Assignment 2

Alex Khaykin

2024-11-3

library(tidyverse)  
library(lubridate)  
library(ggplot2)  
library(corrplot)  
library(rpart)  
library(rpart.plot)  
library(caret)  
library(ranger)

# INTRO

The landscape of modern business increasingly relies on data-driven decision-making, and effective analysis of sales data can significantly enhance operational efficiency and customer satisfaction. For my project, I focus on understanding the relationship between two critical factors influencing Fulfilment\_Speed: Order.Priority and and other influencing factors such as geography and the companies Total.Revenue. By employing a linear regression model, I aim to uncover patterns within my chosen dataset composed of 1,000 sales transactions, examining how these variables interact and impact overall sales outcomes.

Order priority typically reflects the urgency and importance assigned to each sale, potentially impacting the fulfillment process and customer experience. Fulfillment speed, on the other hand, denotes the time taken to process and deliver an order. In a competitive market, optimizing both order priority and fulfillment speed is essential for meeting customer expectations and driving sales growth. Through linear regression analysis, we will explore whether order priority status can predict fulfillment speed, providing valuable insights for stakeholders looking to enhance their operational strategies.

This project will articulate the statistical relationship between order priority and fulfillment speed and provide actionable recommendations for improving order management and fulfillment processes. By leveraging this analysis, businesses can make informed decisions to enhance their performance in a fast-paced market environment.

# DATA

Reading in the data downloaded from the site, and converting it to a data frame from where I will be unitizing R to run summary statistics to learn about the data and the information contained within.

sales\_10000 <- read.csv("C:\\Users\\akhay\\OneDrive\\Documents\\DATA\_SCIENCE\\DATA\_622\\Assignments\\Assignment\_2\\10000 Sales Records.csv")

### Summary

This section gives the names of the columns(variable), and the basic statistics of the dataset. There seems to be no missing data in this dataset.

summary(sales\_10000)

## Region Country Item.Type Sales.Channel   
## Length:10000 Length:10000 Length:10000 Length:10000   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
## Order.Priority Order.Date Order.ID Ship.Date   
## Length:10000 Length:10000 Min. :100089156 Length:10000   
## Class :character Class :character 1st Qu.:321806669 Class :character   
## Mode :character Mode :character Median :548566305 Mode :character   
## Mean :549871874   
## 3rd Qu.:775998104   
## Max. :999934232   
## Units.Sold Unit.Price Unit.Cost Total.Revenue   
## Min. : 2 Min. : 9.33 Min. : 6.92 Min. : 168   
## 1st Qu.: 2531 1st Qu.:109.28 1st Qu.: 56.67 1st Qu.: 288551   
## Median : 4962 Median :205.70 Median :117.11 Median : 800051   
## Mean : 5003 Mean :268.14 Mean :188.81 Mean :1333355   
## 3rd Qu.: 7472 3rd Qu.:437.20 3rd Qu.:364.69 3rd Qu.:1819143   
## Max. :10000 Max. :668.27 Max. :524.96 Max. :6680027   
## Total.Cost Total.Profit   
## Min. : 125 Min. : 43.4   
## 1st Qu.: 164786 1st Qu.: 98329.1   
## Median : 481606 Median : 289099.0   
## Mean : 938266 Mean : 395089.3   
## 3rd Qu.:1183822 3rd Qu.: 566422.7   
## Max. :5241726 Max. :1738178.4

The dataset contains 10,000 sales records, with 14 columns. However for the purpose of the question I am asking, I will be treating these as company records and dropping the following columns: Country, Order Id. Additionally I will be making a new primary predictor variable Fullfilment Speed, which is the lag between Order Date and Ship Date, after which I will drop the two date columns entirely. I am treating this as the average fulfillment speed for each of the hypothetical companies and their records contained in the dataset.

sales\_10000 <- sales\_10000 %>%   
 select(-Country, -Order.ID) %>%   
 mutate(Order.Date=as.Date(Order.Date, format="%m/%d/%Y"),   
 Ship.Date=as.Date(Ship.Date, format= "%m/%d/%Y"),  
 Fulfilment\_Speed =as.numeric(Ship.Date - Order.Date)) %>%   
 select(-Order.Date, -Ship.Date) %>%   
 mutate\_if(is.character, as.factor)

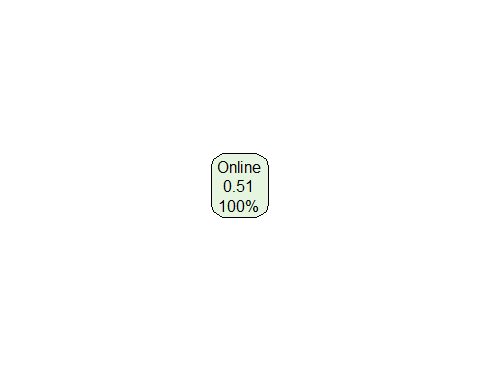
# MODELS

## Decision Tree 1

### Sales Channel

To predict whether an order would come in through Online or Offline sales channel, I built a decision tree model that uses the following predictors. Region, Item Type, Order Priority. Units Sold, Unit Price, Unit Cost Total Revenue, Total Cost, Total Profit, and a new variable I created called Fulfillment Speed. Which is the lag between when the order was placed and when the order was shipped. I created a training a training and test set from the original dataset using a 75 / 25 split, where 75% of the data went into the training dataset and 25% went into the test dataset. I trained a classification tree and used predictions from the model to test the accuracy of the model on the testing dataset.

set.seed(25)  
ind <- createDataPartition(sales\_10000$Sales.Channel, p=0.75, list=FALSE)  
train\_1 <- sales\_10000[ind, ]  
test\_1 <- sales\_10000[-ind, ]  
cart\_1 <- rpart(Sales.Channel~.,data = train\_1, method = "class")  
rpart.plot(cart\_1)

 The above tree has only a root node, indicating that the only predictive split is between whether data are categorized as online or offline. 51% of the dataset are online orders, leaving 49% offline in the dataset. This suggests that there are no significant predictors of whether an order comes from an online or an offline sales channel.

preds\_1 <- predict(cart\_1, newdata = test\_1, type = "class")  
confusionMatrix(preds\_1, test\_1$Sales.Channel)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Offline Online  
## Offline 0 0  
## Online 1234 1265  
##   
## Accuracy : 0.5062   
## 95% CI : (0.4864, 0.526)  
## No Information Rate : 0.5062   
## P-Value [Acc > NIR] : 0.508   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.5062   
## Prevalence : 0.4938   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : Offline   
##

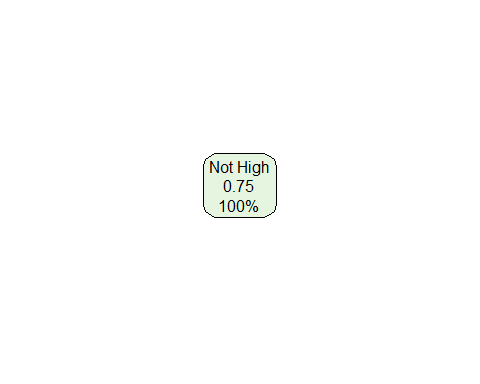
The confusion matrix from the predictions reveals very poor accuracy of only 0.51, which is the same as the proportion of online orders in the dataset. The specificity is 100%, meaning that the model can identify when an order is not an online order, but has zero percent sensitivity, meaning that it is unable to predict when it is an online order. This is likely due to the data being artificial and not real sales data.

## Decision Tree Model 2

### Order Priority

In an attempt to predict whether an order would come in as high priority, I first re-coded the original order priority variable as either high or not high (everything else). I then built another decision tree model that uses the following predictors. Region, Item Type, Sales Channel, Units Sold, Unit Price, Unit Cost Total Revenue, Total Cost, Total Profit, and Fulfillment Speed. I again used a partition of 75 / 25.

sales\_10000 <- sales\_10000 %>%   
 mutate(Order.Priority=ifelse(Order.Priority=="H","High", "Not High"))  
ind <- createDataPartition(sales\_10000$Order.Priority, p=0.75, list=FALSE)  
train\_2 <- sales\_10000[ind, ]  
test\_2 <- sales\_10000[-ind, ] %>%   
 mutate\_if(is.character, as.factor)  
cart\_2 <- rpart(Order.Priority~.,data = train\_2, method = "class")  
rpart.plot(cart\_2)



The above tree also has only a root node, indicating that the only predictive split is between whether data are categorized as high priority or not. 75% of the dataset are not high priority orders, leaving 25% high priority orders in the dataset. This suggests that there are no significant predictors of whether an order with be designated as a high priority.

preds\_2 <- predict(cart\_2, newdata = test\_2, type = "class")  
confusionMatrix(preds\_2, test\_2$Order.Priority)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction High Not High  
## High 0 0  
## Not High 625 1874  
##   
## Accuracy : 0.7499   
## 95% CI : (0.7324, 0.7668)  
## No Information Rate : 0.7499   
## P-Value [Acc > NIR] : 0.5107   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.7499   
## Prevalence : 0.2501   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : High   
##

The confusion matrix from the predictions reveals very poor accuracy of only 0.75, which is the same as the proportion of not high priority order in the dataset. The specificity is also 100%, again meaning that the model can identify when an order is not a high priority, but once again has a zero percent sensitivity, meaning that it cannot predicty high priority orders.

## Random Forest Model

Random forest model can be helpful by making decision trees more robust and thus more accurate, they are especially helpful in cases where the model is overfit. They are less sensitive to multicolinearity and over-training when compared to a decision tree. However, the downside to a random forest is that they are less intuitive and harder to interpret. They will also suffer from the same complaints about decision trees such as being subjective and complex as mentioned in the “the good the bad and the ugly of using decision trees” article.

I used a random forest model on the second training set to test whether I could predict high priority orders and improve the model. To further prevent possible overfitting, I also implemented 10-fold cross-validation from the train model. I then generated predictions and computed a confusion matrix.

train\_cont <- trainControl(method = "cv", number = 10)  
rf <- train(Order.Priority~.,data = train\_2, method = "rf", trControl = train\_cont, tuneLength = 5)  
preds\_3 <- predict(rf, test\_2)  
confusionMatrix(preds\_3, test\_2$Order.Priority)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction High Not High  
## High 0 0  
## Not High 625 1874  
##   
## Accuracy : 0.7499   
## 95% CI : (0.7324, 0.7668)  
## No Information Rate : 0.7499   
## P-Value [Acc > NIR] : 0.5107   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.7499   
## Prevalence : 0.2501   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : High   
##

Unfortunately, the random forest was unable to build a better predictive model. The confusion matrix from the predictions reveals the same accuracy, sensitivity, specificity.

# CONCLUSION

I fitted three models, two decision trees and one random forest to try to build a robust classification model from 10000 simulated sales. My two outcomes of interest sales channel and order priority were re-coded order priority into a binary variable “high” and “not high”. My first decision for sales channel performed very poorly with only 51% accuracy and no significant predictors. This was very likely because the data was simulated but had it been real data I may have attained a usable tree. As discussed in the article a usable tree would be one that is easy to understand with strategic insights that could be interpreted easily and intuitively. For example, if the tree had child nodes instead of just a root node, the splits at those nodes could be used to derive the probability that the region, item type or some characteristic predicts the sales channel of an order. This could in turn be used strategically by a company to better understand what types of orders are likely to be online or vs. offline.

The second decision tree attempted to predict order priority, but similarly performed poorly, again likely due to the artificial nature of the data. The random forest yielded the same accuracy, sensitivity, specificity, meaning that these models are unable to predict high priority order from any of the predictors chosen.