purrr - for working with functions and vectors (we won't use this)

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')
```

Load the tidyverse collection of packages by running the next code chunk.

```
library(tidyverse)
```

Make sure that you have also downloaded the tecaRegressionData.rds file into the same folder in which this file is saved. The .rds format has two benefits over a .csv format. 1. It compresses the file so that it doesn't take up as much space, and also can be loaded into R faster. 2. It preserves the data type. This is especially helpful with dates and columns that you want to keep as either factors or character strings. In a .csv format, these columns will either all be read in as a character string or factor format.

Use the next code chunk to read in the data and load it as a dataframe object.

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

Getting to Know the Data

Column Name	Definition
site_name	The name of the site where the data was collected.

Column Name	Definition
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.

Let's explore the structure of the data by either clicking on the blue down arrow next to the trd dataframe, or by running the `str()` function.

```r
str(trd)
```

```
## tibble [564 x 13] (S3: tbl_df/tbl/data.frame)
##  $ site_name                : chr [1:564] "120 Clanton" "120 Clanton" "120 Clanton" "120 Clanton" ..
##  $ quarter                  : num [1:564] 2019 2019 2019 2019 2019 ...
##  $ quarterNoYear            : Factor w/ 4 levels "First","Second",..: 1 2 3 4 1 2 3 4 1 2 ...
##  $ lat                      : Factor w/ 2 levels "Northern","Southern": 1 1 1 1 1 1 1 1 2 2 ...
##  $ long                     : Factor w/ 2 levels "Eastern","Western": 1 1 1 1 2 2 2 2 1 1 ...
##  $ totalRevenue             : num [1:564] 7523 7586 8333 8882 7993 ...
##  $ Pop_py1                  : num [1:564] 0.025 0.0265 0.0228 0.0265 0.0244 ...
##  $ Fuel_py1                 : num [1:564] 0.559 0.502 0.517 0.502 0.568 ...
##  $ Juicetonics_py1          : num [1:564] 0.0152 0.0151 0.0245 0.0201 0.0166 ...
##  $ ColdDispensedBeverage_py1: num [1:564] 0.0165 0.0118 0.0196 0.022 0.0105 ...
##  $ OffInvoiceCigs_py1       : num [1:564] 0.0456 0.0543 0.0613 0.0778 0.0389 ...
##  $ Lottery_py1              : num [1:564] 0.0591 0.082 0.0547 0.0633 0.0524 ...
##  $ Other_py1                : num [1:564] 0.279 0.308 0.3 0.288 0.289 ...
```

We can see that there are 564 rows of data and 13 columns. The first six columns, site_name through totalRevenue should be pretty self explanatory while the last seven columns need more explanation. Regardless, we will explain each one of them:
1. **site_name** = a character string with the unique identifier of the store
2. **quarter** = a numeric value of the year and quarter of the data such that 2019.1 = first quarter of 2019
3. **quarterNoYear** = a factor data type that has a label of the quarter in a factor format. The value of First for the first observation that occurred in 2019 corresponds to 2019.1 in the quarter column.
4. **lat** = a factor data type that has a label to indicate whether the store falls in the Northern or Southern half of the stores
5. **long** = a factor data type that has a label to indicate whether the store falls in the Eastern or Western half of the stores

6. **totalRevenue** = the total amount of revenue for that store during the quarter. These numbers are low because it's only a sample of the data. *This is the main variable that we would like to predict and explain.*
7. **Pop__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the Pop parent name
8. **Fuel__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the Fuel parent name
9. **Juicetonics__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the Juicetonics parent name
10. **ColdDispensedBeverage__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the Fuel ColdDispensedBeverage parent name
11. **OffInvoiceCigs__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the OffInvoiceCigs parent name
12. **Lottery__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from the Lottery parent name
13. **Other__py1** = the percentage of totalRevenue from the same quarter during the prior year that came from one of the other parent names

Check for Missing Values and Completeness

Let's now check to see if there are any missing values in the data by adding up the number of missing values using the `sum()` and `is.na()` functions.

```
sum(is.na(trd))
```

```
## [1] 0
```

The result is zero, so there are no missing values.

The sum of the last six columns should add up to 1, meaning 100% of revenue during the same quarter of the prior year. Let's check to see if the rows of the last six columns add up to 1.

```
rowSums(trd[,7:13]) # This returns the sum of the last six columns for every row.
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [334] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [371] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [408] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [445] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [482] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [519] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [556] 1 1 1 1 1 1 1 1 1
```

```r
sum(rowSums(trd[,7:13])) # This adds up the prior values. Should equal 564--1 for each row.
```

```
## [1] 564
```

Now let's see how many unique stores there are.

```r
n_distinct(trd$site_name)
```

```
## [1] 141
```

If there are four observations for each store, then that would correspond to the 564 rows in the **trd** dataframe ($141 * 4 = 564$). It looks like we have a dataframe that is complete and ready for analysis.

Descriptive Statistics

We will now evaluate the univariate statstics for each column using the `summary()` function.

```r
summary(trd)
```

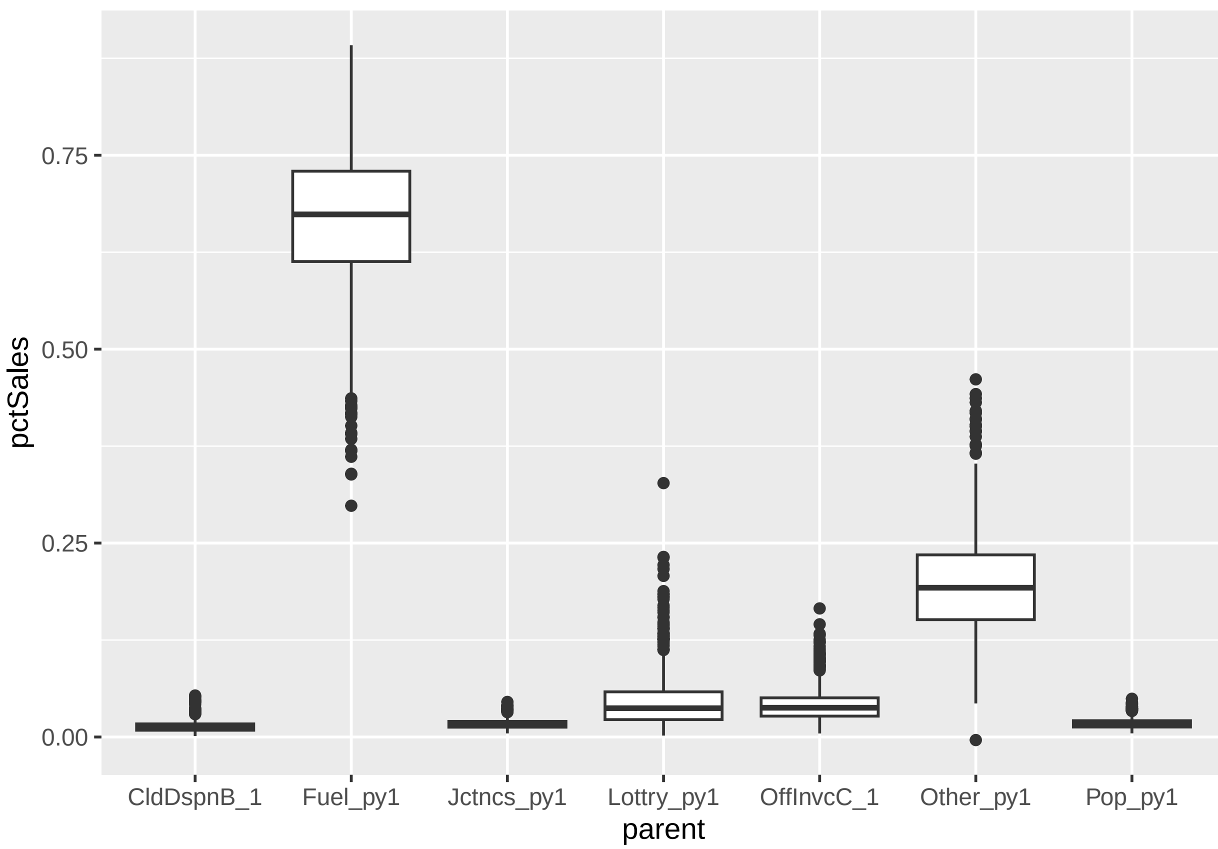
```
##   site_name            quarter     quarterNoYear       lat            long
##  Length:564         Min.   :2019   First :141    Northern:284   Eastern:280
##  Class :character   1st Qu.:2019   Second:141    Southern:280   Western:284
##  Mode  :character   Median :2019   Third :141
##                     Mean   :2019   Fourth:141
##                     3rd Qu.:2019
##                     Max.   :2019
##   totalRevenue      Pop_py1            Fuel_py1      Juicetonics_py1
##  Min.   : 2885   Min.   :0.004697   Min.   :0.2981   Min.   :0.004641
##  1st Qu.: 7986   1st Qu.:0.012874   1st Qu.:0.6130   1st Qu.:0.012668
##  Median :11203   Median :0.016673   Median :0.6738   Median :0.015875
##  Mean   :11751   Mean   :0.017603   Mean   :0.6627   Mean   :0.016937
##  3rd Qu.:14375   3rd Qu.:0.021017   3rd Qu.:0.7294   3rd Qu.:0.020111
##  Max.   :41026   Max.   :0.049268   Max.   :0.8919   Max.   :0.045178
##  ColdDispensedBeverage_py1 OffInvoiceCigs_py1  Lottery_py1
##  Min.   :0.001273          Min.   :0.004543   Min.   :0.00180
##  1st Qu.:0.008716          1st Qu.:0.026914   1st Qu.:0.02241
##  Median :0.012009          Median :0.037688   Median :0.03711
##  Mean   :0.013638          Mean   :0.041787   Mean   :0.04829
##  3rd Qu.:0.016896          3rd Qu.:0.050480   3rd Qu.:0.05823
##  Max.   :0.053464          Max.   :0.165751   Max.   :0.32730
##    Other_py1
##  Min.   :-0.003938
##  1st Qu.: 0.151279
##  Median : 0.192421
##  Mean   : 0.199002
##  3rd Qu.: 0.234813
##  Max.   : 0.461098
```

- The values for quarter appear to be rounded to the nearest integer. We can visually explore the data to check that there is a value in the tenths place for the quarter.

- quarterNoYear has 141 observations for each quarter, which is what we would expect
- lat and long are not quite evenly split because there is an odd number of sites, so one category will have four more observations
- totalRevenue indicates that the values range from 2885 to 41,026. The mean and median are pretty close to each other, so we would expect the distribution to be symmetric.
- If we look at the median values in the last six columns, we can see that Fuel and Other contributed the most to the totalRevenue values from the same quarter last year

Visualizations Those summary statistics are helpful, but the relative distributions can be identified much quicker by using box plots.

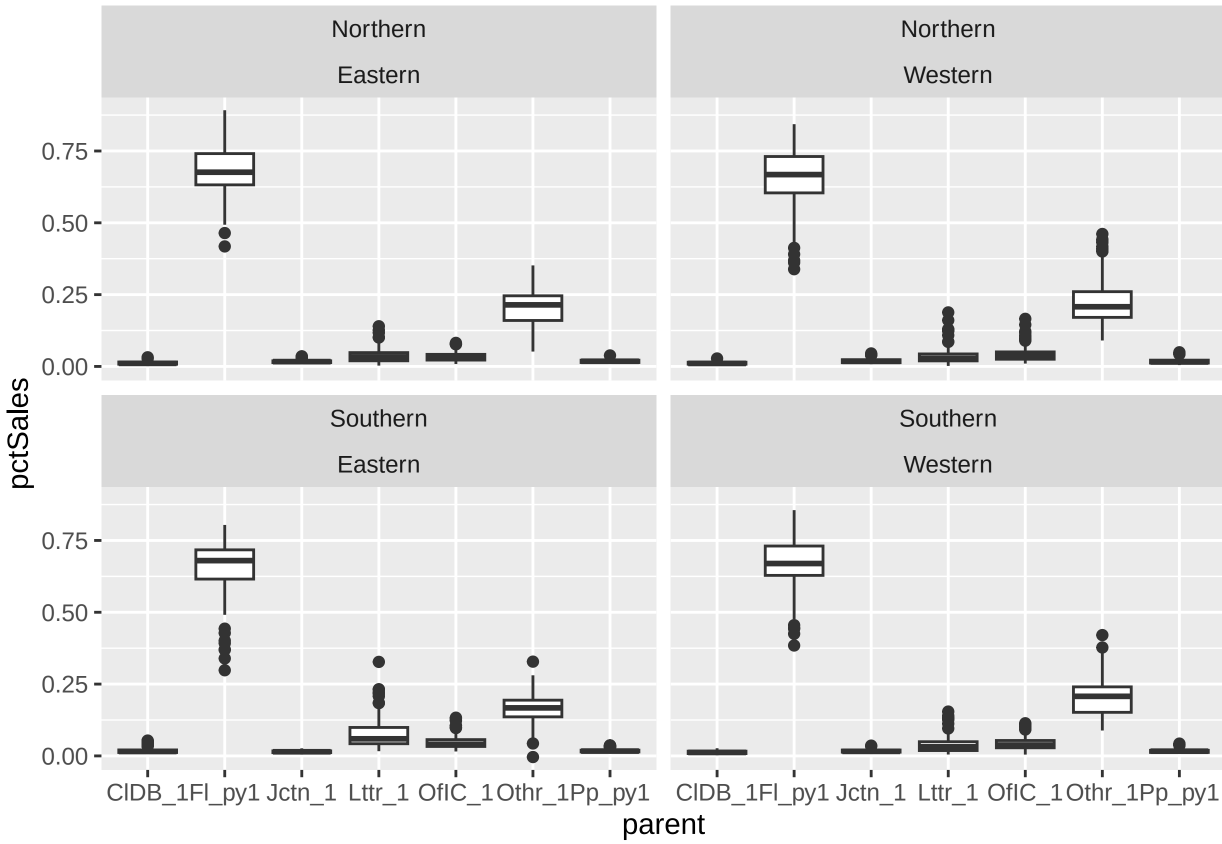
```
trd %>%
  pivot_longer(cols = 7:13, names_to = 'parent', values_to = 'pctSales') %>%
  mutate(parent = abbreviate(parent, 10)) %>%
  ggplot(aes(x = parent, y = pctSales)) +
  geom_boxplot()
```



This is a much faster way of communicating the relative percentage of sales that each parent makes up. It also helps us see that Lottery and OffInvoiceCigs contribute a little more to revenue last year relative to Pop, Juicetonics, and ColdDispensedBeverages.

Distribution of Parent Name by Region To get a quick idea of whether the parent categories make up a different percentage of revenue by region, we can use the `facet_grid()` function along with the lat and long columns.

```
trd %>%
  pivot_longer(cols = 7:13, names_to = 'parent', values_to = 'pctSales') %>%
  mutate(parent = abbreviate(parent, 6)) %>%
  ggplot(aes(x = parent, y = pctSales)) +
  geom_boxplot() +
  # facet_grid(~lat + long)
  facet_wrap(facets = vars(lat, long), nrow = 2)
```



There appears to be minor variations, but nothing major. The biggest difference appears to be in the South-Eastern region where there are often more lottery tickets sold than off-invoice cigs.

Concluding Comments

It's always to get a good idea of the data that you have before you analyze it. There would typically be a lot more data wrangling tasks, but this data has already been cleaned up so that we can focus on the analysis.