# Business Price Prediction

pranat

2024-03-31

## Stating objectives

In this project report our goal will be to answer and answer the following objectives:

1. Predict the future sales across all stores and across all departements from the given data sets.

2.Find whether there is a significant correlation between Markdowns and holiday weeks.

3. Provide any other meaningful insights for the business from the given data and summarize the results of above stated objectives with a possible solution if required.

## Data Gathering and Cleaning

The following data sets were taken from Kaggle, . We have two data sets with names features data set and sales data set. We can retreive information about sales on different dates across all stores and departements from Sales data set and information about temprature, Consumer price index, etc from features data set.

Focusing ourselves on cleaning the features data set by removing all the NA values from Markdown 1 through 5 columns and from all other integer and decimal values columns since we cannot reach out and get the missing data, it is the only meaningful way to work with a clean data set. Furthermore, both our datasets are sufficiently large so we don't have to worry about the size shrinkage of data sets having a negative impact on tests/analyses that we will run. Next, since the IsHoliday column is a boolean valued column representing whether the given week contained a Holiday or not. So, we convert the FALES values to 0 and TRUE values to 1. Following is the SQL query required to do these simple data cleaning processes.

## Analyses

First we start by running a point biserial correlation test between Markdown column and IsHoliday to check for correlation between them. Point biserial correlation is special kind of pearson correlation test between continuous and dichotomus variables. Following code was written on python:

```
import pandas as pd
import numpy as np
from scipy import stats

df = pd.read_csv(r"C:\Users\Pranat\Documents\Analysis\Features-data-set_1_-_2_.csv")
MarkDown_1 = df['MarkDown1']
print(MarkDown_1)
```

```
## 0        410.31
## 1       5629.51
## 2       4640.65
## 3       5011.32
## 4       2725.36
##           ...
## 2513    4842.29
## 2514    9090.48
## 2515    3789.94
## 2516    2961.49
## 2517     212.02
## Name: MarkDown1, Length: 2518, dtype: float64
```

```
MarkDown_2 = df['MarkDown2']
MarkDown_3 = df['MarkDown3']
MarkDown_4 = df['MarkDown4']
MarkDown_5 = df['MarkDown5']
IsHoliday = df['IsHoliday']

#list_1 = df['Markdown1'].tolist()
x = np.array(df['MarkDown1'])
y = np.array(df['MarkDown2'])
z = np.array(df['MarkDown3'])
n = np.array(df['MarkDown4'])
m = np.array(df['MarkDown5'])
a = np.array(df['IsHoliday'])
A = [x,y,z,n,m]
B=[]
for i in range(len(A)):
    b = stats.pointbiserialr(A[i],a)
    #print(b)
    B.append(b)
print(B)
```

```
## [SignificanceResult(statistic=0.17571231189716227, pvalue=6.567084851843017e-19), SignificanceResult
```

```
#a = stats.pointbiserialr(x,a)
#print(a)
```

Our next analysis would be to predict the sales for next year across all departments of all stores. Since there are data of sales from all departments and stores we get many repeated dates. We want to create a time series from this data of distinct dates and total sales on that date. So, we make a dictionary having keys as distinct dates and values as total sales on that given date. Then we standardize our sales data using z-score and rename the column date to 'ds' and sales value as 'y'. Since, it is requirement for the prophet package we're using today for time series analysis, it was developed by Meta. We predict sales for the next year with a confidence level 95%.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
```

```
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(skimr)
library(lubridate)
library(readr)
library(prophet)
```

```
## Loading required package: Rcpp
## Loading required package: rlang
##
## Attaching package: 'rlang'
##
## The following objects are masked from 'package:purrr':
##
##     %@%, flatten, flatten_chr, flatten_dbl, flatten_int, flatten_lgl,
##     flatten_raw, invoke, splice
```

```
df <- read_csv("Analysis/cleaned_new-_1_.csv")
```

```
## New names:
## Rows: 143 Columns: 3
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (2): ...1, y date (1): ds
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
z_score <- c((df$y-mean(df$y))/sd(df$y))
y<-z_score
dF = data.frame(df$ds,y)
DF <- dF %>% rename(ds=df.ds)
df_<- DF %>% filter(y< 3)
m<-prophet(df_)
```

```
## Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.
```

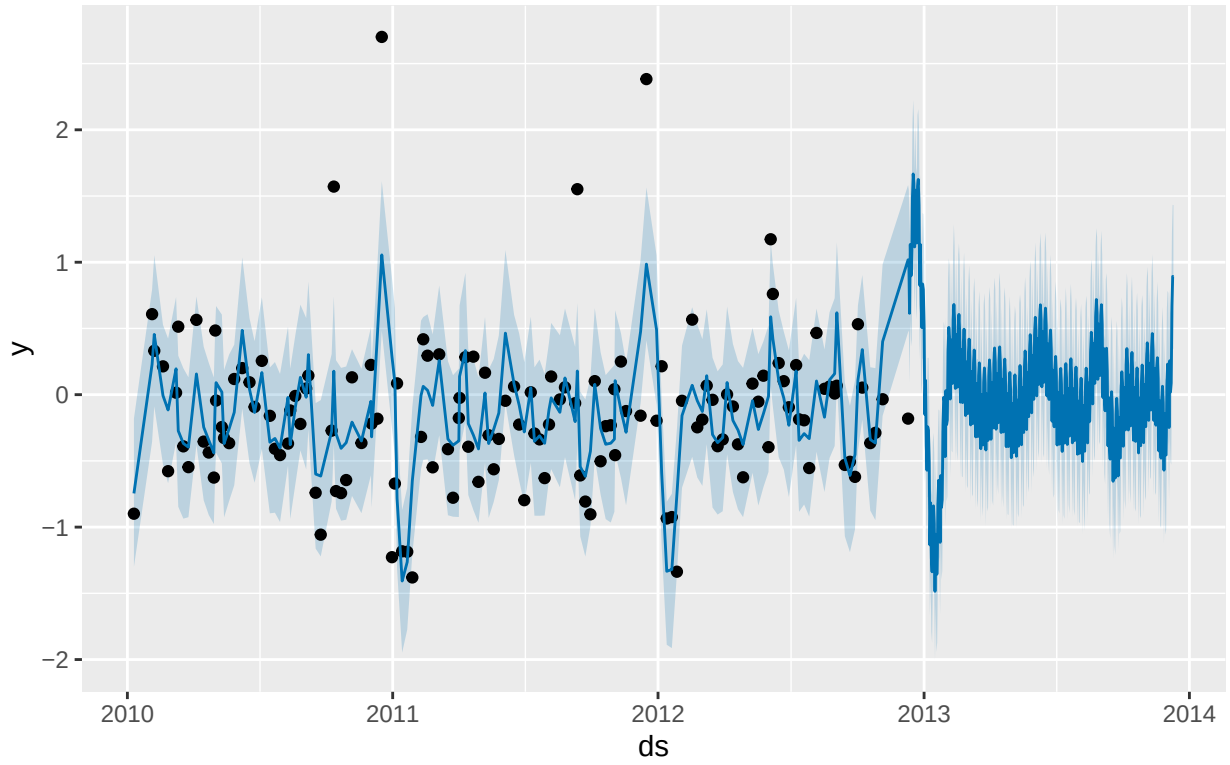
```
future<-make_future_dataframe(m,periods=365)
head(future)
```

```
##           ds
## 1 2010-01-10
## 2 2010-02-04
## 3 2010-02-07
## 4 2010-02-19
## 5 2010-02-26
## 6 2010-03-09
```

```r
forecast <- predict(m,future)
#bias(df,forecast)
head(forecast)
```

```
##           ds       trend additive_terms additive_terms_lower
## 1 2010-01-10 -0.02396061    -0.72160604          -0.72160604
## 2 2010-02-04 -0.02272690     0.25158465           0.25158465
## 3 2010-02-07 -0.02257885     0.47753821           0.47753821
## 4 2010-02-19 -0.02198667     0.01474488           0.01474488
## 5 2010-02-26 -0.02164123    -0.09374617          -0.09374617
## 6 2010-03-09 -0.02109839     0.21648960           0.21648960
##   additive_terms_upper     weekly weekly_lower weekly_upper      yearly
## 1          -0.72160604  0.2039249    0.2039249    0.2039249 -0.92553089
## 2           0.25158465  0.1015302    0.1015302    0.1015302  0.15005446
## 3           0.47753821  0.2039249    0.2039249    0.2039249  0.27361336
## 4           0.01474488 -0.2324456   -0.2324456   -0.2324456  0.24719045
## 5          -0.09374617 -0.2324456   -0.2324456   -0.2324456  0.13869940
## 6           0.21648960  0.1960712    0.1960712    0.1960712  0.02041842
##   yearly_lower yearly_upper multiplicative_terms multiplicative_terms_lower
## 1  -0.92553089  -0.92553089                    0                           0
## 2   0.15005446   0.15005446                    0                           0
## 3   0.27361336   0.27361336                    0                           0
## 4   0.24719045   0.24719045                    0                           0
## 5   0.13869940   0.13869940                    0                           0
## 6   0.02041842   0.02041842                    0                           0
##   multiplicative_terms_upper  yhat_lower yhat_upper trend_lower trend_upper
## 1                          0 -1.30016401 -0.1812161 -0.02396061 -0.02396061
## 2                          0 -0.35623834  0.7933462 -0.02272690 -0.02272690
## 3                          0 -0.08815801  1.0536483 -0.02257885 -0.02257885
## 4                          0 -0.52699447  0.5269605 -0.02198667 -0.02198667
## 5                          0 -0.66796847  0.4238449 -0.02164123 -0.02164123
## 6                          0 -0.35935639  0.7373684 -0.02109839 -0.02109839
##           yhat
## 1 -0.745566647
## 2  0.228857755
## 3  0.454959361
## 4 -0.007241788
## 5 -0.115387401
## 6  0.195391209
```

```r
plot(m,forecast)
```




For our third and last objective we will see whether there is a correlation between fuel price, CPI and temperature.

```
## STANDARDIZE ALL DATA
library(tidyverse)
library(skimr)
library(lubridate)
library(readr)
library("corrplot")
```

```
## corrplot 0.92 loaded
```

```
library("psych")
```

```
##
## Attaching package: 'psych'
```

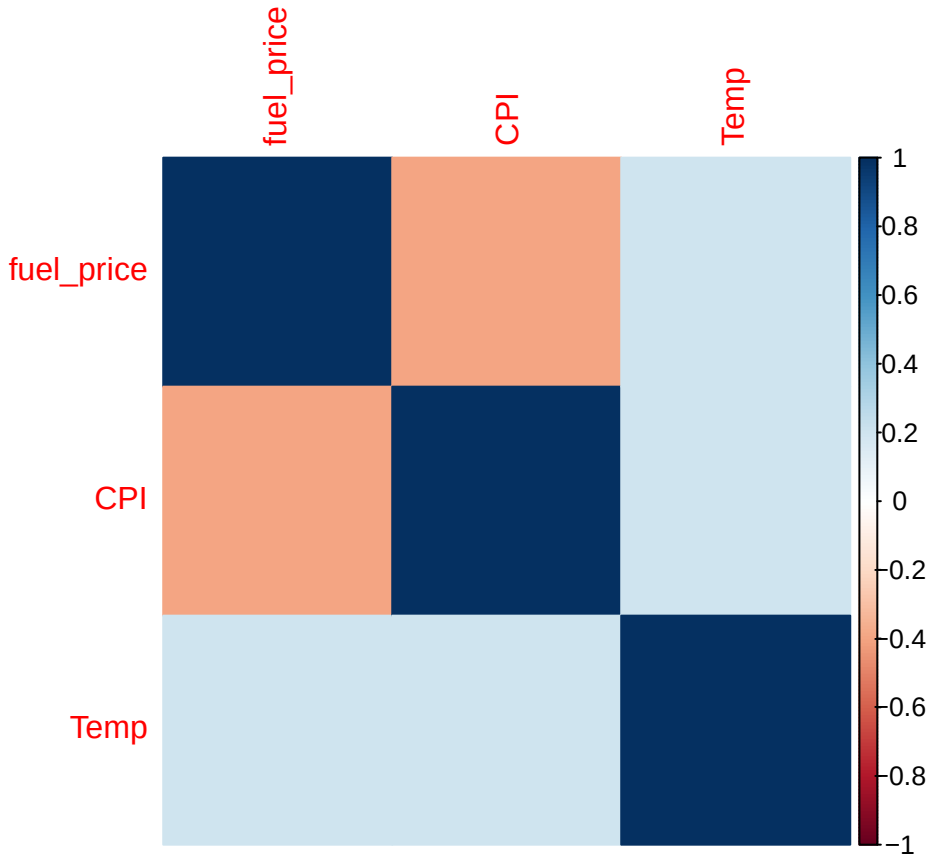
```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
features <- read_csv("Analysis/Features-data-set_1_-_2_.csv")
```

```
## New names:
## * `` -> `...13`
## * `` -> `...14`
```

```
## Rows: 2518 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Date
## dbl (11): Store, Temperature, Fuel_Price, MarkDown1, MarkDown2, MarkDown3, M...
## lgl  (2): ...13, ...14
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
features1 <- features %>% drop_na(CPI)
Temp <- c((features1$Temperature-mean(features1$Temperature))/sd(features1$Temperature))
CPI <- c((features1$CPI-mean(features1$CPI))/sd(features1$CPI))
fuel_price <- c((features1$Fuel_Price-mean(features1$Fuel_Price))/sd(features1$Fuel_Price))
z <- data.frame(fuel_price,CPI,Temp)
cor_matrix <- cor(z)
corrplot(cor_matrix, method = "color")
```




```
cor.test(z$Temp,z$CPI)
```

```
##
##  Pearson's product-moment correlation
##
## data:  z$Temp and z$CPI
## t = 9.4339, df = 2065, p-value < 2.2e-16
```

```
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1615696 0.2442422
## sample estimates:
##       cor
## 0.2032682
```

```
cor.test(z$fuel_price,z$CPI)
```

```
##
##  Pearson's product-moment correlation
##
## data:  z$fuel_price and z$CPI
## t = -19.468, df = 2065, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4296227 -0.3567450
## sample estimates:
##       cor
## -0.3938025
```

```
##print corresponding p-values as well..----done
## print graphs of Temp Vs CPI and Fuel Vs CPI
```

## Conclusion

From the graph predicting the following year's sales shows a similar pattern as before and sales should follow the rocky path as before. However it shows a net growth in sales towards the end. This resut was derived with 95% confidence level.

We saw slight correlation between MarkDown values and whether the week was a holidy week or not.

There is also slight but not significant negative correlation between CPI, temperature and Fuel Price. Thus showing us that mainting higher tempratures might have slight increasing affect on CPI.