PREDICTING SALES IN ONE OF THE BIGGEST ECUADORIAN PHARMACEUTICAL INDUSTRIES - LIFE

Romina Jaramillo

27/4/2021

Introduction

LIFE is one of the 10 biggest Ecuadorian Pharmaceutical Industries in the country. It has been part of the ecuadorians life since 1940. This project pretends to apply regression analysis in order to predict the sales of the next years based on the previous ones.

This project works with two data sets, the first one is the training data that contains information between 2016 and 2019. And the other is the validation data set that contains information of 2020 and beginings of 2021.

The data structure contains 7 columns. The columns are:

1. Province: Ecuador is divided in 24 provinces (states) that are:

```
unique(trainingData$PROVINCE)
```

```
##  [1] "AZUAY"                          "BOLIVAR"
##  [3] "CAÑAR"                          "CARCHI"
##  [5] "CHIMBORAZO"                     "COTOPAXI"
##  [7] "EL ORO"                         "ESMERALDAS"
##  [9] "GALAPAGOS"                      "GUAYAS"
## [11] "IMBABURA"                       "LOJA"
## [13] "LOS RÍOS"                       "MANABI"
## [15] "MORONA SANTIAGO"                "NAPO"
## [17] "ORELLANA"                       "PASTAZA"
## [19] "PICHINCHA"                      "SANTA ELENA"
## [21] "SANTO DOMINGO DE LOS TSACHILAS" "SUCUMBIOS"
## [23] "TUNGURAHUA"                     "ZAMORA CHINCHIPE"
```

2. Presentation: each laboratory has their own product presentation. For example, one presentation of LIFE´s products for headaches is "BUPREXMIGRA TABL RECUB. x 20"; But, for the competence it is "MIGRA DORIXINA TABL x 20". We are analyzing the star product of the company (LIFE) known as "BUPREXMIGRA"

```
unique(trainingData$PRESENTATION)
```

```
## [1] "BUPREXMIGRA TABL RECUB. x 20"    "MIGRA DORIXINA TABL x 20"
## [3] "MIGRAFLASH CAPS BLANDA x 10"     "MIGRAX TABL RECUBI. x 10"
## [5] "NIMPAS TA.REC 30MG/ 200 MG x 10" "TONOPAN GRAG. x 100"
```

3. Laboratory: LIFE has considered 5 companies as competence. In total 6 laboratories

```
unique(trainingData$LABORATORY)
```

```
## [1] "LIFE"               "MEGALABS"           "JAMES BROWN PHARMA"
## [4] "SAVAL"              "VITA BEAUTY"        "GLAXOSMITHKLINE CH"
```

4. Year: for the training data, we are going to analyze the information between 2016 and 2019

5. Month: 12 months

6. Unit Sales: value in USD of the sales registrated

7. RX: represents the number of prescriptions in each month for that presentation.

A glimpse of our data is shown below:

```
glimpse(trainingData)
```

```
## Rows: 4,506
## Columns: 7
## $ PROVINCE      <chr> "AZUAY", "AZUAY", "AZUAY", "AZUAY", "AZUAY", "AZUAY",...
## $ PRESENTATION  <chr> "BUPREXMIGRA TABL RECUB. x 20", "BUPREXMIGRA TABL REC...
## $ LABORATORY    <chr> "LIFE", "LIFE", "LIFE", "LIFE", "LIFE", "LIFE", "LIFE...
## $ YEAR          <dbl> 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016,...
## $ MONTH         <dttm> 2016-01-01, 2016-02-01, 2016-03-01, 2016-04-01, 2016...
## $ `SALES UNITS` <dbl> 1965.000, 4029.001, 2090.000, 1966.000, 2160.000, 218...
## $ RX            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
```

Data Overview

The structure of the data is described above.

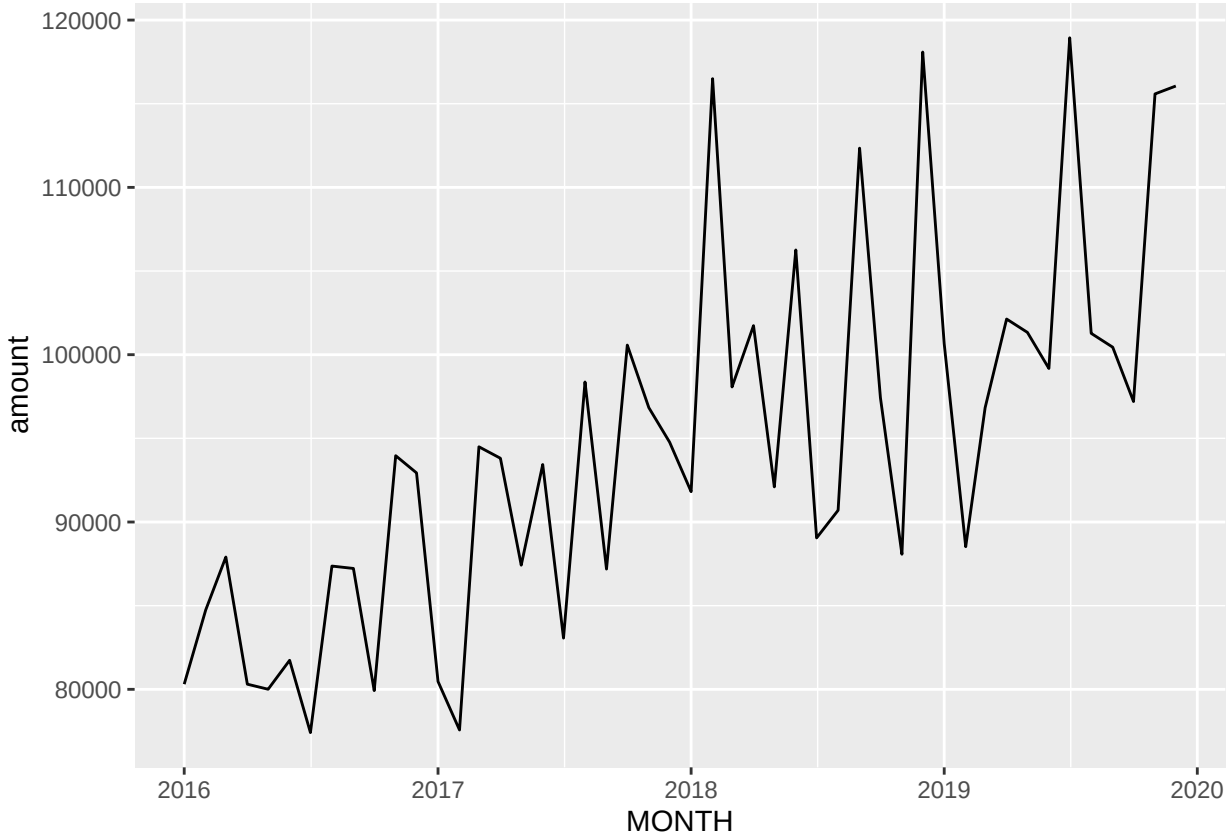
```
str(trainingData)
```

```
## 'data.frame':    4506 obs. of  7 variables:
##  $ PROVINCE    : chr  "AZUAY" "AZUAY" "AZUAY" "AZUAY" ...
##  $ PRESENTATION: chr  "BUPREXMIGRA TABL RECUB. x 20" "BUPREXMIGRA TABL RECUB. x 20" "BUPREXMIGRA TABL
##  $ LABORATORY  : chr  "LIFE" "LIFE" "LIFE" "LIFE" ...
##  $ YEAR        : num  2016 2016 2016 2016 2016 ...
##  $ MONTH       : POSIXct, format: "2016-01-01" "2016-02-01" ...
##  $ SALES UNITS : num  1965 4029 2090 1966 2160 ...
##  $ RX          : num  NA NA NA NA NA NA NA NA NA NA ...
```

The sales over the years has increased from around 80k to over 115k for the migraine market. Presenting the behaviour of sales units over the years:

```
## presenting all the sales accumulated per month since 2016 to December 2019
trainingData %>% group_by(MONTH) %>% summarize( amount = sum(`SALES UNITS`)) %>%
  ggplot(aes(MONTH,amount)) + geom_line()
```

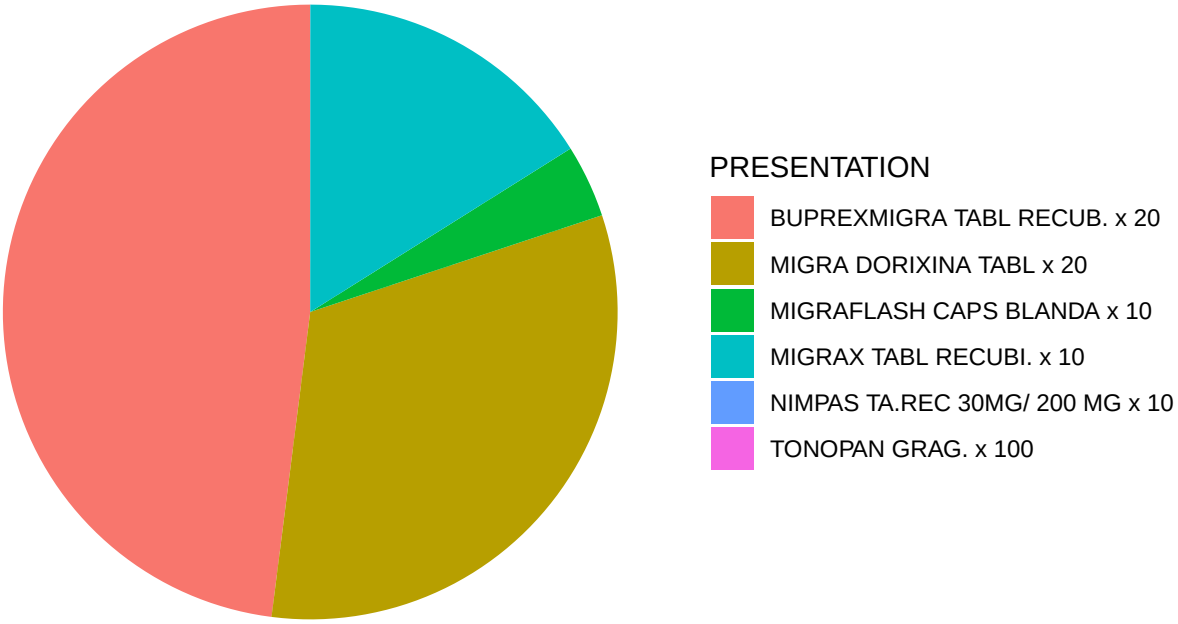
```
## `summarise()` ungrouping output (override with `.groups` argument)
```



As mentioned in the introduction, this project is evaluating 6 compaies and their product version for the same patology (Migraine). In this chart we can see which segment of the whole market is taken by each product:

```
## Market Share of each product
trainingData %>% group_by(PRESENTATION) %>%
summarize( amount = sum(`SALES UNITS`)) %>%
ggplot(aes(x="",y=amount,fill=PRESENTATION)) +
geom_bar(stat="identity", width=1) + coord_polar("y", start=0) + theme_void()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
## Amount of sales per presentation
trainingData %>% group_by(PRESENTATION) %>%
summarize( amount = sum(`SALES UNITS`))
```

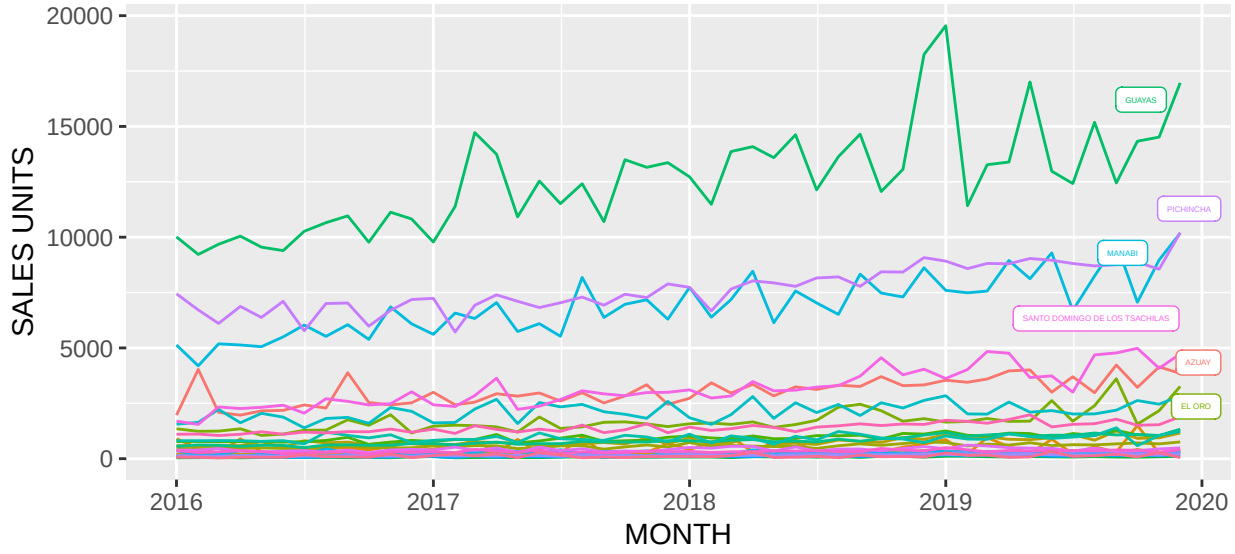
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 6 x 2
##   PRESENTATION                        amount
##   <chr>                                <dbl>
## 1 BUPREXMIGRA TABL RECUB. x 20    2179764.
## 2 MIGRA DORIXINA TABL x 20        1458340.
## 3 MIGRAFLASH CAPS BLANDA x 10      173782.
## 4 MIGRAX TABL RECUBI. x 10         730094.
## 5 NIMPAS TA.REC 30MG/ 200 MG x 10     153.
## 6 TONOPAN GRAG. x 100                 10.0
```

Now, we are going to analyze the sales per province for the 3 laboratories with the best sales rate.

```
# plot sales per province for LIFE laboratories
trainingData %>% filter(LABORATORY=="LIFE") %>%
  ggplot(aes(MONTH,`SALES UNITS`, group = PROVINCE, color = PROVINCE )) +
  geom_line() +
  geom_label_repel(data = subset(trainingData, trainingData$MONTH ==
                                   max(trainingData$MONTH) & trainingData$LABORATORY == "LIFE"),aes(labe
```

```
## Warning: ggrepel: 18 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

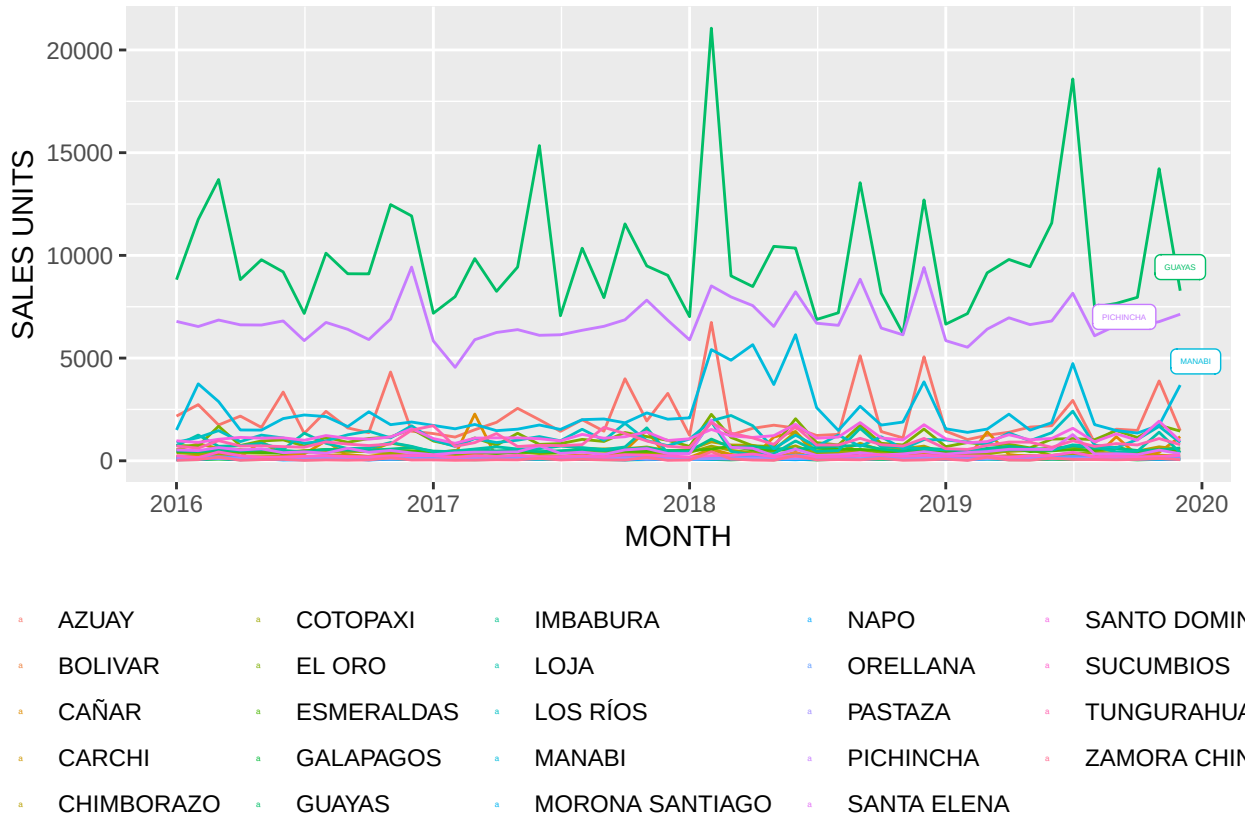


AZUAY	COTOPAXI	IMBABURA	NAPO	SANTO DOMIN
BOLIVAR	EL ORO	LOJA	ORELLANA	SUCUMBIOS
CAÑAR	ESMERALDAS	LOS RÍOS	PASTAZA	TUNGURAHUA
CARCHI	GALAPAGOS	MANABI	PICHINCHA	ZAMORA CHIN
CHIMBORAZO	GUAYAS	MORONA SANTIAGO	SANTA ELENA	

In LIFE Laboratories, the best sales range correspond to the Guayas province

```
# plot sales per province for MEGALABS laboratories
trainingData %>% filter(LABORATORY=="MEGALABS") %>%
  ggplot(aes(MONTH,`SALES UNITS`, group = PROVINCE, color = PROVINCE )) +
  geom_line() +
  geom_label_repel(data = subset(trainingData, trainingData$MONTH == max(trainingData$MONTH) &
                                 trainingData$LABORATORY == "MEGALABS"),aes(label =
  PROVINCE),size=1, nudge_x = 45, segment.color = NA) +
  theme(legend.position="bottom")
```

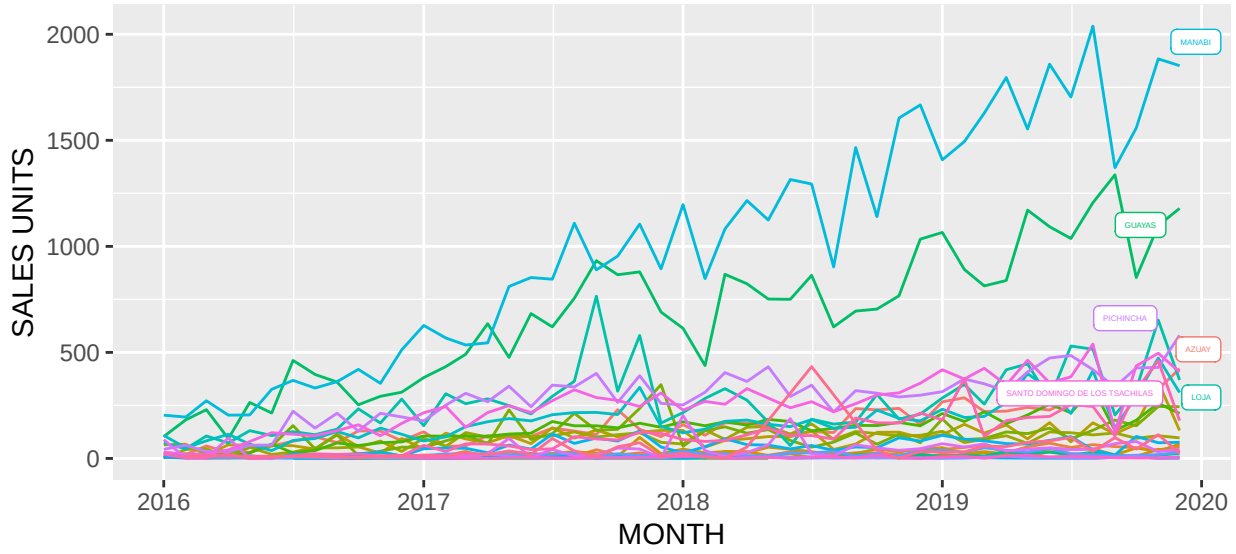
```
## Warning: ggrepel: 21 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Similarly, MEGALABS and their product for Migraine has a consistent position in the Guayas province market.

```
# plot sales per province for JAMES BROWN PHARMA laboratories
trainingData %>% filter(LABORATORY=="JAMES BROWN PHARMA") %>%
  ggplot(aes(MONTH,`SALES UNITS`, group = PROVINCE, color = PROVINCE )) +
  geom_line() +  geom_label_repel(data = subset(trainingData, trainingData$MONTH == max(trainingData$MO
  trainingData$LABORATORY == "JAMES BROWN PHARMA"),aes(label = PROVINCE),size=1, nudge_x = 45,
  segment.color = NA) +theme(legend.position="bottom")
```

```
## Warning: ggrepel: 17 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



AZUAY	COTOPAXI	IMBABURA	NAPO	SANTO DOMIN
BOLIVAR	EL ORO	LOJA	ORELLANA	SUCUMBIOS
CAÑAR	ESMERALDAS	LOS RÍOS	PASTAZA	TUNGURAHUA
CARCHI	GALAPAGOS	MANABI	PICHINCHA	ZAMORA CHIN
CHIMBORAZO	GUAYAS	MORONA SANTIAGO	SANTA ELENA	

On the other hand, JAMES BROWN PHARMA Company best sales correspond to Manabi province.

Methods

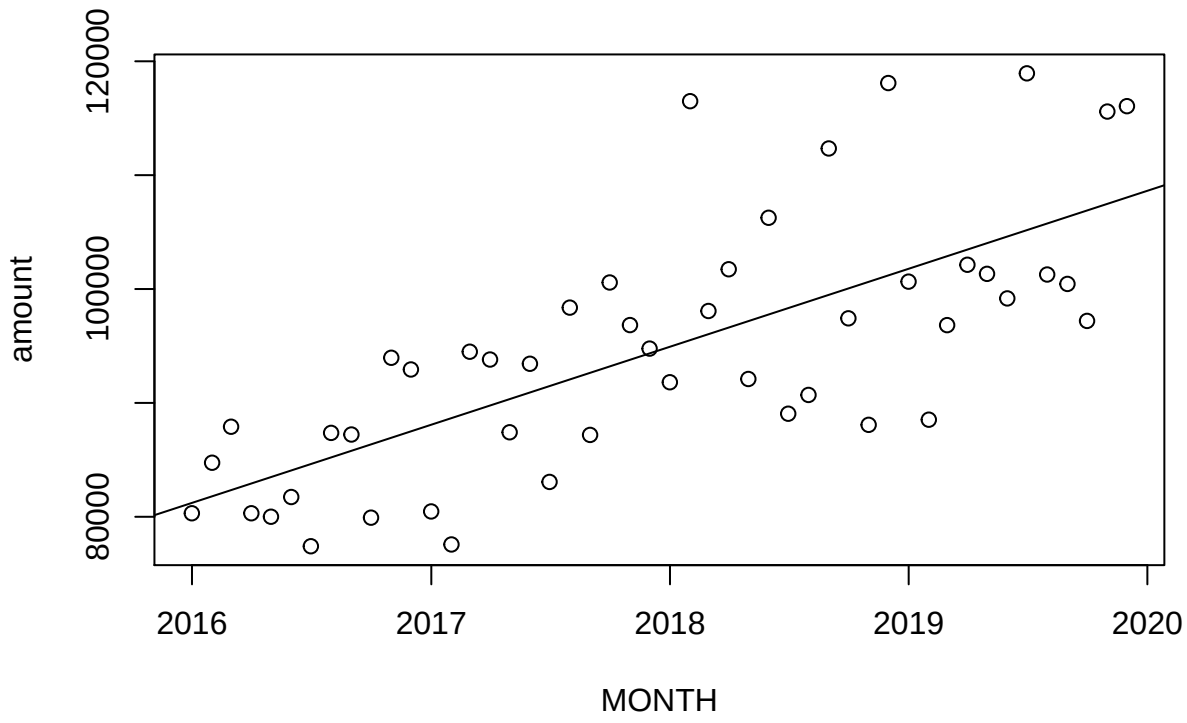
In this project, we will use two models. First, as the Professor Ragazzi taught us, Linear regression is the perfect model to predict some Y values based on X values.

Predicting sales based on the historical data using linear regression:

```
all_sales <- trainingData %>% group_by(MONTH) %>% summarize( amount = sum(`SALES UNITS`))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
linearModel2 = lm(amount ~ MONTH , data= all_sales)
plot(amount ~ MONTH , data= all_sales)
abline(linearModel2)
```




```
## Summary
summary(linearModel2)
```

```
##
## Call:
## lm(formula = amount ~ MONTH, data = all_sales)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -13832  -5561  -1229   5338  20987
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.340e+05  4.658e+04  -5.023 8.15e-06 ***
## MONTH        2.171e-04  3.077e-05   7.056 7.50e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7762 on 46 degrees of freedom
## Multiple R-squared:  0.5198, Adjusted R-squared:  0.5093
## F-statistic: 49.79 on 1 and 46 DF,  p-value: 7.504e-09
```

```
##training our model
modelFit <- train(amount ~ MONTH , data = all_sales)
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
```

```
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
```

```
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range
```

```
modelFit
```

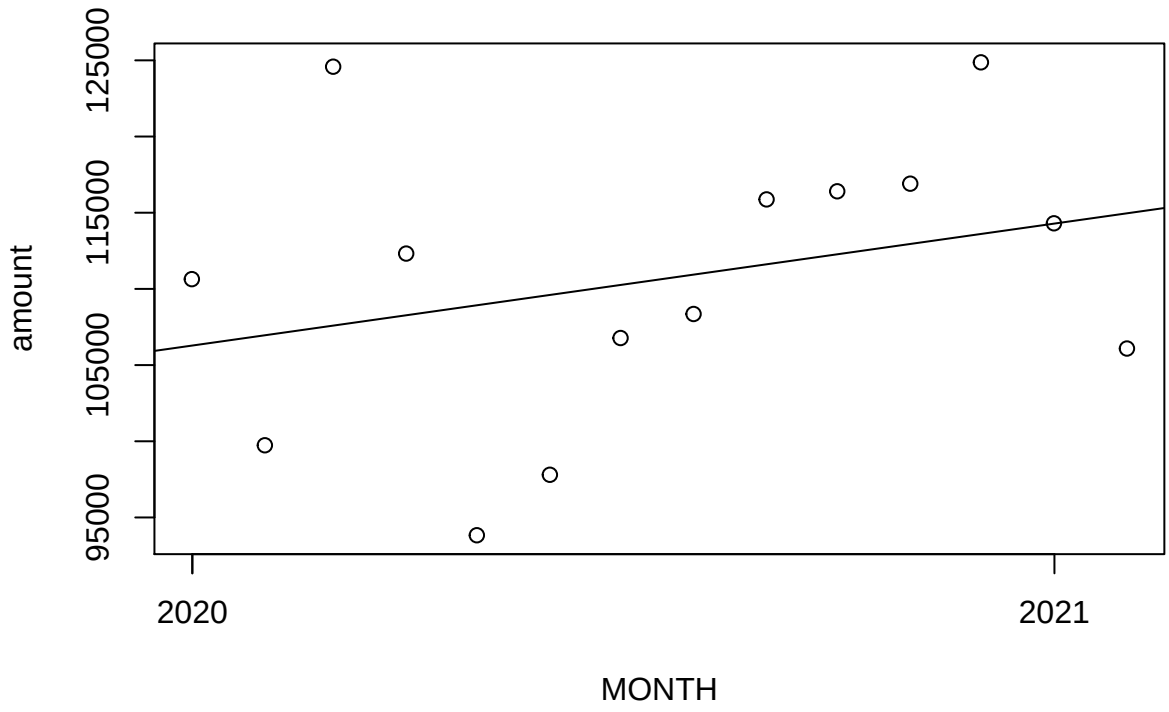
```
## Random Forest
##
## 48 samples
##  1 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 48, 48, 48, 48, 48, 48, ...
## Resampling results:
##
##   RMSE     Rsquared   MAE
##   10050.6  0.3186043  8153.224
##
## Tuning parameter 'mtry' was held constant at a value of 2
```

Now we are going to apply the linear regression in the test set

```
all_sales_test <- testData %>% group_by(MONTH) %>% summarize( amount = sum(`SALES UNITS`))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
linearModel3 = lm(amount ~ MONTH , data= all_sales_test)
plot(amount ~ MONTH , data= all_sales_test)
abline(linearModel3)
```




```
## Summary
summary(linearModel3)
```

```
##
## Call:
## lm(formula = amount ~ MONTH, data = all_sales_test)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -15091  -6284   1990   4228  16993
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.931e+05  3.693e+05  -0.794    0.443
## MONTH        2.531e-04  2.316e-04   1.093    0.296
##
## Residual standard error: 9213 on 12 degrees of freedom
## Multiple R-squared:  0.09055,    Adjusted R-squared:  0.01476
## F-statistic: 1.195 on 1 and 12 DF,  p-value: 0.2958
```

```
## test Data
modelFit2 <- train(amount ~ MONTH , data = all_sales_test)
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid mtry:
## reset to within valid range
```

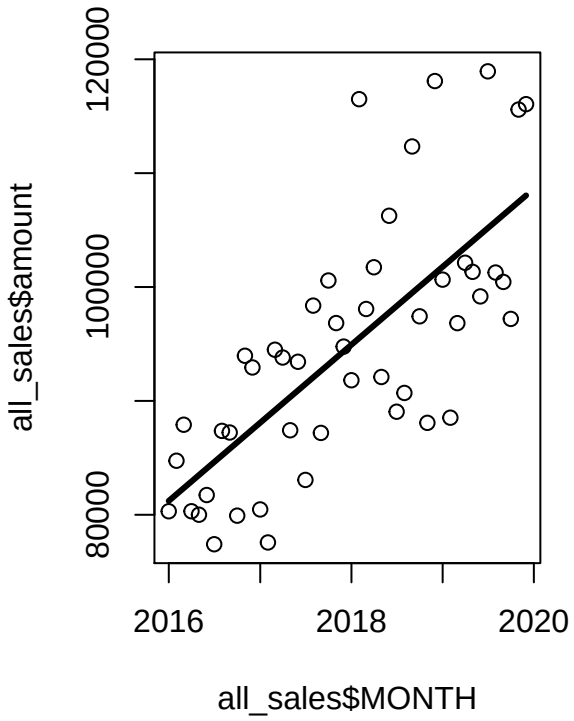
```
modelFit2
```

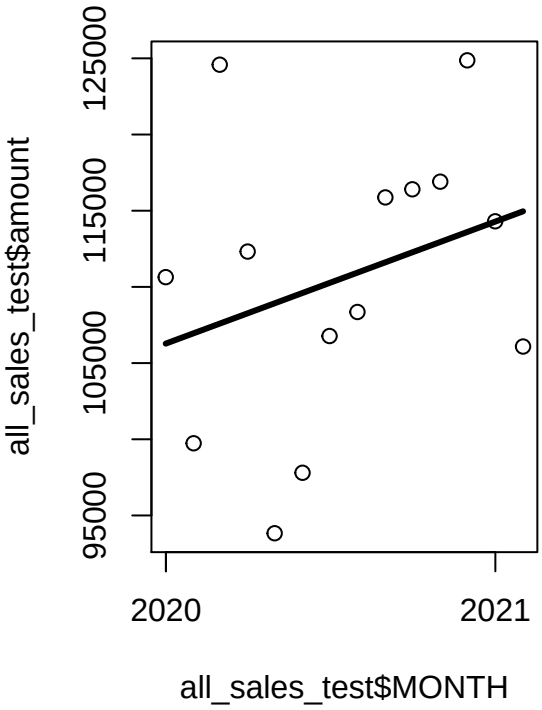
```
## Random Forest
##
## 14 samples
##  1 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14, 14, 14, 14, 14, 14, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   10353.69  0.2127389  8639.598
##
## Tuning parameter 'mtry' was held constant at a value of 2
```

Comparing the 2 data sets:

```
par(mfrow=c(1,2))
plot(all_sales$MONTH,all_sales$amount)
lines(all_sales$MONTH,predict(linearModel2),lwd=3)

plot(all_sales_test$MONTH,all_sales_test$amount)
lines(all_sales_test$MONTH,predict(linearModel3),lwd=3)
```





Results

We fit our regression model to predict de amount of sales based on time. We have the followings RMSE values

```
modelFit
```

```
## Random Forest
##
## 48 samples
##  1 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 48, 48, 48, 48, 48, 48, ...
## Resampling results:
##
##   RMSE     Rsquared   MAE
##   10050.6  0.3186043  8153.224
##
## Tuning parameter 'mtry' was held constant at a value of 2
```

```
modelFit2
```

```
## Random Forest
```

```
##
## 14 samples
##  1 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14, 14, 14, 14, 14, 14, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   10353.69  0.2127389  8639.598
##
## Tuning parameter 'mtry' was held constant at a value of 2
```

Conclusion

1. The relationship between "Sales Units" and "RX" prescriptions is not linear.
2. We can conclude that the linear model is not perfectly accurate because it has a high RMSE.
3. The RMSE shows us how far from the regression line is our data.
4. The data is not concentrated around the line of best fit

Vocabulary

Migraine: A migraine is usually a moderate or severe headache felt as a throbbing pain on 1 side of the head.