Capestone Project

James Strayer

02/01/2020

# Introduction

I was asked by a manager of Restaurant that does retail, to have a model to predict his sales for a given week, he gave me a file with the sales since end of March 2015. He started as the manager on 01-10-2018

## Data import and cleaning

## Loading required package: tidyverse

## -- Attaching packages ----------------------------------------------------- tidyverse 1.2.1 --

## <U+2713> ggplot2 3.2.1 <U+2713> purrr 0.3.2  
## <U+2713> tibble 2.1.3 <U+2713> dplyr 0.8.3  
## <U+2713> tidyr 0.8.3 <U+2713> stringr 1.4.0  
## <U+2713> readr 1.3.1 <U+2713> forcats 0.4.0

## -- Conflicts -------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

## Parsed with column specification:  
## cols(  
## Date = col\_date(format = ""),  
## Weekday = col\_character(),  
## Year = col\_double(),  
## Month = col\_character(),  
## Day = col\_double(),  
## `annee fiscale` = col\_character(),  
## Sales = col\_number(),  
## Retail = col\_number(),  
## TakeOutSales = col\_number(),  
## Bar\_Sales = col\_number(),  
## Sales\_Restaurant = col\_number()  
## )

## Observations: 1,715  
## Variables: 11  
## $ Date <date> 2015-03-29, 2015-03-30, 2015-03-31, 2015-04-01, 201…  
## $ Weekday <chr> "SUNDAY", "MONDAY", "TUESDAY", "WEDNESDAY", "THURSDA…  
## $ Year <dbl> 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015…  
## $ Month <chr> "March'15", "March'15", "March'15", "April'15", "Apr…  
## $ Day <dbl> 29, 30, 31, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1…  
## $ `annee fiscale` <chr> "2014-15", "2014-15", "2014-15", "2014-15", "2014-15…  
## $ Sales <dbl> 1792.00, NA, 1526.30, 2250.26, 2077.57, 2357.48, 150…  
## $ Retail <dbl> 373.00, NA, 380.95, 363.72, 268.80, 394.23, 476.17, …  
## $ TakeOutSales <dbl> 0.00, 0.00, 99.25, 402.75, 0.00, 32.50, 93.73, 0.00,…  
## $ Bar\_Sales <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 227.…  
## $ Sales\_Restaurant <dbl> 1419.00, 0.00, 1046.10, 1483.79, 1808.77, 1930.75, 9…

The manager is using the data to have a day to day idea of the health of his buisness and therefore he added an accounting variable, annee fiscale (fiscal year) ,that we don’t need and that we can remove.The Month variable is in a format not pratical for analysis (Month, YY) we want to have just the Full moth written without the year. So for simplicity and because we have the full date in the Date column, we are going to remove the month column as it is and create a new one based on the date column, we will do the same for the weekdays column. Furthermore we don’t need the day number in our analysis so we will remove it too:

#Remove unncessary column for the analysis  
DataForAnalysis<-DailySales%>%select(-Weekday,-Month,-Day,-`annee fiscale`)  
#Add a column for weekday and month  
DataForAnalysis<-DataForAnalysis%>%mutate(Weekday=weekdays(Date),Month=months(Date))

If we look again at the data we see that we have a Sales column, representing the Total Sales for the day and then each column after it, is the total sales for each day for each component of the restaurant possible sales revenu, so Retail, Take-out, Bar and Restaurant. We can see that there is NAs in all of those data

#looking for NAs in the sales data  
sum(is.na(DataForAnalysis$Sales))

## [1] 175

sum(is.na(DataForAnalysis$Bar\_Sales))

## [1] 208

sum(is.na(DataForAnalysis$Retail))

## [1] 188

sum(is.na(DataForAnalysis$TakeOutSales))

## [1] 1

sum(is.na(DataForAnalysis$Sales\_Restaurant))

## [1] 1

we will change those NAs to 0, considering a $0 CAD sales for that day and variable

#Chaning NAs to 0 in the sales data  
DataForAnalysis[is.na(DataForAnalysis$Sales),]$Sales<-0  
DataForAnalysis[is.na(DataForAnalysis$Bar\_Sales),]$Bar\_Sales<-0  
DataForAnalysis[is.na(DataForAnalysis$Retail),]$Retail<-0  
DataForAnalysis[is.na(DataForAnalysis$TakeOutSales),]$TakeOutSales<-0  
DataForAnalysis[is.na(DataForAnalysis$Sales\_Restaurant),]$Sales\_Restaurant<-0

Once we Cleaned the data. We are interested to add some classification for the days, specialy considering that holidays and special event should have an impact on the sales of a restaurant, to test this hypothesis we create a data frame event for all the bank holidays and events in the province of Quebec during the year:

here is an example of the data used to create that data frame: <https://www.statutoryholidays.com/2017.php>, all dates with observance National, QC and event such as Mother’s day and Valentine’s day. Once this vector is created we can create a new variable called EventDay which is true if the date equals one of the date in the vector

#Add a column for EventDay  
DataForAnalysis<-DataForAnalysis%>%mutate(EventDay=ifelse(Date %in% Event,TRUE,FALSE))

# Data exploration

#look at the structure of the data  
glimpse(DataForAnalysis)

## Observations: 1,715  
## Variables: 10  
## $ Date <date> 2015-03-29, 2015-03-30, 2015-03-31, 2015-04-01, 201…  
## $ Year <dbl> 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015…  
## $ Sales <dbl> 1792.00, 0.00, 1526.30, 2250.26, 2077.57, 2357.48, 1…  
## $ Retail <dbl> 373.00, 0.00, 380.95, 363.72, 268.80, 394.23, 476.17…  
## $ TakeOutSales <dbl> 0.00, 0.00, 99.25, 402.75, 0.00, 32.50, 93.73, 0.00,…  
## $ Bar\_Sales <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00…  
## $ Sales\_Restaurant <dbl> 1419.00, 0.00, 1046.10, 1483.79, 1808.77, 1930.75, 9…  
## $ Weekday <chr> "Sunday", "Monday", "Tuesday", "Wednesday", "Thursda…  
## $ Month <chr> "March", "March", "March", "April", "April", "April"…  
## $ EventDay <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL…

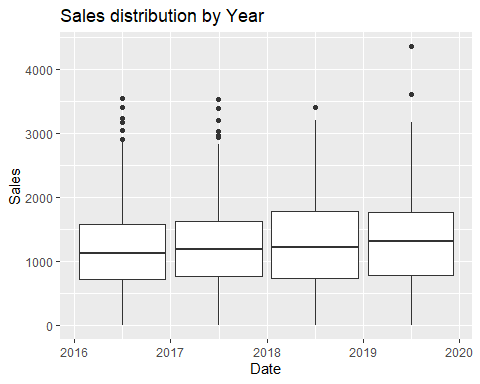
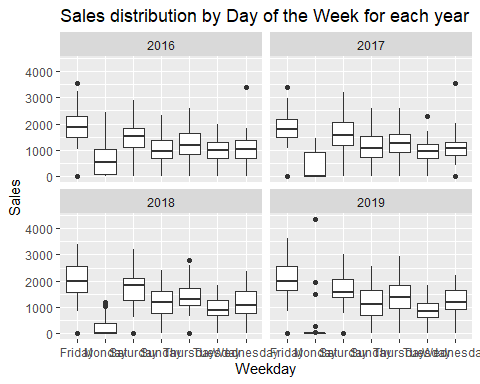
We want to have full years to have the same thing amount of days in the year to predict 2019 weeks. So we filter for 2016 and more

#Summary statistics for the variable  
summary(DataForAnalysis)

## Date Year Sales Retail   
## Min. :2016-01-01 Min. :2016 Min. : 0.0 Min. : 0.0   
## 1st Qu.:2016-12-25 1st Qu.:2016 1st Qu.: 756.8 1st Qu.: 160.2   
## Median :2017-12-19 Median :2017 Median :1212.2 Median : 257.7   
## Mean :2017-12-19 Mean :2017 Mean :1229.4 Mean : 266.5   
## 3rd Qu.:2018-12-13 3rd Qu.:2018 3rd Qu.:1703.3 3rd Qu.: 367.9   
## Max. :2019-12-29 Max. :2019 Max. :4360.8 Max. :1436.9   
## TakeOutSales Bar\_Sales Sales\_Restaurant Weekday   
## Min. : 0.00 Min. : 0.0 Min. :-2299.8 Length:1437   
## 1st Qu.: 0.00 1st Qu.: 70.0 1st Qu.: 358.0 Class :character   
## Median : 64.00 Median : 153.8 Median : 665.0 Mode :character   
## Mean : 83.85 Mean : 177.0 Mean : 702.0   
## 3rd Qu.: 120.00 3rd Qu.: 257.5 3rd Qu.: 985.2   
## Max. :2965.72 Max. :1045.0 Max. : 4284.2   
## Month EventDay   
## Length:1437 Mode :logical   
## Class :character FALSE:1437   
## Mode :character   
##   
##   
##

In our data set we have 10 variables for 1715 observations corresponding to the number of days the restaurant was open since the 29-03-2015.

## Sales Variable

 We see In the box plot above that 2019 was similar to 2018 in term of sales but with 2 outliers 

In the box plot above we see that in 2019 they were very few Sales on Mondays and that the biggest are through the years were Thursday, Friday and Saturday. ##Preparing Data for Modeling I will be considering this data as a time series and such I will use a library made to handle them

## Loading required package: timetk

## Warning: package 'timetk' was built under R version 3.6.2

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

## Loading required package: forecast

## Warning: package 'forecast' was built under R version 3.6.2

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':  
## method from   
## fitted.fracdiff fracdiff  
## residuals.fracdiff fracdiff

## Loading required package: sweep

## Warning: package 'sweep' was built under R version 3.6.2

We then transform the data to train and test data and transform the train data into a time serie

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.6.2

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Warning in set.seed(412, sample.kind = "Rounding"): non-uniform 'Rounding'  
## sampler used

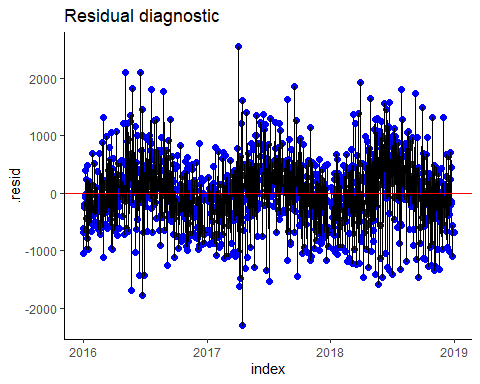
# Modeling Data

We are going to test to 2 models to predict sales for 2019. We are going to compare Linear Regression and ARIMA (Auto Regressive Integrated Moving Average) model (you can have a brief summary of ARIMA model here:<https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average>) ## ARIMA Model We need the forecast package to be able to use ARIMA model and :

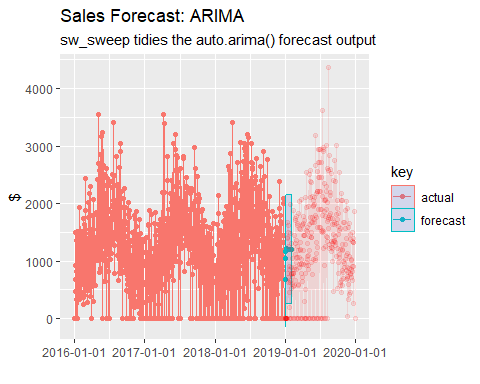
## Warning in set.seed(412, sample.kind = "Rounding"): non-uniform 'Rounding'  
## sampler used

## Series: ts\_train   
## ARIMA(2,0,0) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 mean  
## 0.5438 -0.1033 1208.0598  
## s.e. 0.0300 0.0301 34.3104  
##   
## sigma^2 estimated as 405360: log likelihood=-8629.87  
## AIC=17267.74 AICc=17267.78 BIC=17287.74  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set 0.4771356 635.8073 494.3612 -Inf Inf 0.8409041 -0.004392576

## # A tibble: 1,096 x 4  
## index .actual .fitted .resid  
## <date> <dbl> <dbl> <dbl>  
## 1 2016-01-01 0 1045. -1045.   
## 2 2016-01-02 0 609. -609.   
## 3 2016-01-03 0 676. -676.   
## 4 2016-01-04 0 676. -676.   
## 5 2016-01-05 478. 676. -198.   
## 6 2016-01-06 683. 936. -252.   
## 7 2016-01-07 920. 998. -78.0  
## 8 2016-01-08 1496. 1106. 391.   
## 9 2016-01-09 1365. 1395. -29.9  
## 10 2016-01-10 822. 1263. -442.   
## # … with 1,086 more rows

We see that as we go forward in time the model tries to correct the residual but fails miserably. get corrected by the mean of the previous days we can visualize the evolution of the residuals:  We then forecast for 2019

## # A tibble: 1,126 x 7  
## index key value lo.80 lo.95 hi.80 hi.95  
## <date> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2016-01-01 actual 0 NA NA NA NA  
## 2 2016-01-02 actual 0 NA NA NA NA  
## 3 2016-01-03 actual 0 NA NA NA NA  
## 4 2016-01-04 actual 0 NA NA NA NA  
## 5 2016-01-05 actual 478. NA NA NA NA  
## 6 2016-01-06 actual 683. NA NA NA NA  
## 7 2016-01-07 actual 920. NA NA NA NA  
## 8 2016-01-08 actual 1496. NA NA NA NA  
## 9 2016-01-09 actual 1365. NA NA NA NA  
## 10 2016-01-10 actual 822. NA NA NA NA  
## # … with 1,116 more rows



## # A tibble: 30 x 5  
## date actual pred error error\_pct  
## <date> <dbl> <dbl> <dbl> <dbl>  
## 1 2019-01-01 0 676. -676. -Inf   
## 2 2019-01-02 0 1043. -1043. -Inf   
## 3 2019-01-03 0 1173. -1173. -Inf   
## 4 2019-01-04 0 1206. -1206. -Inf   
## 5 2019-01-05 0 1211. -1211. -Inf   
## 6 2019-01-06 0 1210. -1210. -Inf   
## 7 2019-01-07 0 1209. -1209. -Inf   
## 8 2019-01-08 0 1208. -1208. -Inf   
## 9 2019-01-09 0 1208. -1208. -Inf   
## 10 2019-01-10 786. 1208. -422. -0.536  
## # … with 20 more rows

## Linear Regression

We take this time the all database and transform into a time serie before splitting into train and test set

## # A tibble: 1,437 x 38  
## Date Year Sales Retail TakeOutSales Bar\_Sales Sales\_Restaurant Weekday  
## <date> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>   
## 1 2016-01-01 2016 0 0 0 0 0 Friday   
## 2 2016-01-02 2016 0 0 0 0 0 Saturd…  
## 3 2016-01-03 2016 0 0 0 0 0 Sunday   
## 4 2016-01-04 2016 0 0 0 0 0 Monday   
## 5 2016-01-05 2016 478. 261. 29.5 25.8 162. Tuesday  
## 6 2016-01-06 2016 683. 189. 88 111 295. Wednes…  
## 7 2016-01-07 2016 920. 102. 87.8 116 614. Thursd…  
## 8 2016-01-08 2016 1496. 288. 102 262. 844. Friday   
## 9 2016-01-09 2016 1365. 347. 114. 164 740. Saturd…  
## 10 2016-01-10 2016 822. 96.6 26 121 578 Sunday   
## # … with 1,427 more rows, and 30 more variables: Month <chr>, EventDay <lgl>,  
## # index.num <int>, diff <int>, year <int>, year.iso <int>, half <int>,  
## # quarter <int>, month <int>, month.xts <int>, month.lbl <ord>, day <int>,  
## # hour <int>, minute <int>, second <int>, hour12 <int>, am.pm <int>,  
## # wday <int>, wday.xts <int>, wday.lbl <ord>, mday <int>, qday <int>,  
## # yday <int>, mweek <int>, week <int>, week.iso <int>, week2 <int>,  
## # week3 <int>, week4 <int>, mday7 <int>

We can know fit our training data

set.seed(42,sample.kind = "Rounding")

## Warning in set.seed(42, sample.kind = "Rounding"): non-uniform 'Rounding'  
## sampler used

fit\_Lr<-lm(Sales~.,data=select(trainLr\_set,-Date))  
summary(fit\_Lr)

##   
## Call:  
## lm(formula = Sales ~ ., data = select(trainLr\_set, -Date))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2364.85 -242.80 9.68 253.26 2497.16   
##   
## Coefficients: (31 not defined because of singularities)  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.233e+07 7.660e+07 -0.291 0.770728   
## Year 4.853e+04 3.350e+04 1.448 0.147795   
## WeekdayMonday -1.036e+03 2.795e+02 -3.707 0.000221 \*\*\*  
## WeekdaySaturday -3.548e+02 8.537e+01 -4.156 3.50e-05 \*\*\*  
## WeekdaySunday -9.528e+02 1.464e+02 -6.510 1.15e-10 \*\*\*  
## WeekdayThursday -4.997e+02 8.718e+01 -5.731 1.30e-08 \*\*\*  
## WeekdayTuesday -6.070e+02 2.114e+02 -2.872 0.004161 \*\*   
## WeekdayWednesday -5.791e+02 1.464e+02 -3.954 8.18e-05 \*\*\*  
## MonthAugust 1.459e+04 1.024e+04 1.424 0.154670   
## MonthDecember 2.774e+04 2.046e+04 1.356 0.175544   
## MonthFebruary -7.126e+03 6.318e+03 -1.128 0.259620   
## MonthJanuary -1.084e+04 7.496e+03 -1.446 0.148475   
## MonthJuly 1.104e+04 7.553e+03 1.461 0.144237   
## MonthJune 7.605e+03 7.786e+03 0.977 0.328863   
## MonthMarch -3.668e+03 7.238e+03 -0.507 0.612461   
## MonthMay 3.838e+03 3.829e+03 1.002 0.316472   
## MonthNovember 2.448e+04 1.764e+04 1.388 0.165424   
## MonthOctober 2.100e+04 1.519e+04 1.383 0.167083   
## MonthSeptember 1.787e+04 1.357e+04 1.317 0.188141   
## index.num -3.550e-04 1.234e-03 -0.288 0.773618   
## diff NA NA NA NA   
## year NA NA NA NA   
## year.iso -3.719e+04 2.497e+04 -1.490 0.136643   
## half NA NA NA NA   
## quarter NA NA NA NA   
## month NA NA NA NA   
## month.xts NA NA NA NA   
## month.lbl.L NA NA NA NA   
## month.lbl.Q NA NA NA NA   
## month.lbl.C NA NA NA NA   
## month.lbl^4 NA NA NA NA   
## month.lbl^5 NA NA NA NA   
## month.lbl^6 NA NA NA NA   
## month.lbl^7 NA NA NA NA   
## month.lbl^8 NA NA NA NA   
## month.lbl^9 NA NA NA NA   
## month.lbl^10 NA NA NA NA   
## month.lbl^11 NA NA NA NA   
## day 1.193e+02 1.279e+02 0.933 0.350980   
## hour NA NA NA NA   
## minute NA NA NA NA   
## second NA NA NA NA   
## hour12 NA NA NA NA   
## am.pm NA NA NA NA   
## wday NA NA NA NA   
## wday.xts NA NA NA NA   
## wday.lbl.L NA NA NA NA   
## wday.lbl.Q NA NA NA NA   
## wday.lbl.C NA NA NA NA   
## wday.lbl^4 NA NA NA NA   
## wday.lbl^5 NA NA NA NA   
## wday.lbl^6 NA NA NA NA   
## mday NA NA NA NA   
## qday -9.500e-01 1.085e+02 -0.009 0.993013   
## yday 3.196e+01 6.285e+01 0.508 0.611245   
## mweek -3.149e+00 1.895e+01 -0.166 0.868059   
## week -9.656e+01 5.611e+01 -1.721 0.085575 .   
## week.iso -7.223e+02 4.756e+02 -1.519 0.129119   
## week2 1.089e+02 3.168e+01 3.437 0.000611 \*\*\*  
## week3 1.990e+01 1.776e+01 1.121 0.262743   
## week4 -3.182e+01 1.418e+01 -2.245 0.025004 \*   
## mday7 -1.338e+01 5.157e+01 -0.259 0.795304   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 467 on 1064 degrees of freedom  
## (1 observation deleted due to missingness)  
## Multiple R-squared: 0.6066, Adjusted R-squared: 0.5955   
## F-statistic: 54.68 on 30 and 1064 DF, p-value: < 2.2e-16

Now we can predict

predict\_lr<-predict(fit\_Lr,newdata=select(testLr\_set,-Date))

## Warning in predict.lm(fit\_Lr, newdata = select(testLr\_set, -Date)): prediction  
## from a rank-deficient fit may be misleading

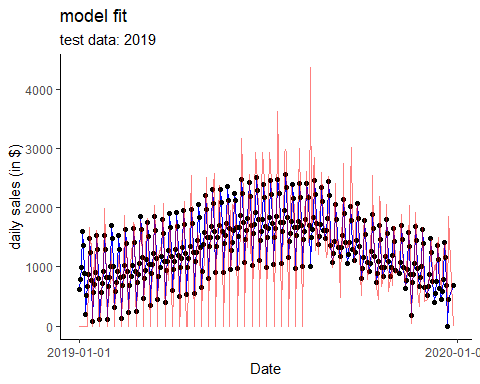
error\_lr<-testLr\_set%>%select(Date,actual=Sales)%>%  
 mutate(pred=predict\_lr,  
 error=actual-predict\_lr,  
 error\_pct=error/actual)  
error\_lr

## # A tibble: 341 x 5  
## Date actual pred error error\_pct  
## <date> <dbl> <dbl> <dbl> <dbl>  
## 1 2019-01-01 0 622. -622. -Inf   
## 2 2019-01-02 0 785. -785. -Inf   
## 3 2019-01-03 0 984. -984. -Inf   
## 4 2019-01-04 0 1604. -1604. -Inf   
## 5 2019-01-05 0 1368. -1368. -Inf   
## 6 2019-01-06 0 890. -890. -Inf   
## 7 2019-01-07 0 188. -188. -Inf   
## 8 2019-01-08 0 519. -519. -Inf   
## 9 2019-01-09 0 666. -666. -Inf   
## 10 2019-01-10 786. 866. -79.1 -0.101  
## # … with 331 more rows

dataToPlot<-testLr\_set%>%add\_column(pred=predict(fit\_Lr,testLr\_set)%>%tibble::enframe(name = NULL) %>% pull(value))

## Warning in predict.lm(fit\_Lr, testLr\_set): prediction from a rank-deficient fit  
## may be misleading

dataToPlot %>%  
 ggplot(aes(x = Date, y = pred)) +  
 geom\_line(color='blue') +  
geom\_point()+  
geom\_line(data = testLr\_set,aes(x=Date,y=Sales),color="red",alpha=0.5)+  
 scale\_x\_date(date\_breaks = "1 year", date\_labels = "%F") +  
 scale\_color\_manual(  
 values = c(  
 "weekly\_sales" = "black",  
 "lm\_pred" = "#fdc7d7"  
 )  
 ) +  
 theme\_classic() +  
 labs(title = "model fit",  
 subtitle = "test data: 2019",  
 x = "Date",  
 y = "daily sales (in $)"  
 )

 On the plot abobe we see in red The actual value of sales and in black the prediction made by the linear regression model.

# Discussion

In conclusion, we see that the linear regression has a better accuracy than the ARIMA model for predicting the sales for 2019. Both these models could be improved by splitting the data on weekdays and looking at the sales by weekday. We saw that the restaurant was not open on (made no sales) Monday for 2019. When comparing the sales by year we see that 2019 was similar in term of sales to the other year with only 341 days opened instead of 365 days. It might be interesting to predict the sales for 2019 considering the full 365 days.