Leading Indicators for LJ Kirkland

Preliminary Analysis

Keaton Markey

04 December, 2020

Contents

0.0.0.1 Statement From the last 5 years of data, we are trying to find external indicators of a variety of business metrics within the Lee Johnson Auto Family's Kirkland location. I will gather a number of longitudinal economic data sets and match them against a few metrics provided by the company.

0.0.0.2 Lets begin with librarying in packages that we will need

0.0.1 1. Packages

0.0.1.1 These will allow us to plot, impute missing data, and search free data.

0.0.1.2 tidyverse(), mice(), and Quandl()

```
KDARaw <- read.csv("C:/Users/keato/Dropbox/Shop Data/Keaton Data Analysis Project-2016-2020.csv",
                   na.strings = c("", "-", "--", "---", "    -   ", " -   ", " ", "  ", strip.white = TR
colnames(KDARaw) <- (c("Date", "DealNum", "VStock", "Year", "Make",
                       "Model", "NU", "Front_Gross_Profit", "Back_Gross_Profit",
                       "Total_Gross_Profit", "Cash_Price", "PL", "Sale_Type",
```

```
                        "Salesman", "Salesmanager", "FIManager"))
#remove first row
KDARaw<-as.data.frame(KDARaw)
KDARaw<- KDARaw[-c(1),]
#remove parenthesis and change parse
pear<- function(x){
  x<- gsub("[()]", "", x)
  x<- gsub(" ", "", x)
  x<- as.numeric(gsub(",", "", x))
}
KDARaw$Front_Gross_Profit<-pear(KDARaw$Front_Gross_Profit)
KDARaw$Back_Gross_Profit<-pear(KDARaw$Back_Gross_Profit)
KDARaw$Total_Gross_Profit<- pear(KDARaw$Total_Gross_Profit)
KDARaw$Cash_Price<-pear(KDARaw$Cash_Price)
KDARaw$Date<-as.Date(KDARaw$Date, format = "%m/%d/%Y")
KDARaw$PL<-as.factor(KDARaw$PL)
KDARaw<- arrange(KDARaw, Date)
#create a test df
KDA1<-KDARaw
```

0.0.1.3 Read in data

0.0.2 2. Summary statistics

```
summary(KDA1)
```

```
  Date                DealNum              VStock               Year
```

Min. :2016-01-01 Length:19552 Length:19552 Length:19552
1st Qu.:2017-02-18 Class :character Class :character Class :character
Median :2018-03-17 Mode :character Mode :character Mode :character
Mean :2018-04-16
3rd Qu.:2019-06-07
Max. :2020-11-16

```
 Make               Model                NU             Front_Gross_Profit
```

Length:19552 Length:19552 Length:19552 Min. : 0.04
Class :character Class :character Class :character 1st Qu.: 477.98
Mode :character Mode :character Mode :character Median : 1014.00
Mean : 1282.53
3rd Qu.: 1798.78
Max. :16992.60
NA's :205
Back_Gross_Profit Total_Gross_Profit Cash_Price PL
Min. : 0.01 Min. : 0.41 Min. : 0.02 L: 3018
1st Qu.: 476.86 1st Qu.: 890.10 1st Qu.: 18601.00 P:16534
Median : 1081.61 Median : 1840.36 Median : 26495.00
Mean : 1439.08 Mean : 2248.79 Mean : 27453.33
3rd Qu.: 2112.00 3rd Qu.: 3166.55 3rd Qu.: 34095.00
Max. :10287.98 Max. :18464.00 Max. :148585.00
NA's :3058 NA's :38 NA's :251
Sale_Type Salesman Salesmanager FIManager
Length:19552 Length:19552 Length:19552 Length:19552
Class :character Class :character Class :character Class :character

Mode :character Mode :character Mode :character Mode :character

```
str(KDA1)
```

'data.frame': 19552 obs. of 16 variables: $ Date : Date, format: "2016-01-01" "2016-01-02" … $ DealNum : chr "134833" "134966" "135381" "135380" … $ VStock : chr "6091571" "609295" "M160907" "K1604151" … $ Year : chr "10" "15" "16" "14" … $ Make : chr "CHEV" "CADI" "MAZD" "KIA" … $ Model : chr "CAMAC" "SRX" "MAZDA3" "SORENT" … $ NU : chr "USED" "USED" "NEW" "USED" … $ Front_Gross_Profit: num 1312 1822 1567 3501 1617 … $ Back_Gross_Profit : num NA 5246 4101 1998 3479 … $ Total_Gross_Profit: num 1312 7067 5668 5499 5096 … $ Cash_Price : num 20781 36400 21415 30995 22995 … $ PL : Factor w/ 2 levels "L","P": 2 2 2 2 2 2 2 2 2 2 … $ Sale_Type : chr NA NA NA NA … $ Salesman : chr "WERNER,RICHARD T" "MIKOLASY,ROBERT A" "ROCHE,NICHOLAS B" "ARIAS,BRAULIO" … $ Salesmanager : chr "THOENSEN" "THOENSEN" "ANDREWS" "THOENSEN" … $ FIManager : chr "GOLDMAN" "MILLER" "GREENE JR" "GOLDMAN" …

```
glimpse(KDA1)
```

Rows: 19,552 Columns: 16 $ Date 2016-01-01, 2016-01-02, 2016-01-02, 2016-01-02,… $ DealNum "134833", "134966", "135381", "135380", "135401"… $ VStock "6091571", "609295", "M160907", "K1604151", "609… $ Year "10","15","16","14","15","16","11","16", … $ Make "CHEV","CADI","MAZD","KIA","CHEV","MAZD","… $ Model "CAMAC", "SRX", "MAZDA3", "SORENT", "EQUIN", "CX… $ NU "USED","USED","NEW","USED","USED","U $ Front_Gross_Profit 1311.73, 1821.68, 1567.00, 3500.84, 1617.22, 106… $ Back_Gross_Profit NA, 5245.62, 4100.74, 1998.48, 3478.80, 3691.52,… $ Total_Gross_Profit 1311.73, 7067.30, 5667.74, 5499.32, 5096.02, 475… $ Cash_Price 20781.19, 36400.00, 21415.00, 30995.00, 22995.00… $ PL P, P, P, P, P, P, P, P, P, P, P, P, P, P, P, P, … $ Sale_Type NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, … $ Salesman "WERNER,RICHARD T", "MIKOLASY,ROBERT A", "ROCHE,… $ Salesmanager "THOENSEN","THOENSEN","ANDREWS","THOENSEN","… $ FIManager "GOLDMAN", "MILLER", "GREENE JR", "GOLDMAN", "RO…

0.0.2.1 We have some NA's, or empty spaces, in the data. This could be for a

variety of reasons, someone didn't enter in a number, it got lost in translation, or just didn't exist in the first place. Before we deal with these missing values, let's see how much they affect our data. We really only care about NAs in numerical columns, because thats when they could affect the stats we will run later.

0.0.2.2 We can use logistic regression to tell us how much the NAs might influence

the data longitudinally. This is probably unnecessary but I'm going to do it anyway.

```
summary(glm(is.na(Back_Gross_Profit) ~ Date, KDA1, family = 'binomial'))
```

Call: glm(formula = is.na(Back_Gross_Profit) ~ Date, family = "binomial", data = KDA1)

Deviance Residuals: Min 1Q Median 3Q Max
-0.5957 -0.5884 -0.5812 -0.5732 1.9494

Coefficients: Estimate Std. Error z value Pr(>|z|) (Intercept) -7.043e-01 6.964e-01 -1.011 0.312 Date -5.563e-05 3.949e-05 -1.409 0.159

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 16958  on 19551  degrees of freedom
```

Residual deviance: 16956 on 19550 degrees of freedom AIC: 16960

Number of Fisher Scoring iterations: 3

```
summary(glm(is.na(Front_Gross_Profit) ~ Date, KDA1, family = 'binomial'))
```

Call: glm(formula = is.na(Front_Gross_Profit) ~ Date, family = "binomial", data = KDA1)

Deviance Residuals: Min 1Q Median 3Q Max
-0.1933 -0.1618 -0.1392 -0.1221 3.2141

Coefficients: Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.640e+01 2.525e+00 -6.492 8.45e-11 ***Date 6.687e-04 1.418e-04 4.715 2.41e-06*** — Signif. codes: 0 '*' ***0.001*** '**' *0.01* '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 2276.6  on 19551  degrees of freedom
```

Residual deviance: 2254.0 on 19550 degrees of freedom AIC: 2258

Number of Fisher Scoring iterations: 7

```
summary(glm(is.na(Total_Gross_Profit) ~ Date, KDA1, family = 'binomial'))
```

Call: glm(formula = is.na(Total_Gross_Profit) ~ Date, family = "binomial", data = KDA1)

Deviance Residuals: Min 1Q Median 3Q Max
-0.0645 -0.0634 -0.0624 -0.0613 3.5542

Coefficients: Estimate Std. Error z value Pr(>|z|) (Intercept) -4.808e+00 5.751e+00 -0.836 0.403 Date -8.132e-05 3.263e-04 -0.249 0.803

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 550.41  on 19551  degrees of freedom
```

Residual deviance: 550.35 on 19550 degrees of freedom AIC: 554.35

Number of Fisher Scoring iterations: 9

```
summary(glm(is.na(Cash_Price) ~ Date, KDA1, family = 'binomial'))
```

Call: glm(formula = is.na(Cash_Price) ~ Date, family = "binomial", data = KDA1)

Deviance Residuals: Min 1Q Median 3Q Max
-0.2388 -0.1845 -0.1487 -0.1227 3.2430

Coefficients: Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.151e+01 2.349e+00 -9.157 < 2e-16 ***Date 9.668e-04 1.314e-04 7.358 1.87e-13*** — Signif. codes: 0 '*' ***0.001*** '**' *0.01* '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 2685.2  on 19551  degrees of freedom
```

Residual deviance: 2628.6 on 19550 degrees of freedom AIC: 2632.6

Number of Fisher Scoring iterations: 7 #### Based on the models, we noticed a significant trend for all the metrics. It shouldn't impact analysis too much because we didn't test numeric variables, but its something to keep in mind

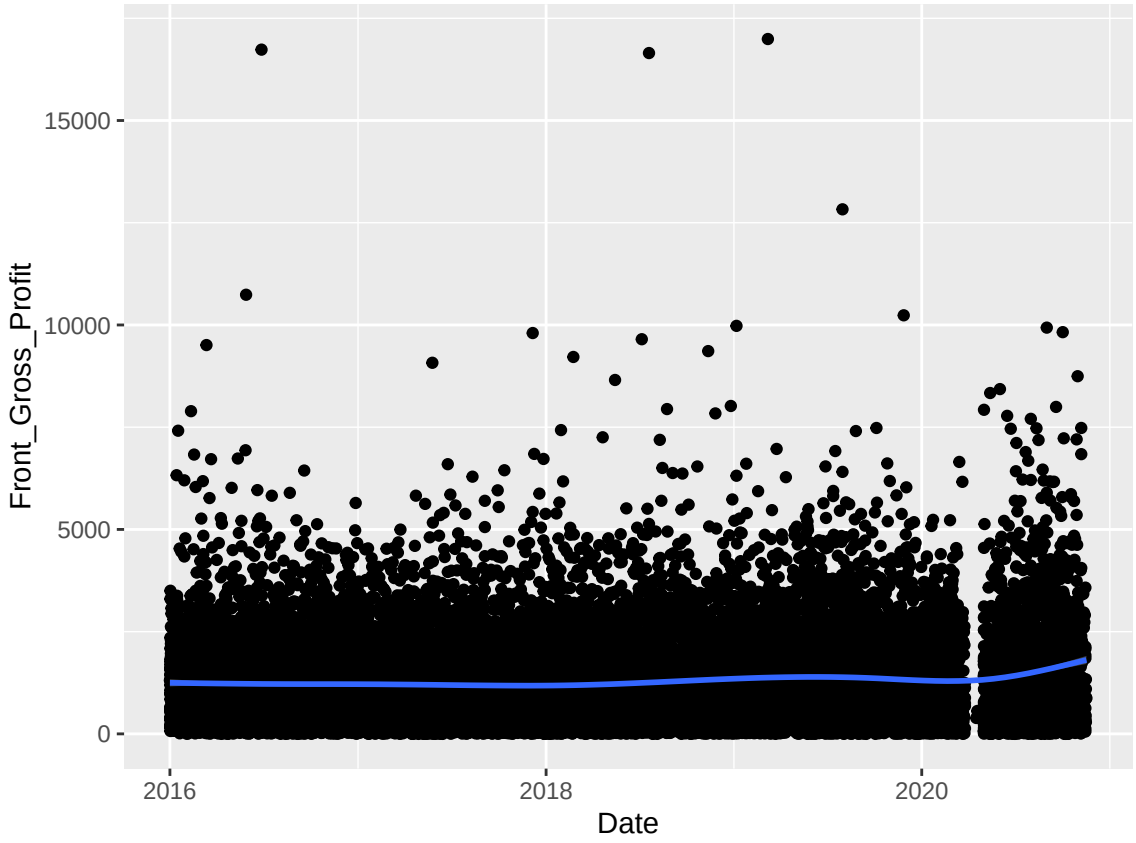
Generally, we saw more missing Front Gross Profit values and Cash Price values at the end of the data set, and we saw more missing values for Back Gross Profit and Total Gross Profit at the beginning of the data set.

0.0.3 3. Plotting

```
ggplot(KDA1, aes(Date, Front_Gross_Profit)) + geom_point() + geom_smooth()
```

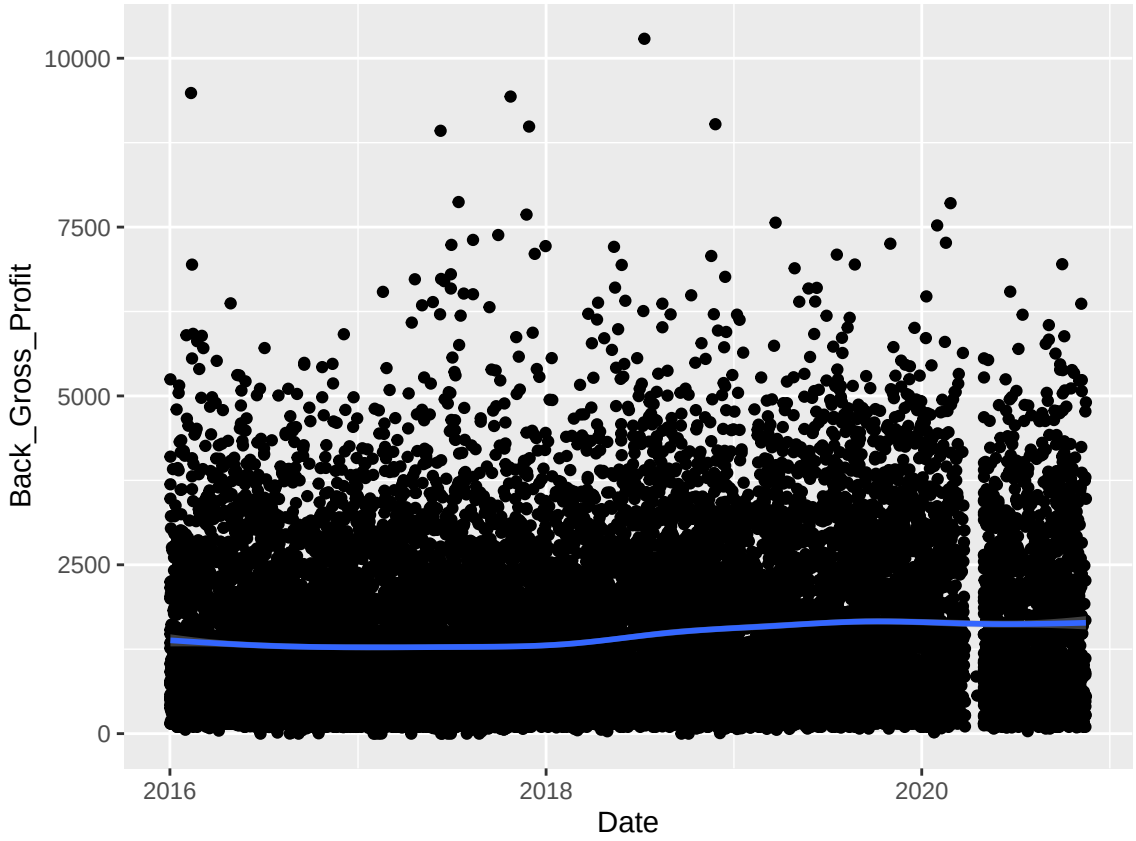
0.0.3.1 Now, lets look at some quick plots of the data.

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



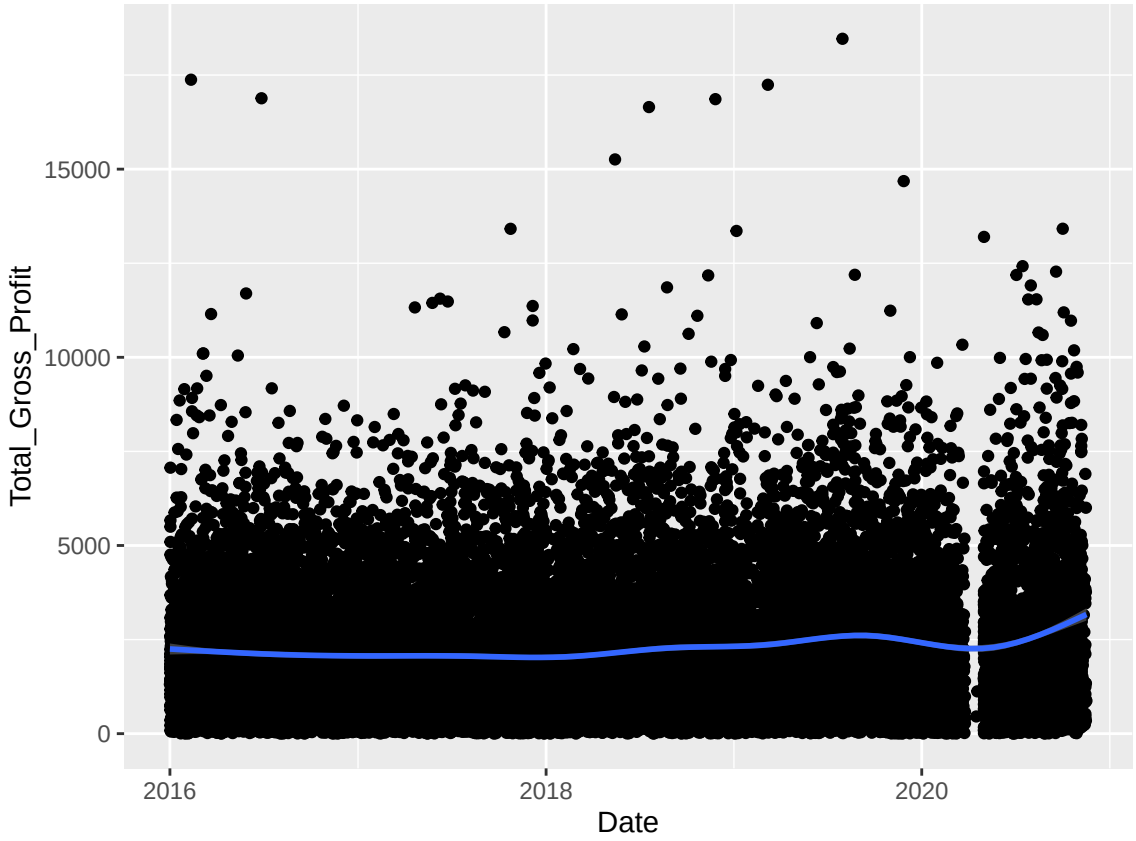

```
ggplot(KDA1, aes(Date, Back_Gross_Profit)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



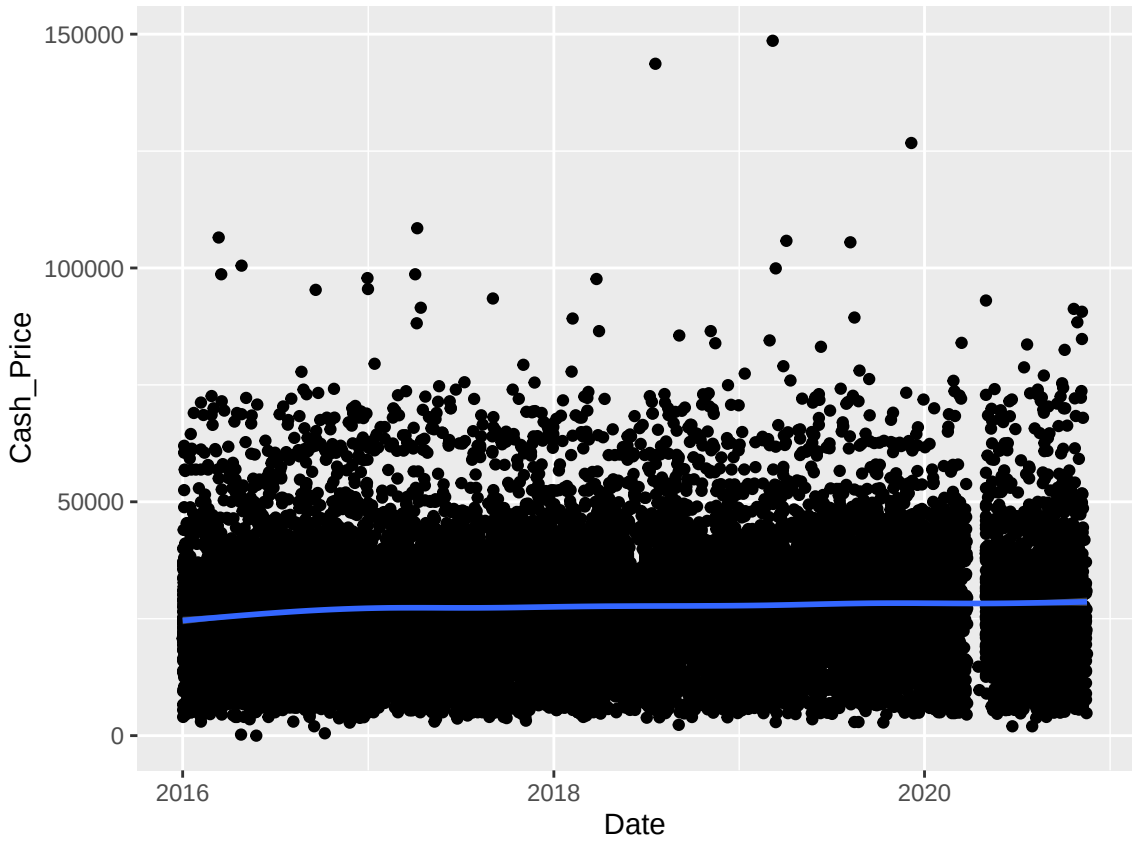

```
ggplot(KDA1, aes(Date, Total_Gross_Profit)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
ggplot(KDA1, aes(Date, Cash_Price)) + geom_point() + geom_smooth()
```

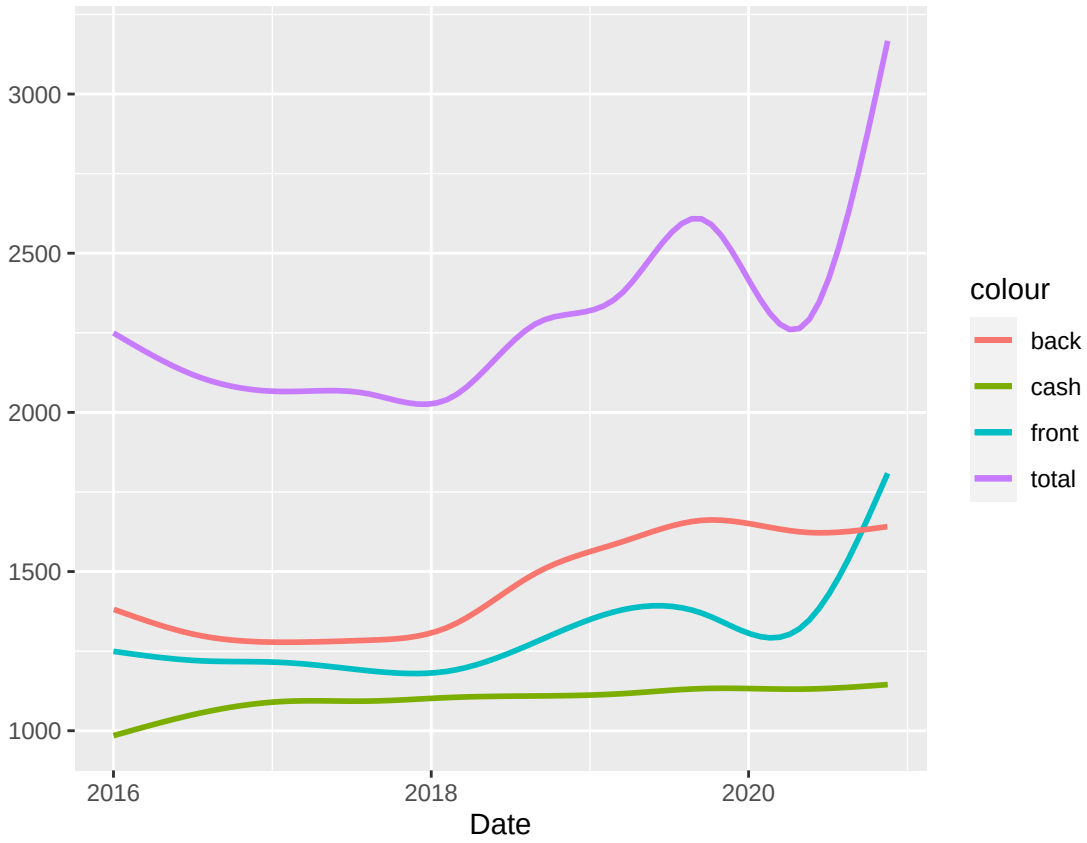
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



0.0.3.2 NOt too shabby. Things are a little hard to see with all 20,000 of these data points, but the smoothing spline helps a bit. But, we can do even better.

```
ggplot(KDA1, aes(x = Date)) + geom_smooth(aes(y = Front_Gross_Profit, col = "front"), se = F) +
  geom_smooth(aes(y = Back_Gross_Profit, col = "back"), se = F) + ylab("") +
  geom_smooth(aes(y = Total_Gross_Profit, col = "total"), se = F) +
  geom_smooth(aes(y = .04*Cash_Price, col = "cash"), se = F)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



0.0.3.3 If we hope to do anything with this, we will have to extract these trends. To make this easier, lets create a new df for each day.

0.0.4 4. Condensing

```
Day<- as.data.frame(seq(as.Date("2016-01-01"), as.Date("2020-11-16"), by="days"))
colnames(Day)<-c("Date")
temp<- group_by(KDA1, Date) %>%
  mutate(sum.fgp = sum(Front_Gross_Profit, na.rm = TRUE)) %>%
  summarise(sum.fgp, .groups = 'drop') %>%
  unique()
Day<-left_join(Day, count(group_by(KDA1, Date)), by = "Date")
Day<-left_join(Day, temp, by = "Date")
temp<- group_by(KDA1, Date) %>%
  mutate(sum.bgp= sum(Back_Gross_Profit, na.rm = TRUE)) %>%
  summarise(sum.bgp, .groups = 'drop') %>%
  unique()
Day<-left_join(Day, temp, by = "Date")
temp<- group_by(KDA1, Date) %>%
  mutate(sum.tgp = sum(Total_Gross_Profit, na.rm = TRUE)) %>%
  summarise(sum.tgp, .groups = 'drop') %>%
  unique()
Day<-left_join(Day, temp, by = "Date")
temp<- group_by(KDA1, Date) %>%
  mutate(sum.cp = sum(Cash_Price, na.rm = TRUE)) %>%
```

```
  summarise(sum.cp, .groups = 'drop') %>%
  unique()
Day<-left_join(Day, temp, by = "Date")
```

0.0.4.1 There are 1782 days, with 55 NAs in the mix. Most of them are probably within

that stretch in April of 2019. We will fix them by imputation. First lets take a look at this new df, now with the number of sales per day [n]. The missing values will automatically be predicted using a simple linear model (loess).

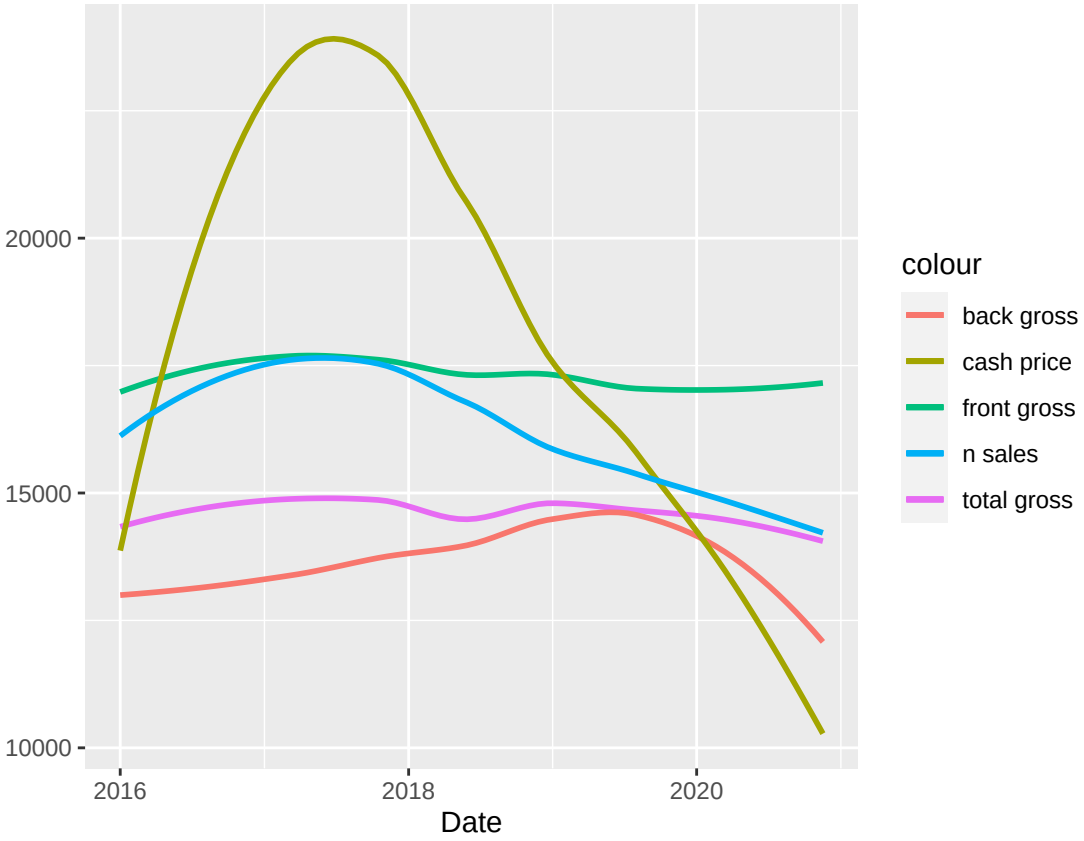
```
summary(Day)
```

```
  Date                 n            sum.fgp           sum.bgp
```

Min. :2016-01-01 Min. : 1.00 Min. : 388.9 Min. : 0
1st Qu.:2017-03-21 1st Qu.: 7.00 1st Qu.: 8334.3 1st Qu.: 7508
Median :2018-06-09 Median :10.00 Median :12620.6 Median :11736
Mean :2018-06-09 Mean :11.32 Mean :14367.7 Mean :13744
3rd Qu.:2019-08-28 3rd Qu.:14.00 3rd Qu.:18541.5 3rd Qu.:17866
Max. :2020-11-16 Max. :56.00 Max. :64659.8 Max. :88741
NA's :55 NA's :55 NA's :55
sum.tgp sum.cp
Min. : 456.9 Min. : 9741
1st Qu.: 14816.4 1st Qu.: 184423
Median : 22530.0 Median : 271831
Mean : 25409.9 Mean : 306819
3rd Qu.: 32343.8 3rd Qu.: 392585
Max. :131013.3 Max. :1857949
NA's :55 NA's :55

```
Day %>%
ggplot(aes(x = Date)) +
  geom_smooth(aes(y =.6*sum.tgp-600, color = "total gross"),
              se = F, method = 'loess') +
  geom_smooth(aes(y = sum.bgp, color = "back gross"),
              se = F, method = 'loess') +
  geom_smooth(aes(y = .6*sum.fgp + 8700, color = "front gross"),
              se = F, method = 'loess') +
  geom_smooth(aes(y = .15*(sum.cp-183000), color = "cash price"),
              se = F, method = 'loess') +
  geom_smooth(aes(y = 1000*n+5000, color = "n sales"),
              se = F, method = 'loess') + ylab("")
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

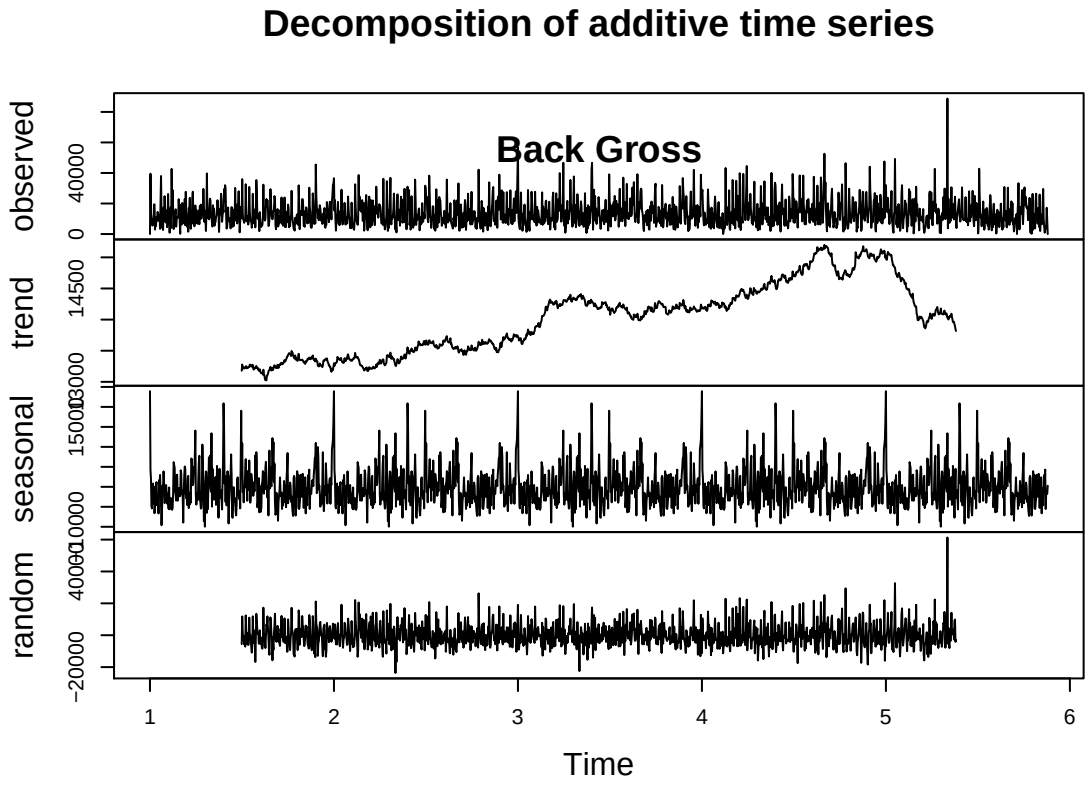


0.0.5 5. Imputation

0.0.5.1 I've chosen to impute the mssing values so we can get a complete df that we can run through functions and expect a complete output. To do this, we can use a variety of methods. I've chosen to use predictive mean matching, which uses a kind of regression that builds relationships around the mean of each variable. Again, there are many ways to do this and this might not be the best one. First, we train the algorithm 20 times [m = 20], and we get its best guess at what the missing values would be according to pmm. We can easily change the method later too.

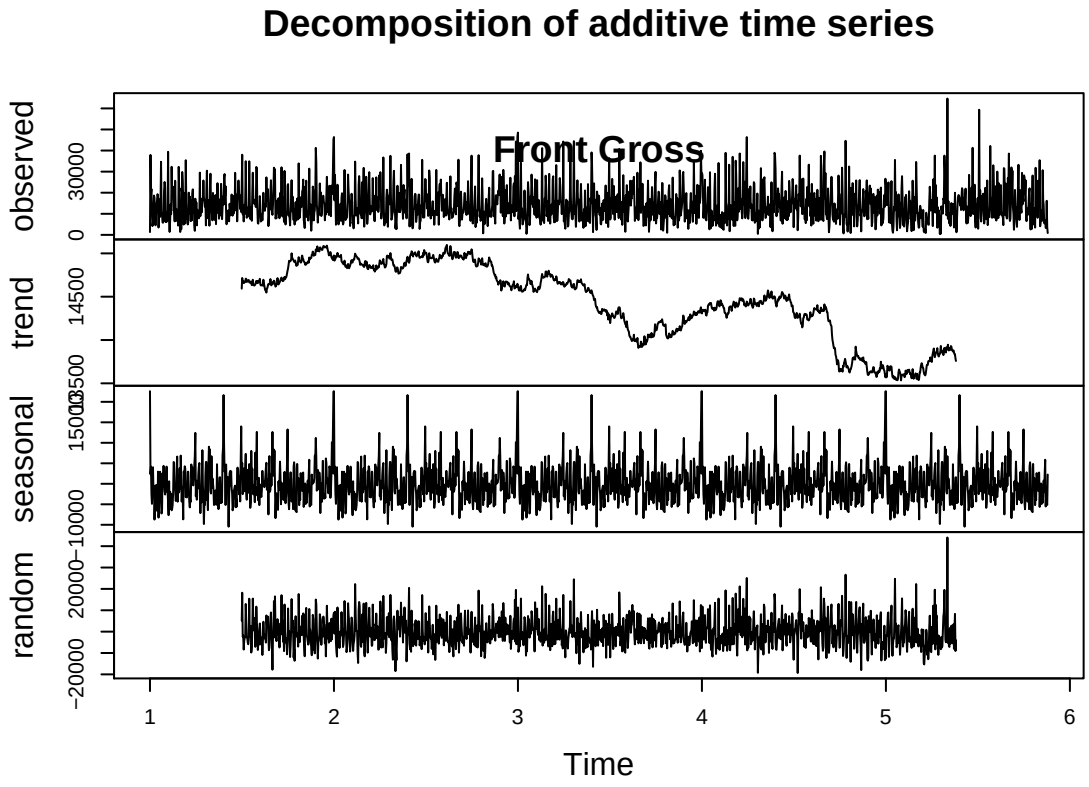
0.0.5.2 Now, since we have data for each day, we can do some timeries analysis for all of our metrics. [freq = 365] because we have 365 observations per year. Remember here, these values are taken from the sum of that value every day. From, these plots, we really just need to pay attention to the trend.

```
b<-decompose(ts(Day.full$sum.bgp, frequency =365))
plot(b) + title("Back Gross", line = -1)
```



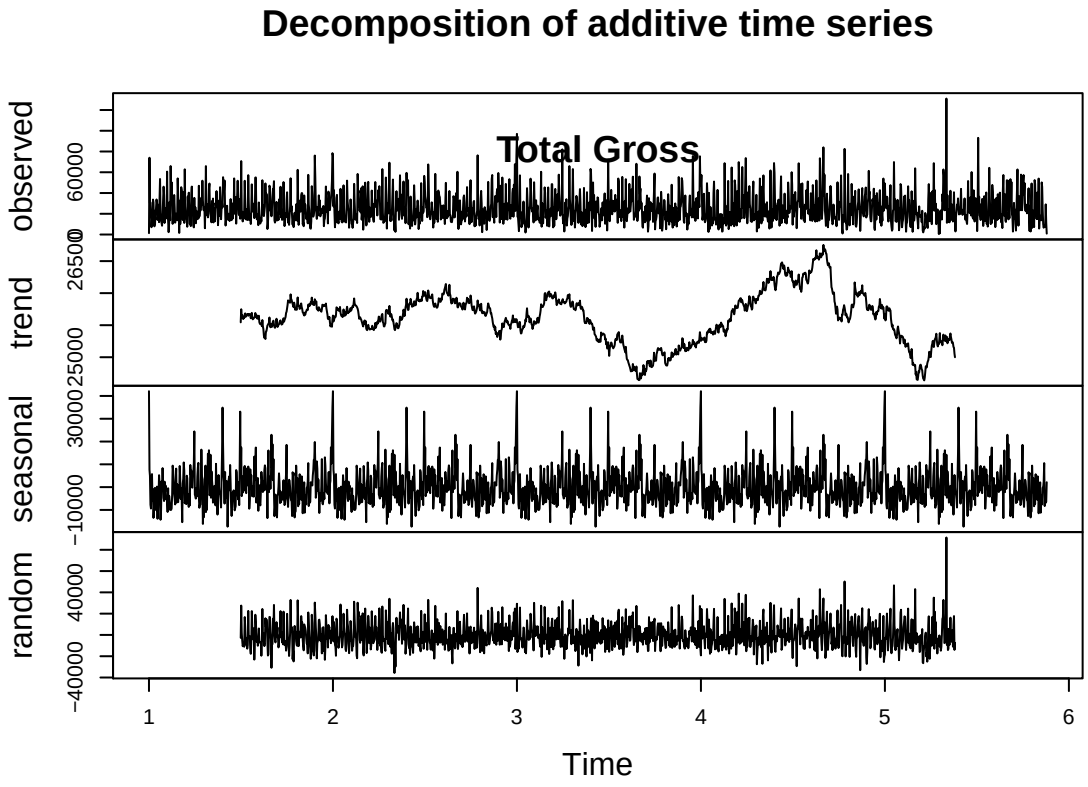
integer(0)

```
f<-decompose(ts(Day.full$sum.fgp, frequency =365))
plot(f)+ title("Front Gross", line = -1)
```



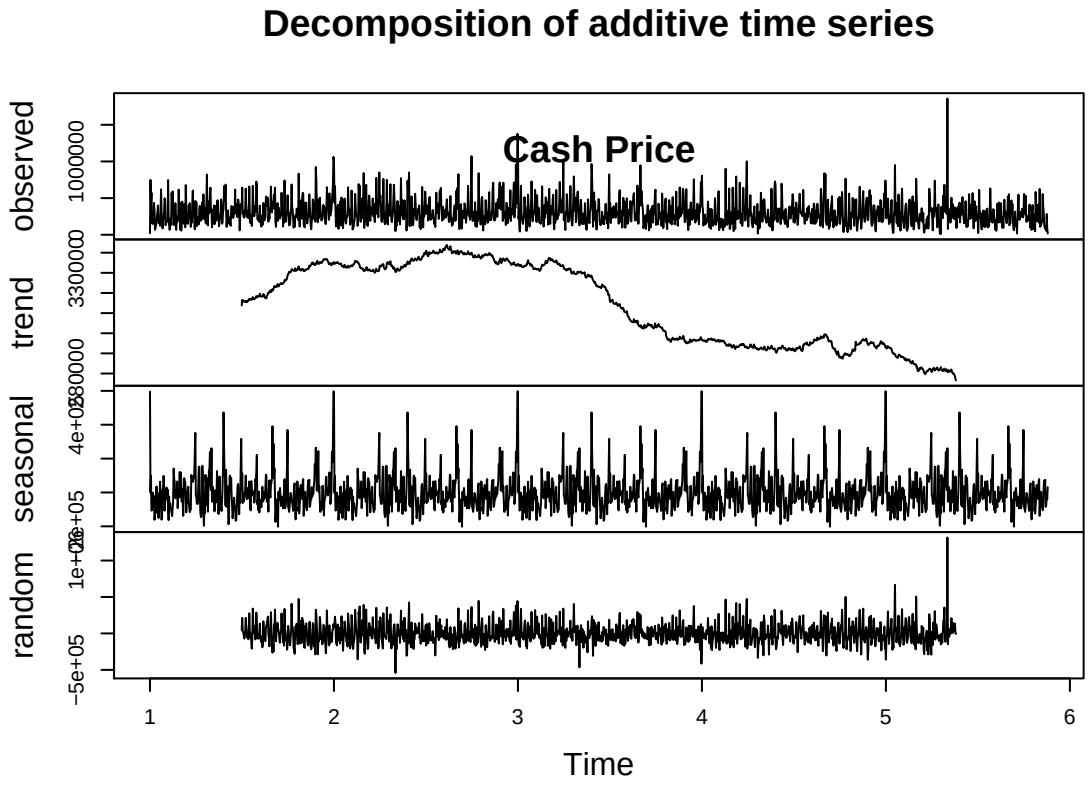
integer(0)

```
t<-decompose(ts(Day.full$sum.tgp, frequency =365))
plot(t) + title("Total Gross", line = -1)
```



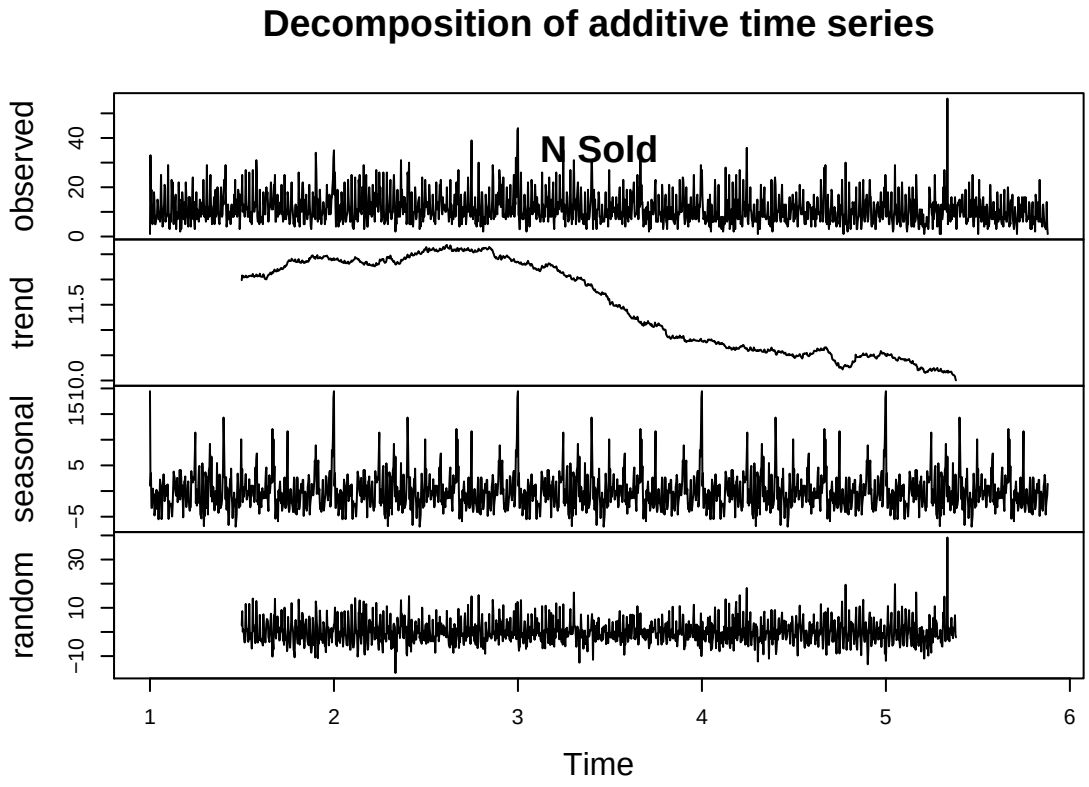
integer(0)

```
c<-decompose(ts(Day.full$sum.cp, frequency =365))
plot(c) + title("Cash Price", line = -1)
```



integer(0)

```
n<-decompose(ts(Day.full$n, frequency =365))
plot(n) + title("N Sold", line =  -1)
```

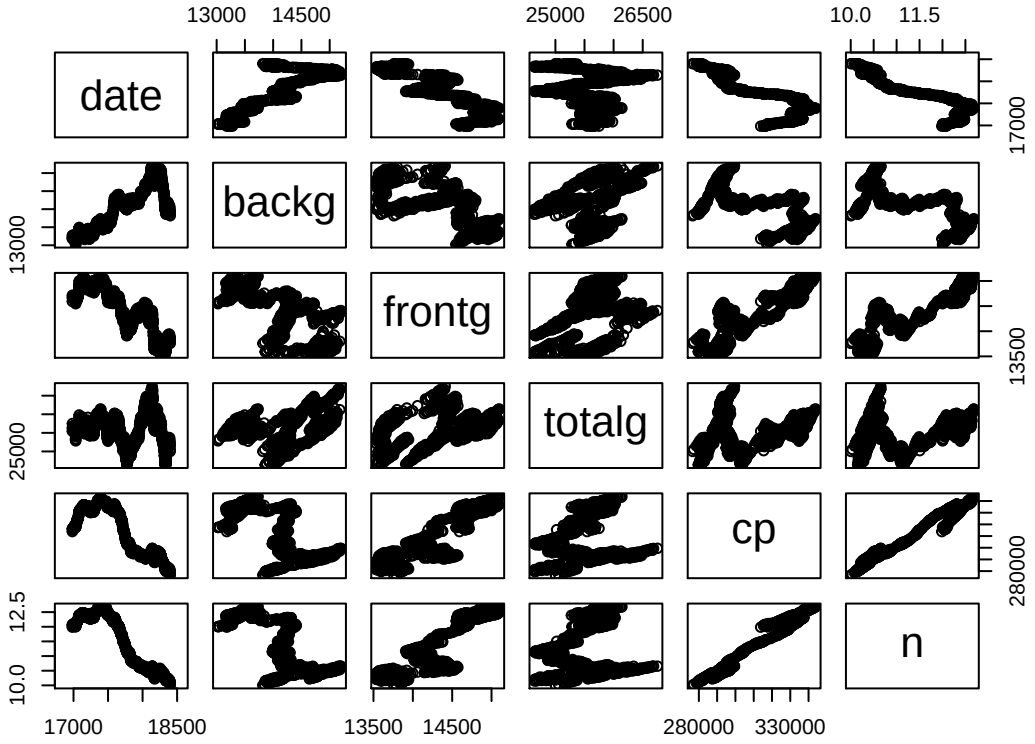


integer(0) #### Looks nice and informative. Now, lets make sure all our data is in the standard style so we can easily compare it to other economic trends.

```
#dummy for joining
time.seq<-seq.Date(from = as.Date("2016-01-01"), to = as.Date("2020-11-16"), by = 'day')
blank<-as.data.frame(time.seq)
colnames(blank)<-c("date")
## Then the final df
rogue<- cbind(as.data.frame(Day.full$Date), as.data.frame(b$trend),
              as.data.frame(f$trend),
              as.data.frame(t$trend), as.data.frame(c$trend),
              as.data.frame(n$trend))
colnames(rogue)<-c("date", "backg", "frontg", "totalg", "cp", "n")
rogue<-left_join(blank, rogue)
```

```
## Joining, by = "date"
```

```
#Finally, a very informative plot of our trends so far
plot(rogue)
```


date
backg
frontg
totalg
cp
n
13000
14500
25000
26500
10.0
11.5
17000
13000
13500
25000
280000
12.5
10.0
17000
18500
13500
14500
280000
330000