Projeto Lauro

Lauro

11 de outubro de 2017

ESPECIFICAÇÃO DO PROBLEMA

O Magazine Luiza figura atualmente entre os maiores varejistas do país e por consequência também enfrenta o desafio de fazer uma predição adequada a sua demanda. Pensando nisso, você determinará quantas unidades de cada produto devemos comprar do fornecedor, lembrando que excessos significam estoque parado e escassez significa cliente perdido.

Abaixo no tópico "Dados", você encontrará as informações de como acessar o arquivo csv com os dados históricos de venda de produtos. Os dados que seguem possuem a quantidade vendida e o valor de venda.

a) Faça uma separação em grupos de produtos, usando um algoritmo de agrupamento não supervisionado. Isso será muito importante para o próximo ítem, pois como já exposto antes, existem produtos com características particulares. Avalie a qualidade do agrupamento, assim como as características que definem cada grupo.

b) Faça a previsão de venda para cada um dos produtos para os meses de junho, julho e agosto de 2017. Imagine que você tem que fazer a compra para reposição desses três meses e que os estoques estão zerados, quantas peças de cada tipo você compraria? Também demonstre as métricas de qualidade do modelo gerado, discorrendo sobre os parâmetros escolhidos para a execução do algoritmo.

c) Faça uma análise dos resultados que encontrou, discorra sobre o problema e exponha suas percepções e descobertas. Tem algum dado que seria relevante e que não foi fornecido?

TRATAMENTO E PREPARACAO DOS DADOS

A linguagem de programação R foi escolhida por ser uma linguagem estatística robusta e significativamente utilizada para análise de dados nas diversas comunidades de cientistas de dados espalhadas pelo mundo. Poderia ter sido utilizada a linguagem Python para esse mesmo propósito. Porém, como eu estou atualmente praticando a linguagem R no curso Formação Cientista de Dados, da Data Science Academy, optei por utilizar tal linguagem. Todas as linhas de código estão devidamente comentadas visando uma explicação simples para quem é leigo na linguagem R. Os comentário em R são feitos utilizando o caractere '#'. Portanto, tudo que aparece após o símbolo '#' é tratado como comentário.

```
#inclusao dos pacotes necessarios
#install.packages('dplyr') #Pacote para a transformacao dos dados
suppressMessages(library(dplyr))
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
#troque pelo seu diretorio de trabalho
#faz a leitura do arquivo
arquivo = read.csv("C:/Users/Lauro Martins/Desktop/desafio.csv")


#obtem apenas os produtos que foram vendidos de fato
produtosVendidos = filter(arquivo, process_status == 'processado')
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.1
```

```r
#mostra as 6 primeiras linhas do conjunto de dados obtido pela linha acima
#Obs.: algumas colunas foram ocultadas
head(produtosVendidos[, c(2, 3, 4, 8)])
```

```
##                               code quantity   price
## 1 e6762ba2ffbca07ab6cee7551caeaad5        1  978.90
## 2 e6762ba2ffbca07ab6cee7551caeaad5        1 1036.29
## 3 e6762ba2ffbca07ab6cee7551caeaad5        1  978.90
## 4 e6762ba2ffbca07ab6cee7551caeaad5        1  976.05
## 5 e6762ba2ffbca07ab6cee7551caeaad5        1 1089.10
## 6 e6762ba2ffbca07ab6cee7551caeaad5        1  949.00
##                            category
## 1 4ece547755cba9e7fc14125bc895f31b
## 2 4ece547755cba9e7fc14125bc895f31b
## 3 4ece547755cba9e7fc14125bc895f31b
## 4 4ece547755cba9e7fc14125bc895f31b
## 5 4ece547755cba9e7fc14125bc895f31b
## 6 4ece547755cba9e7fc14125bc895f31b
```

```r
#Para facilitar a compreensão, o código (code) e a categoria de cada produto foram
#transformados em um número inteiro
produtosVendidos[, 'code'] = as.numeric(produtosVendidos$code)
produtosVendidos[, 'category'] = as.numeric(produtosVendidos$category)


#mostra as 6 primeiras linhas do conjunto de dados após a conversão acima
#Obs.: algumas colunas foram ocultadas
head(produtosVendidos[, c(2, 3, 4, 8)])
```

```
##   code quantity   price category
## 1  125        1  978.90        2
## 2  125        1 1036.29        2
## 3  125        1  978.90        2
## 4  125        1  976.05        2
## 5  125        1 1089.10        2
## 6  125        1  949.00        2
```

```r
#retira a coluna order_id por não ser necessária na análise
produtosVendidos['order_id'] = NULL


#obtem a quantidade de vendas de cada produto em ordem decrescente
qtdVendasCadaProduto = count(produtosVendidos, code, sort = TRUE)


#mostra os 6 primeiros produtos com suas respectivas quantidades (n)
#Por exemplo, o produto 25 foi vendido 18943 vezes
head(qtdVendasCadaProduto)
```

```
## # A tibble: 6 x 2
##    code     n
##   <dbl> <int>
## 1    25 18943
## 2    46 14899
## 3    28  7990
```

```
## 4    27  7864
## 5    63  5402
## 6    18  5370
```

```
#obtem a quantidade de produtos diferentes (131 produtos)
qtdProdutos = nrow(qtdVendasCadaProduto)

#obtem a quantidade de vendas em cada categoria em ordem decrescente
qtdVendasCadaCategoria = count(produtosVendidos, category, sort = TRUE)

#mostra as 6 primeiras categorias com suas respectivas quantidade de vendas
#Por exemplo a categoria 1 foi a mais vendida, com 133046 vendas
head(qtdVendasCadaCategoria)
```

```
## # A tibble: 6 x 2
##   category      n
##      <dbl>  <int>
## 1        1 133046
## 2        7  15449
## 3        5   4255
## 4       10    853
## 5        6    272
## 6        4    200
```

```
#obtem a quantidade de categorias diferentes (11 categorias)
qtdCategorias = nrow(qtdVendasCadaCategoria)
```

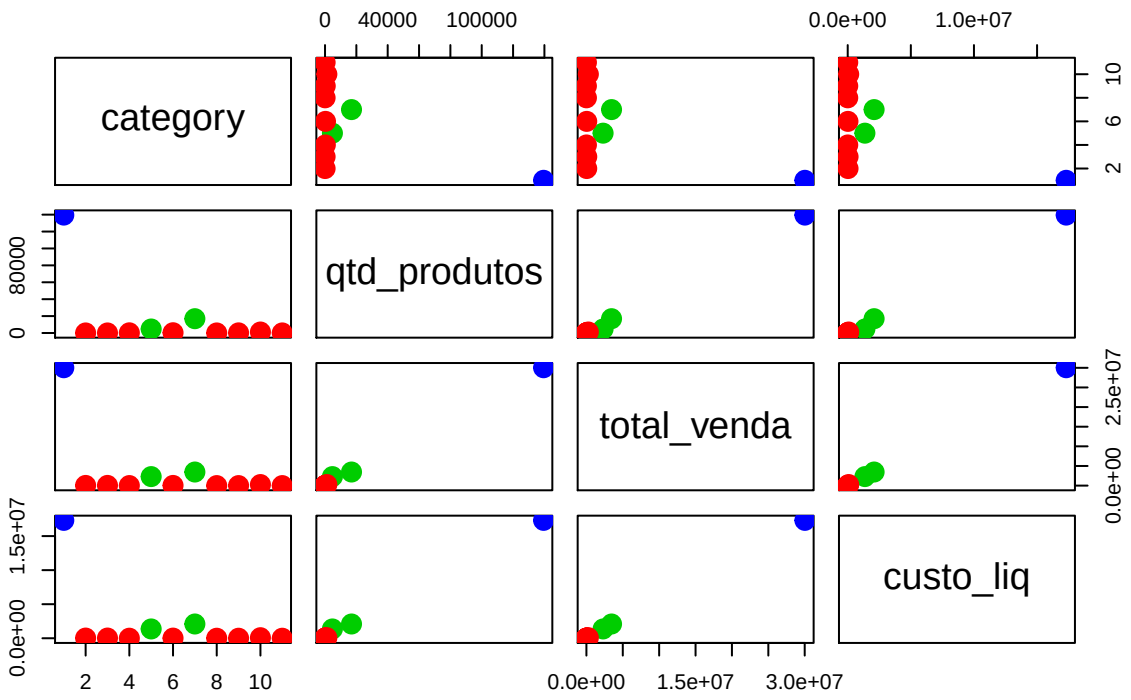
ANÁLISE DOS DADOS - agrupamento

```
#obtem apenas as colunas numericas que interessam para o agrupamento
analise1 = select(produtosVendidos, code, quantity, price, pis_cofins, icms, tax_substitution,
                  category, liquid_cost)

#ordena as linhas pelo codigo do produto
analise1 = arrange(analise1, code)


#agrupa pela categoria do produto
#obtem a quantidade de cada produto vendido
#obtem o preco total das vendas de cada produto
#obtem o  valor total do custo liquido de cada produto
#ordena em ordem decrescente pelo preco total das vendas de cada produto
x = analise1 %>%
  group_by(category) %>%
  summarise(qtd_produtos = sum(quantity),
            total_venda = sum(price),
            custo_liq = sum(liquid_cost)) %>%
  arrange(desc(total_venda))

#Segue abaixo uma tabela que mostra a relação de cada categoria com os seus respectivos valores
#Por exemplo, a categoria 1 foi a mais vendida
print(x)
```

```
## # A tibble: 11 x 4
```

```
##    category qtd_produtos total_venda    custo_liq
##       <dbl>        <int>       <dbl>        <dbl>
##  1        1       139471 30026635.14 17307181.097
##  2        7        16914  3456917.71  2091054.593
##  3        5         4728  2293140.51  1362891.759
##  4       10         1172   268772.40    77164.446
##  5        3           56    47695.48    26900.718
##  6        2           53    43960.44    25011.718
##  7        6          304    43437.44    19403.014
##  8        4          203     5278.71     2572.440
##  9        8           58     3945.08     2231.268
## 10       11           69     2757.27     1049.588
## 11        9          140     1074.91      575.974
```

```r
#utiliza o algoritmo k-means da função padrão em R com 3 clusters (grupos)
#k-means é um algoritmo de agrupamento não supervisionado
km = kmeans(x, 3)


#plota um grafico que mostra 6 comparacoes:
#categoria de cada produto vs. quantidade de cada produto vendido
#categoria de cada produto vs. preco total das vendas de cada produto
#categoria de cada produto vs. valor total do custo liquido de cada produto
#quantidade cada produto vendido vs. preco total das vendas de cada produto
#quantidade cada produto vendido vs. valor total do custo liquido de cada produto
#preco total das vendas de cada produto vs. valor total do custo liquido de cada produto
plot(x, col = km$cluster+1, main = 'Resultado agrupamento com 3 clusters', pch = 20, cex = 3)
```



É possível perceber que a categoria 1 (ponto vermelho no gráfico) é disparadamente a categoria mais vendida. Consequentemente, essa categoria fornece o maior valor de venda e o maior lucro. Além disso, nota-se que as categorias 7 e 5 (pontos verdes no gráfico) são a segunda e terceira categorias mais vendidas, respectivamente. Por fim, as demais categorias foram agrupadas (pontos azul no gráfico) em um mesmo cluster por terem valores totais de vendas relativamente próximos.

ANÁLISE DOS DADOS - previsão

```
#pacotes para análise de séries temporais
#install.packages("forecast")
suppressMessages(library(xts))
```

```
## Warning: package 'xts' was built under R version 3.4.2
```

```
## Warning: package 'zoo' was built under R version 3.4.2
```

```
suppressMessages(library(forecast))
```

```
## Warning: package 'forecast' was built under R version 3.4.2
```

```
#obtem os dados ordenados pela data de processamento (process_date)
analise2 = arrange(produtosVendidos, process_date)

#retira as duas primeiras linhas da tabela porque as datas sao invalidas: 0000-00-00
analise2 = analise2[-c(1, 2), ]
```

```r
#obtem um subconjunto para a categoria 1
#para cada dia, mostra:
#a quantidade de produtos vendidos,
#o valor total de vendas,
#o valor total do custo líquido de cada produto,
#o lucro obtido no dia
y = analise2 %>%
  filter(category == 1) %>%
  group_by(process_date) %>%
  summarise(qtd_produtos = sum(quantity),
            total_venda = sum(price),
            custo_liq = sum(liquid_cost),
            lucro = total_venda - custo_liq) #%> arrange(desc(lucro))


#mostra as 6 primeiras linhas do subconjunto obtido acima
head(y)
```

```
## # A tibble: 6 x 5
##   process_date qtd_produtos total_venda custo_liq    lucro
##         <fctr>        <int>       <dbl>     <dbl>    <dbl>
## 1   2016-06-01          140    36345.78  19540.65 16805.13
## 2   2016-06-02          181    49164.53  27142.53 22022.00
## 3   2016-06-03          249    59977.31  33492.85 26484.46
## 4   2016-06-04          231    57202.15  30233.34 26968.81
## 5   2016-06-05          164    43591.65  24715.46 18876.19
## 6   2016-06-06          233    60280.80  34124.94 26155.86
```

```r
#mostra as 6 últimas linhas do mesmo subconjunto
tail(y)
```

```
## # A tibble: 6 x 5
##   process_date qtd_produtos total_venda  custo_liq      lucro
##         <fctr>        <int>       <dbl>      <dbl>      <dbl>
## 1   2017-06-01          476    91875.80 46145.5545 45730.2455
## 2   2017-06-02          108    23023.75 11907.9070 11115.8430
## 3   2017-06-03           41     8993.37  4880.1967  4113.1733
## 4   2017-06-06            8     1277.12   751.4686   525.6514
## 5   2017-06-07            6      211.46    57.5133   153.9467
## 6   2017-07-11            1       83.36    68.9802    14.3798
```

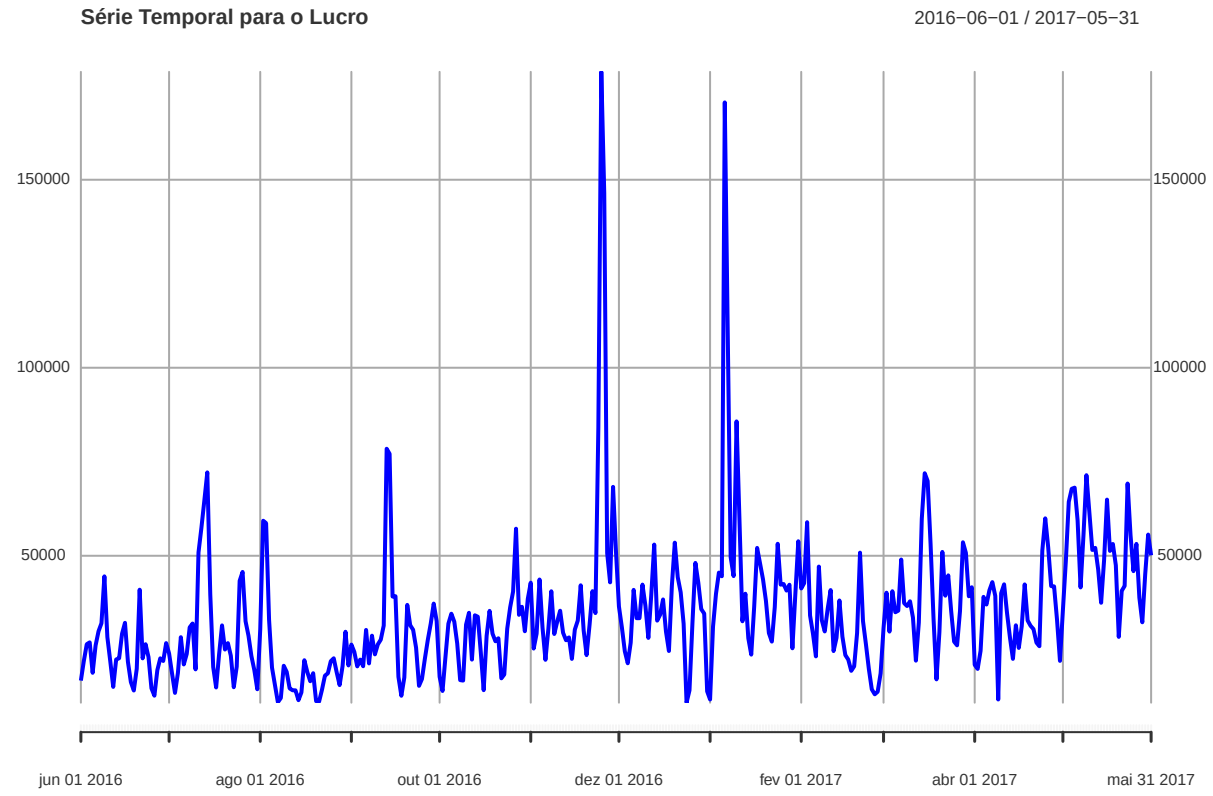
Na tabela acima, nota-se que a primeira linha refere-se a data em que completa exatamente 1 ano da data de início do período analisado (01 de junho de 2016). Portanto, todas essas 6 linhas mostradas acima serão ignoradas para que a análise de série temporal seja realizada dentro do período de 1 ano.

```r
#retira as 6 últimas linhas do subconjunto
#mostra que a última data agora é 31 de maio de 2017
y = y[-c((nrow(y)-5):nrow(y)), ]
tail(y)
```

```
## # A tibble: 6 x 5
##   process_date qtd_produtos total_venda custo_liq    lucro
##         <fctr>        <int>       <dbl>     <dbl>    <dbl>
## 1   2017-05-26          547   118288.60  65135.48 53153.12
## 2   2017-05-27          361    82943.10  44061.59 38881.51
## 3   2017-05-28          326    69053.18  36744.12 32309.06
```

```
## 4   2017-05-29          452    96817.81  51657.23 45160.58
## 5   2017-05-30          588   119291.53  63651.38 55640.15
## 6   2017-05-31          513   107530.48  57410.51 50119.97
```

```
#mostra uma serie temporal para o lucro da categoria 1
serie_lucro1 = xts(y$lucro, as.Date(y$process_date), frequency = 12)
plot(serie_lucro1, type = 'l', xlab = 'Data', ylab = 'Lucro',
     main = 'Série Temporal para o Lucro', col = 'blue')
```



Analisando o gráfico acima, percebe-se que o maior lucro obtido na categoria 1 foi no dia 25 de novembro de 2016, obtendo o valor de R$178.691,01. Ao fazer uma pesquisa rápida, foi possível constatar que o black friday aconteceu nesse dia. Portanto, isso nos fornece uma forte evidência de que a categoria 1, além de ser a mais vendida durante todo o período analisado, tende a ser a categoria mais vendida no black friday. Consequentemente, a categoria 1 fornece o maior lucro dentre todas as categorias de produtos.

A tabela abaixo mostra (em ordem decrescente) os 6 dias em que a categoria 1 forneceu os maiores lucros (lucro = total_venda - custo_liq). Pelos valores obtidos, nota-se que os últimos dias de novembro e os primeiros dias de janeiro tendem a ser os dias em que mais vende-se os produtos da categoria 1. Em consequência disso, obtem-se os maiores lucros.

```
## # A tibble: 6 x 5
##   process_date qtd_produtos total_venda custo_liq     lucro
##          <fctr>        <int>       <dbl>     <dbl>     <dbl>
## 1   2016-11-25         2880    554991.9  376300.9 178691.01
## 2   2017-01-06         3007    453438.7  282874.6 170564.13
## 3   2016-11-26         2216    425208.7  278730.2 146478.51
## 4   2017-01-07         1839    288820.0  179376.7 109443.32
## 5   2017-01-10         1299    220498.7  134762.2  85736.47
```

```
## 6   2016-11-24         1385    313192.3  229209.5  83982.79
```

Para determinar essa possível sazonalidade (tendência), torna-se necessário uma análise por parte de um especialista da área de negócios ou especialista em vendas da empresa. Entretanto, acredita-se que, devido a black friday acontecer na última sexta-feira do mês de novembro e ser o dia em que mais se vende produtos, os dias subsequentes a black friday possivelmente atraem clientes desejando realizar troca de produtos. Com isso, alguns clientes acabam trocando seus produtos por produtos mais caros e pagando a diferença.

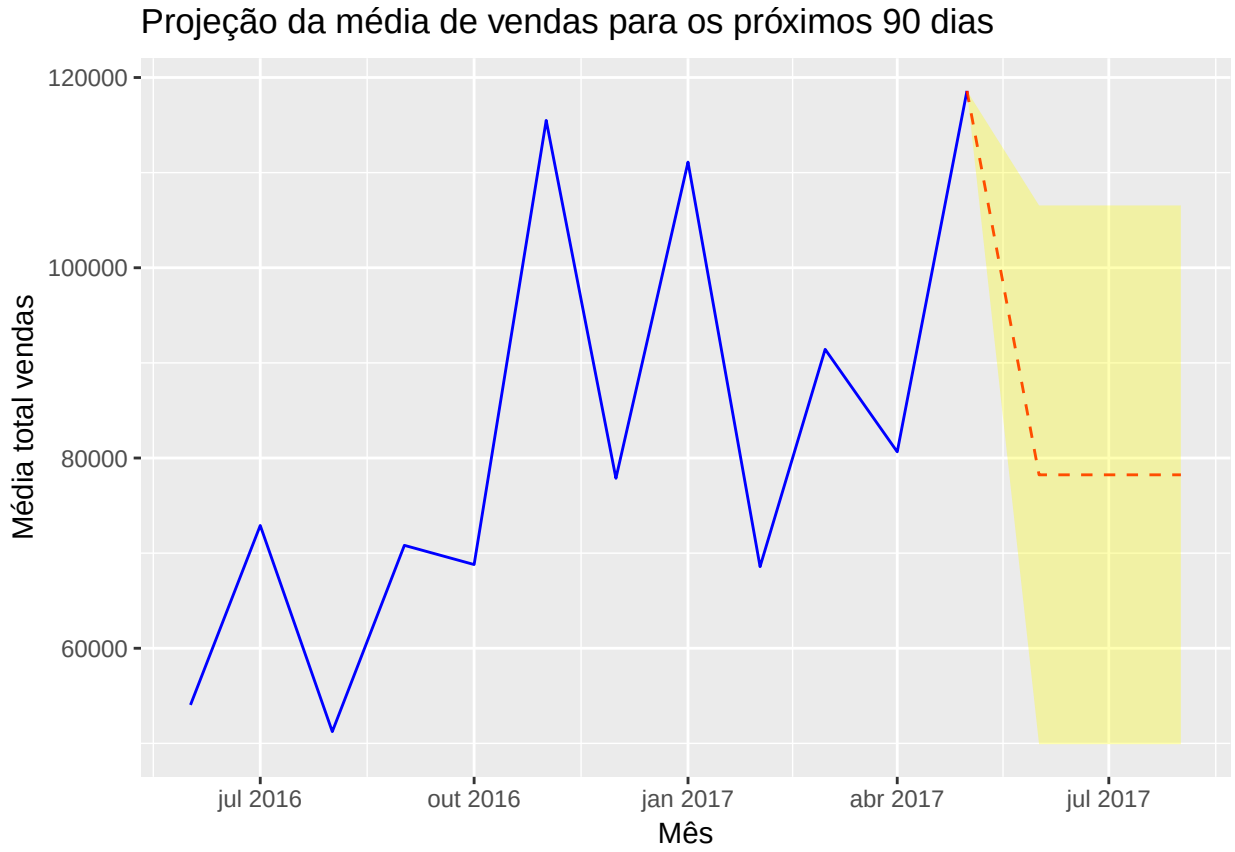
Nesse sentido, considera-se que essa mesma possibilidade de tendência aplica-se aos primeiros dias de janeiro devido a alguns clientes aproveitarem o término das festividades de natal e réveillon para efetuar a troca de seus produtos. Além disso, ao fazer uma rápida pesquisa, foi possível constatar que o Magazine Luiza realiza há 24 anos uma promoção chamada "Liquidação Fantástica". Essa promoção acontece nos primeiros dias do mês de janeiro. Logo, essa pode ser uma possível explicação para o caso em que o dia 06 de janeiro de 2017 ter sido o dia que forneceu o segundo maior lucro.

O trecho de código abaixo calcula e gera um gráfico com uma previsão (forecast) da média do total de vendas de produtos da categoria 1 para os próximos 90 dias (junho, julho e agosto de 2017).

```
#install.packages("ggfortify")
suppressMessages(library(ggfortify)) #pacote para recursos avançados de gráfico
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
#a média do total de vendas para cada mês foi calculada separadamente
#devido ao período das datas abranger dois anos diferentes (2016 e 2017)
mes = c('2016-06-30', '2016-07-31', '2016-08-31', '2016-09-30', '2016-10-31',
        '2016-11-30', '2016-12-31', '2017-01-31', '2017-02-28', '2017-03-31',
        '2017-04-30', '2017-05-31')
media_totalVenda = c(54043.64, 72910.23, 51237.94, 70816.98, 68800.35, 115493,
                     77868.55, 111107.1, 68578.55, 91426.53, 80651.83, 118593.3)

#obtem uma tabela com a média do total de vendas para cada mês
df = data.frame(mes, media_totalVenda)


#obtem uma série temporal
ts_datas = ts(df$media_totalVenda, start = c(2016, 6), frequency = 12)
#plot(ts_datas)
ets_datas = ets(ts_datas)
f_ets = forecast(ets_datas, h = 3)
#plot(f_ets)

#plota um gráfico da série temporal com a previsão
autoplot(f_ets, ts.colour = 'blue', predict.colour = 'red', predict.linetype = 'dashed',
         conf.int = TRUE, conf.int.fill = 'yellow') +
         ggtitle('Projeção da média de vendas para os próximos 90 dias') +
         labs(x = 'Mês', y = 'Média total vendas')
```



O gráfico acima foi gerado utilizando o modelo ETS do R, que implementa o modelo estatístico "Exponential Smoothing". Este modelo gera no gráfico os valores médio, máximo e mínimo do intervalo de predição. É possível verificar no gráfico que a previsão da média do total de vendas na categoria 1 para os próximos 90 dias é de R$78.234,50. Isso representa um ganho de aproximadamente 35% em comparação ao mesmo período do ano anterior, cuja a média do total de vendas de produtos da categoria 1 para os meses de junho, julho e agosto de 2016 foi de R 59.397,27.

Considerando que a categoria 1 é disparadamente a categoria mais vendida, conclui-se que a escassez de produtos dessa categoria no estoque implica diretamente na redução significativa do total de vendas e, consequentemente, do lucro da empresa. Portanto, devido a essa discrepância (diferença) considerável entre a categoria 1 e as categorias 2, 3, 4, . . . , e 11, as demais categorias não serão analisadas neste relatório.

O trecho de código abaixo obtém e mostra (apenas) os 6 primeiros produtos mais vendidos da categoria 1.

```
#obtem a quantidade de vendas para cada produto da categoria 1 em ordem decrescente
w1 = produtosVendidos %>%
  filter(category == 1) %>%
  group_by(code) %>%
  summarise(qtd_itensVendidos = sum(quantity)) %>%
  arrange(desc(qtd_itensVendidos))

head(w1)
```

```
## # A tibble: 6 x 2
##    code qtd_itensVendidos
##   <dbl>             <int>
## 1    25             19654
## 2    28              8483
```

```
## 3    27              8020
## 4    63              5550
## 5   118              5521
## 6    18              5501
```

Com isso, é possível observar que o produto cujo o código é igual a 25 (ou 2e35421c34fb588ba40a0c57b3971d24 no arquivo original) é o mais vendido. Logo, a existência desse produto em estoque é essencial para manter a média do total de vendas da categoria 1 e, consequentemente, a média do lucro da empresa.

CONCLUSÕES

Esse projeto, em forma de um relatório técnico-científico, visou realizar uma análise descritiva e preditiva de um conjunto de dados que representa quase 180000 vendas de um dos maiores varejistas do Brasil. Foi possível concluir que, dentre as 11 categorias de produtos existentes, a categoria 1 é consideravelmente a categoria que contém os produtos mais vendidos. Em consequência disso, essa categoria é capaz de fornecer os maiores lucros para a empresa.

Poderia ter sido feito também uma análise individual das outras categorias. No entanto, a discrepância entre a categoria 1 e as demais categorias, em termos de total de vendas e lucro, é enorme. Com isso, espera-se que as análises dessas categorias poderá fornecer resultados muito próximos entre si. Então, devido a esses detalhes e ao elevado número de páginas desse relatório, tais análises poderão ser realizadas em um próximo projeto.

Como o autor desse relatório não é um especialista em vendas de produtos e nem um especialista na área de negócios da empresa, não considera-se possível afirmar um número exato da quantidade de peças de cada tipo que poderiam ser compradas. Por outro lado, de forma empírica e baseando-se nos insights obtidos por meio das análises, é possível apenas concluir que o produto mais vendido da categoria 1 não deve faltar em estoque.

Segue abaixo um trecho de código capaz de obter a quantidade de venda de todos os produtos em todas as categorias. Baseando-se nesses números, o profissional responsável pela gestão do controle de estoque poderá tomar decisões mais facilmente.

```
#obtem a quantidade de vendas para todos os produtos
w2 = produtosVendidos %>%
  group_by(code) %>%
  summarise(qtd_itensVendidos = sum(quantity)) %>%
  arrange(desc(qtd_itensVendidos))

#mostra somente as 6 primeiras linhas da tabela obtida acima
head(w2)
```

```
## # A tibble: 6 x 2
##    code qtd_itensVendidos
##   <dbl>             <int>
## 1    25             19654
## 2    46             16361
## 3    28              8483
## 4    27              8020
## 5    63              5550
## 6   118              5521
```

Para fins de comparação, a tabela abaixo mostra o mesmo resultado da tabela acima, porém com o código original de cada produto.

```
## # A tibble: 6 x 2
##                              code qtd_itensVendidos
##                            <fctr>             <int>
```

```
## 1 2e35421c34fb588ba40a0c57b3971d24             19654
## 2 4534ea61b50410b3b6243e02b40c8cd1             16361
## 3 3454ea52396a4cfd3fc37414d30c7b9c              8483
## 4 32ceebf3efea1d04ace4183d20d4da5b              8020
## 5 5b7a30a9e6a43b170ad4d9e00d8d9359              5550
## 6 d57911cca4b08f7b46417d952c0ca1dc              5521
```