Desafio

Henrique Lispector *lispe001@umn.edu*

10 de outubro, 2017

Subindo os dados para o R

```
library(readr)
desafio <- read_csv("~/Desktop/desafio.csv")
```

```
## Parsed with column specification:
## cols(
##   order_id = col_character(),
##   code = col_character(),
##   quantity = col_integer(),
##   price = col_double(),
##   pis_cofins = col_double(),
##   icms = col_double(),
##   tax_substitution = col_double(),
##   category = col_character(),
##   liquid_cost = col_double(),
##   order_status = col_character(),
##   capture_date = col_date(format = ""),
##   process_date = col_date(format = ""),
##   process_status = col_character(),
##   source_channel = col_character()
## )
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 24493 parsing failures.
## row # A tibble: 5 x 5 col     row          col   expected     actual                    file expecte
## ... ................. ... ..................................................................... .......
## See problems(...) for more details.
```

Pré-Processamento dos Dados

Antes de tudo, como os algoritmos presentes nesta análise utilizam-se de algumas variáveis randômicas, farei uso da função "set.seed()", para que o código produza as mesmas variáveis randômicas e os mesmos resultados, no caso de ser executado por terceiros.

```
set.seed(5)
```

Ao abrir o desafio.csv, a minha primeira atitude foi tentar entender o que cada coluna significa. Para isso, antes de mais nada, quis saber como era o dataset vendo suas primeiras linhas, que não foi muito útil pelo fato de o dataset estar codificado como classe tibble. Então, em seguida, usei o comando View() para ver o arquivo como um todo, que está comentado, pois gera um output muito grande. Quando se trabalha com dados de maior tamanho, o comando View() não é recomendado e transformar a classe do dataset para dataframe seria o mais indicado.

```
head(desafio)
```

```
## # A tibble: 6 x 14
##                            order_id                              code
##                               <chr>                             <chr>
## 1 bcb59c839e78b2601374cbad9239ca7b e6762ba2ffbca07ab6cee7551caeaad5
## 2 4e91ee6b95895771dc9ee524e910a902 e6762ba2ffbca07ab6cee7551caeaad5
## 3 88eb0ac86af1a521c0831298d22dea8b e6762ba2ffbca07ab6cee7551caeaad5
## 4 dee418152a36314b4aee6ce9cf94fcbf e6762ba2ffbca07ab6cee7551caeaad5
## 5 1c175bc61b9b659bbf011b2e5e3dcec6 e6762ba2ffbca07ab6cee7551caeaad5
## 6 a8ad36828898fa3f6efeb5bd19c076f2 e6762ba2ffbca07ab6cee7551caeaad5
## # ... with 12 more variables: quantity <int>, price <dbl>,
## #   pis_cofins <dbl>, icms <dbl>, tax_substitution <dbl>, category <chr>,
## #   liquid_cost <dbl>, order_status <chr>, capture_date <date>,
## #   process_date <date>, process_status <chr>, source_channel <chr>
```

```
#View(desafio)
```

Para entender melhor o que cada coluna significa, verifiquei a classe que cada variável está codificada e se a classificação faz sentido.

```
lapply(desafio, class)
```

```
## $order_id
## [1] "character"
##
## $code
## [1] "character"
##
## $quantity
## [1] "integer"
##
## $price
## [1] "numeric"
##
## $pis_cofins
## [1] "numeric"
##
## $icms
## [1] "numeric"
##
## $tax_substitution
## [1] "numeric"
##
## $category
## [1] "character"
##
## $liquid_cost
## [1] "numeric"
##
## $order_status
## [1] "character"
##
## $capture_date
## [1] "Date"
##
## $process_date
```

```
## [1] "Date"
##
## $process_status
## [1] "character"
##
## $source_channel
## [1] "character"
```

Vemos que “order_id”, “code”, “category”, “order_status”, “process_status” e “source_channel” estão classificados como “character”, o que não vai nos dizer muita coisa quando usarmos a função “summary()” na sequência. Assim, devemos trocar a classificação dessas variáveis para “factor”. Além dessas variáveis, a variável liquid_cost obviamente é númerica, mas vou trocá-la para “factor” para saber quantos números distintos essa variável possui.

```
desafio1 <- with(desafio,
  data.frame(order_id = factor(order_id), code = factor(code), quantity = quantity, price = price,
  pis_cofins = pis_cofins, icms = icms, tax_substitution = tax_substitution,
  category = factor(category), liquid_cost = factor(liquid_cost), order_status = factor(order_status),
  capture_date = capture_date, process_date = process_date,
  process_status = factor(process_status), source_channel = factor(source_channel)))
```

Verificamos se a mudança foi efetuada com sucesso.

```
lapply(desafio1, class)
```

```
## $order_id
## [1] "factor"
##
## $code
## [1] "factor"
##
## $quantity
## [1] "integer"
##
## $price
## [1] "numeric"
##
## $pis_cofins
## [1] "numeric"
##
## $icms
## [1] "numeric"
##
## $tax_substitution
## [1] "numeric"
##
## $category
## [1] "factor"
##
## $liquid_cost
## [1] "factor"
##
## $order_status
## [1] "factor"
##
## $capture_date
```

```
## [1] "Date"
##
## $process_date
## [1] "Date"
##
## $process_status
## [1] "factor"
##
## $source_channel
## [1] "factor"
```

Sim, as mudanças de classe forem realizadas com sucesso. Neste momento podemos aplicar a função "summary()".

```
summary(desafio1)
```

```
##                               order_id
##  3e3225b6aec78dbdd8ce73773764a1b2:     5
##  a95921da5207ed69e7ef05fefb5ae415:     5
##  e0459fb2a1b29137204e07f3c57e4b95:     5
##  050e40476e96a8781424e3a3def8fd9d:     4
##  0a05f32038b22f199dedd5ad25ea923e:     4
##  2378e930208fc2abd32833e65c322897:     4
##  (Other)                         :179122
##                                 code           quantity
##  2e35421c34fb588ba40a0c57b3971d24: 20944   Min.   :  1.000
##  4534ea61b50410b3b6243e02b40c8cd1: 17225   1st Qu.:  1.000
##  3454ea52396a4cfd3fc37414d30c7b9c: 10674   Median :  1.000
##  32ceebf3efea1d04ace4183d20d4da5b:  8889   Mean   :  1.055
##  0671c2b9132a3f5215a4212ce0691694:  6496   3rd Qu.:  1.000
##  5b7a30a9e6a43b170ad4d9e00d8d9359:  6333   Max.   :100.000
##  (Other)                         :108588
##      price             pis_cofins           icms          tax_substitution
##  Min.   :    1.03   Min.   :   0.00   Min.   :   0.00   Min.   :  0.00
##  1st Qu.:  149.91   1st Qu.:  10.64   1st Qu.:   0.00   1st Qu.:  0.00
##  Median :  194.40   Median :  17.52   Median :  21.49   Median :  0.00
##  Mean   :  234.64   Mean   :  19.53   Mean   :  25.10   Mean   : 17.87
##  3rd Qu.:  309.36   3rd Qu.:  28.16   3rd Qu.:  38.88   3rd Qu.: 30.40
##  Max.   :19993.00   Max.   :1849.35   Max.   :3598.74   Max.   :280.83
##
##                               category        liquid_cost
##  388128822cef4b4f102ae881e040a64b:153943   213.4382:20944
##  9a97178a18aa6333aabdfb21de182b99: 18009   119.5065:17225
##  90cc5bdd050bcd7cf0d50d079d0fda66:  5208   105.3557:15161
##  dda10a917a9ea3120e5d299af5105290:  1053   205.8997:13754
##  98f679396a60f117b171ddedfcc3e5ed:   330   116.6273:10310
##  61ad270def6d4b2403f4536f39cff29a:   200   117.082 : 8606
##  (Other)                         :   406   (Other) :93149
##                                order_status     capture_date
##  entrega total                       :144745   Min.   :2016-06-01
##  cancelado boleto n<U+00E3>o pago    : 18425   1st Qu.:2016-10-05
##  cancelado                           :  5274   Median :2016-12-26
##  em rota de entrega                  :  2649   Mean   :2016-12-20
##  entrega parcial                     :  2516   3rd Qu.:2017-03-15
##  solicita<U+00E7><U+00E3>o de troca:  2400   Max.   :2017-06-01
```

```
##  (Other)                            :  3140
##   process_date           process_status
##  Min.   :2016-06-01   captado   : 24706
##  1st Qu.:2016-10-04   processado:154443
##  Median :2016-12-26
##  Mean   :2016-12-19
##  3rd Qu.:2017-03-16
##  Max.   :2017-07-11
##  NA's   :24493
##                              source_channel
##  b76eb9b8fc0f17098812da9117d3e500:81837
##  fc7020775a7cdf161ab5267985c54601:31239
##  a578e71c3216f513a84ec6a46084fd3a:20730
##  9d3e0fcbc1f16d80a76026e8f1c26002:17714
##  152bf0ce464047b9499ccb9e5b9b77a8: 7251
##  98defd6ee70dfb1dea416cecdf391f58: 5085
##  (Other)                          :15293
```

Posso inferir que a variável "code" pode significar produto, e que entre outras coisas, os seguintes pontos chamam a atenção:
* A variável "order_id" se repete muito pouco e no geral é única para cada pedido;
* A variável "category" possui uma categoria com quase 85% dos produtos;
* As datas dos pedidos vão de 01/junho/2016 até 11/julho/2017;

Através do código abaixo, consigo saber que há exatamente 131 produtos ("codes") distintos no total no arquivo desafio.csv que foi fornecido. Esses 131 produtos possuem 97 custos ("liquid_cost") diferentes, ou seja, alguns produtos de códigos diferentes vão ter o mesmo liquid_cost, há 11 categorias da variável "category", 17 categorias para a variável "order_status" e 16 para a variável "source_channel".

```
summary(desafio1$code, maxsum = 131)
```

```
## 0671c2b9132a3f5215a4212ce0691694 09f544ec2a74c89abeec7b0590fc2d11
##                             6496                              978
## 0ad316f6b5cb5e81ebff73ae2490ccfe 0bbe09e34a11e8e31cf49d6f8df2992d
##                              489                              341
## 0dca7ec6ba9b6e8f17f04f713a6be727 0f38be2df6854b4374f06cae1bc38482
##                              932                             2110
## 118484c270085e811fbbc81978a269b2 13b69fd4bf80b95756e3b138c9169a7f
##                              469                              580
## 174ef25d9556d516a813e2972f3b8351 193628b6634713730d3c506f2da0ff58
##                              603                             3597
## 1a225367d52023424b6f4b2aa8632615 1b3980ee40dc5d60ecae3b19cd41f49a
##                              352                              667
## 1c234775cae774823f38abe6721e61a4 1c619f92929dbaa41df54608dc70a7ce
##                             1167                              217
## 1dbe25b2fd344aed0c444fd6b715525b 1f12f1e1b9e7a20d4ad9dd549ec072c0
##                              450                              205
## 213005fe5d815c539812614f1f6a7768 23056d85a1c9115cd021a6ad35c84aff
##                              168                             5973
## 24549e47832f72f647d40a86b43b6925 270dcdb08fc6470a6976f43b8169b0f6
##                              164                                2
## 29424aaf6e27a8dbe4b7273a0a39131d 29f175b6bccc264e8141481dcda8102d
##                              219                             1115
## 2ab0e87dbce6ac45502aa1d2a8265933 2bc9e1d807d8f9187b8650acd35a14e5
##                              446                                3
## 2e35421c34fb588ba40a0c57b3971d24 30b6a9b1621f558d788ffb3d07a19281
```

```
##                             20944                             1294
## 32ceebf3efea1d04ace4183d20d4da5b 3454ea52396a4cfd3fc37414d30c7b9c
##                              8889                            10674
## 3657af9de7395eaba0dbcbcaa6fd90be 373cc1cfc10a45488be6b97bd5e94c44
##                               182                              254
## 374e1947dcb8f4848f4ada6f04921edd 37a62c0ad48679cee5554655de294721
##                               399                              228
## 37e2a39e829495d13b6cbb5320413e48 390943ce05959ac98c702d250c2ebb54
##                              1024                              178
## 3a36dc63a58442085d0ccd98c4f9c64a 3b4407288e2983a514a241c9b84b7094
##                               349                               11
## 3bc993e0f0c636e9aaaefa0356bdecc0 3d21b63892749e921e3ff5818753bd67
##                               458                             1316
## 3d3d13446c52ecaaed5d0bf55a933d4d 3da22f1b88a20ea8efc3d83fcb872e21
##                               179                              469
## 40bddb00475d65eddb68e9aeb6fab0de 40d98a2375332cb635d4cb28ab68e087
##                              1731                              397
## 411e1404e183718207628232e91ce5a9 41e2bee39c1d3ef52fcedd69d0ab8c8c
##                                82                               59
## 42920a6f15855662ee9a272fbe44cbd1 4534ea61b50410b3b6243e02b40c8cd1
##                               592                            17225
## 453d427550d3816e446d4882bf67a75f 4557c7e5af70efd2e3ca2befd59ccdc3
##                               850                             1486
## 498556cd1dade09f21fea97d3c916875 49d1447a5d1a218169eef2ea58cdc149
##                              1214                              450
## 4b58730cdc153f4eeda0d1321e630ba4 4ceedf57303e127d31a164c7ae5791d8
##                               963                              685
## 4ef261b089aa567ef24778fae254bfe2 51016cd454b391dd0b0e23b2b16b6fc3
##                               273                              326
## 525848b647262de5fd7be193b17cdaac 54209126056016c7c391c0c8fd8e6eff
##                               769                              352
## 547bbdaa191bdb83d5b5b376bf2402fb 5490e2abafb28c022e53b55ba6641122
##                               906                              362
## 55447a73ff140176f4210347854c71f1 5837d4e32e5382af65d33b4dfbd9f561
##                              1270                              710
## 585eead66aaf3b2e140d480979184ae6 59af761735fbd646f8a8f98dc88f6fab
##                               582                               69
## 5b7a30a9e6a43b170ad4d9e00d8d9359 5bcebbc4f704cffaf2e6ccaf1d1fb5c7
##                              6333                               81
## 5e39201e582b1bb89cae7f650e4330c8 5e4c14883e5a606fac56b2630da5dca7
##                              2224                             1267
## 5fb59bee157577ee04a269e52af88598 5ff2d52d31bf9ad42924a35c4729ffdb
##                               704                              346
## 600c486af5e3f2d1624d2fc872e20e16 60424117a2618c7184687046fa5693c4
##                               537                              183
## 618ea23457368cf423b03d6b8fede992 630b051b117d509fc00a7c72328b1ddf
##                               196                                7
## 6411de56554a998e78ae1317fc956e19 6ace63708a91744bea8a68311bb3506c
##                               357                              361
## 6c82ad0e791258434fd42c51409b0239 6e6ae6bc648c6a00b4a4b7a8e976a41a
##                              1427                              451
## 7162b3182cd93dd197257b0339627554 723f73c85e91fc31d147dfade389d4f9
##                               200                              657
## 727673fa3e457bc596532b3eb26b23a0 75caad099f6f8c205e22e93826732c22
```

```
##                              3737                              666
## 760693745e10b0c5e68c42214c729b0d 7c829a5a8a0e4408b682f4394783483e
##                              4865                              140
## 7da116bd1d42f3475803402e710253cf 7e3713530b46887cff58a2e2ac433ac5
##                              2094                             1047
## 8bdd60f700ed8368e66080eb6bb6d313 8ca06ab9bced028b2f180d4fa0527588
##                               193                              454
## 8e5d2c3f2476cb5c507dd0f00f6eabda 9cdf9e07e226869d8da4a3e10ac65c62
##                                75                             1197
## 9e5dd3c1d252136c4351b84589dae2b5 a0c96bb1b4117eeeae27b77a1381f55f
##                               425                             3977
## a2018dae10d736a66eea5a0a349ef9ee a36f5dfa4f08cdfe64594061ba76f30e
##                              1421                               91
## a586c072b4a3958ac72a6fff71730cb3 a7772a34be22f0fd09f0ef36b6eb337f
##                              1100                              134
## abf2d3cb446492ee7897087db9a0b2a0 ae5a740ca930b9149f590179b0dde3f3
##                               784                              830
## b08b7321c4db8f45a1a97a79d1e44dd8 b272ba3f4adb1dd16eaac1b53940629e
##                              4487                               31
## b367e71967b4701550ffce69e729ee6f b3fca69c356bec270dc3a90f8b498883
##                               688                                6
## b763aecfcfc8cbac2c3a51488d33480a b9d929195dcd4e6a36e5e65891746b5e
##                              1005                              562
## bc4d7296a37f5dfa0944b3274229b0a0 bc97d7cb7c13c2a7ff4558bb12a3f047
##                               462                              154
## c254dc11afbcc091678f0ab49a02e7ad c27a276b623c751a6b0a8ad6243d681d
##                              1469                               51
## c32ada18ec4f2992e8c003ffeb73b97b c348d9bc6a152d8d34489a79ab9452a7
##                               864                              446
## c443d252c048280160fc427766d9f1f4 c80a47e4c53432d9ee8cd98c9ce13769
##                               168                              967
## c85bd735a203c905716bc8fba284d02a c85f192a81fa83780e5c012175749eee
##                                58                              774
## cbeb3a98c1c9b01522db6ee2128ed805 ce4baabfbcb1d43e22f7ba44b49f2714
##                               499                              108
## d32bc6c4069b86ba9e9d7d651dbf1a1e d408e1b5e841dde4e15a4cfa182e3812
##                              1220                               80
## d4592ab52cb9cd5af0510943a4c8e28c d57911cca4b08f7b46417d952c0ca1dc
##                              1067                             5809
## d5bc9e14d090330cd07e6ccbcb3c3e4e d709ff164dc53eb7b8470e84c2b60974
##                              4017                              144
## dd1935ffd0ee2b6ec159ba7867d11e57 e0b4fdba39475c05a3ac9e769cce9bfa
##                               881                              463
## e13f7f001fe2b1af072a3d50d3058284 e6017ce062eb324ab446e9983afba369
##                              1883                              156
## e6762ba2ffbca07ab6cee7551caeaad5 ee2b4e97025f5ca840570265c4288c99
##                                57                              438
## f08984b2adcbf33ba61fe13fcfa5b957 f5f92c2a12f182115c45288a6ef28e94
##                               424                              485
## f9a023f31c8087fd0c169b3bedd351d1 fd84644da59504bd9e9dcb4b6db63bea
##                               691                             2917
## ffdad3ddbaf6c76c9bba1b48c51e03c6
##                              1113
```

```
#table(desafio1$source_channel)

summary(desafio1$liquid_cost, maxsum = 97)
```

```
##   4.1141   6.7456  10.2705  12.8622  16.6601  17.8758  27.2847  32.2506
##      140        7       11      200       59        3     3156     1893
##  33.9367  36.5302  36.5678  37.6516  37.7795  37.8157  39.9637  41.6587
##      537     4664     1115      144     1294       81      219      530
##   42.544   42.909  43.8556  45.5046  45.6096  46.7958  47.0044  48.5266
##      156       58      205      154      450       51      580     5809
##  50.4232   53.823  54.1711  55.4998  57.5133  59.7151  60.0668  60.3155
##      183      437      178     2195     1150      346      673     1268
##  60.9701  64.5115   64.932  68.9802  72.4568  72.7082  73.8002  73.9351
##     1220      592      463     2733      458     1058     1428     1163
##  75.9576  76.8654  78.8621  81.6082  82.7021  86.1829  86.2419   88.535
##     5973     1316     1267     2080     1005      667     1779     1047
##  88.9639  90.9232  91.9564   92.701  94.0023  94.2809  97.7355 101.3513
##     1007     1651      662     1648      520      399       31      846
## 105.3557 106.1157 106.4842 106.6572 110.4704 116.6273 117.0742  117.082
##    15161     1100     1353      945      603    10310      906     8606
## 119.5065 124.0901 125.2397 134.6823 137.8216 141.4304 142.3325 168.8458
##    17225     1470      438      108      641       80     2094     1214
##  184.239  184.311 185.3262 191.2586 191.5473 199.3086 205.8997 213.4382
##     1036     1486     2917     1421      134       82    13754    20944
## 219.1066 229.3837 229.5249 234.8044 241.6715 285.8913 289.3563 387.1154
##     1729     4017     3737      671      867      446      657      881
## 414.0702 474.4175 496.9297 542.7065  550.952  564.595 636.5652 641.0867
##      425       69     1270       57      469      784        6        2
## 896.6814
##       75
```

```
summary(desafio1$category, maxsum = 11)
```

```
## 388128822cef4b4f102ae881e040a64b 4ece547755cba9e7fc14125bc895f31b
##                           153943                               64
## 568696c0b6828f77884ea8628fcc6200 61ad270def6d4b2403f4536f39cff29a
##                               71                              200
## 90cc5bdd050bcd7cf0d50d079d0fda66 98f679396a60f117b171ddedfcc3e5ed
##                             5208                              330
## 9a97178a18aa6333aabdfb21de182b99 9cfa7aefcc61936b70aaec6729329eda
##                            18009                               58
## d7ecf0071e88c21e993da125d2229a51 dda10a917a9ea3120e5d299af5105290
##                              140                             1053
## f79dccaf0b890eced0724d8563948c4f
##                               73
```

```
summary(desafio1$order_status, maxsum = 17)
```

```
##                            cancelado
##                                 5274
##      cancelado boleto n<U+00E3>o pago
##                                18425
##           cancelado dados divergentes
##                                  674
##           cancelado fraude confirmada
```

```
##                                     75
##         cancelado n<U+00E3>o aprovado
##                                      2
##      dispon<U+00ED>vel para retirada.
##                                    223
##    em rota de devolu<U+00E7><U+00E3>o
##                                     25
##                     em rota de entrega
##                                   2649
##                        entrega parcial
##                                   2516
##                          entrega total
##                                 144745
##                       fraude confirmada
##                                    163
##                pendente processamento
##                                      5
##                              processado
##                                    888
## solicita<U+00E7><U+00E3>o de cancelamento
##                                   1022
##    solicita<U+00E7><U+00E3>o de troca
##                                   2400
##                     suspeita de fraude
##                                     31
##                      suspenso barragem
##                                     32
```

```
summary(desafio1$source_channel, maxsum = 16)
```

```
## 152bf0ce464047b9499ccb9e5b9b77a8 2934a86a91bfa55d7f20b4f08a441fac
##                             7251                               14
## 3ab2427543039f8c9f17d06f6f65a3a7 5a97b8efd901c1d28ff86522b95babb9
##                             3183                             2638
## 5e38ada24380f6dc6094fbaac452b121 67c19e107de33cab7ea9a9db8bc9ccd2
##                                2                               87
## 7261d300057219056592010c7bdaf5ee 98defd6ee70dfb1dea416cecdf391f58
##                             4579                             5085
## 9d3e0fcbc1f16d80a76026e8f1c26002 a578e71c3216f513a84ec6a46084fd3a
##                            17714                            20730
## af082bb0c2fa1414655017d464aa0262 b76eb9b8fc0f17098812da9117d3e500
##                              826                            81837
## d0dd472c8e766eca296e556085d778f7 e9b49f9086ba813ca3f0b321710fef16
##                                1                                2
## ea2912716be1999ab62d5b9dfa4f58f9 fc7020775a7cdf161ab5267985c54601
##                             3961                            31239
```

Para entrar na análise de cluster, me perguntei quais fatores deveria levar em conta para poder agrupar os 131 códigos dos produtos. Ao analisar o dataset, logo percebi que um "code" pode ter: mais de um "price", mais de um "source_channel" e mais de um "order_status". Assim, decidi descartar essas variáveis, pois não caracterizariam de maneira única os produtos. Me perguntei também sobre o comportamento das variáveis "pis_cofins", "icms", "tax_substitution". Será que todas as linhas tinham imposto? Será que o imposto é um fator que vai diferenciar grupos de produtos? Talvez, mas através do código seguinte, verifiquei que há muitas linhas zeradas com esses três campos de imposto.

```
colSums(desafio1 == 0)
```

```
##         order_id             code         quantity            price
##                0                0                0                0
##       pis_cofins             icms tax_substitution         category
##            18433            74633           105491                0
##      liquid_cost     order_status     capture_date     process_date
##                0                0                0               NA
##   process_status   source_channel
##                0                0
```

Logo, decidi ignorá-los para a análise de cluster. Como verifiquei, através de tabelas dinâmicas no excel, que "category" era única para cada um dos 131 produtos, bem como "liquid_cost" era único para cada um dos 131 produtos, decidi que essas variáveis seriam levadas em consideração. Além de "category" e "liquid_cost", também cheguei na conclusão de que talvez a soma total da quantidade vendida de cada produto poderia ser uma variável que diferenciasse grupos de produtos. Como o "price" pode variar para um mesmo produto, decidi então levar em conta a soma do "price" para cada um dos 131 produtos, ou seja, quanto em dinheiro cada produto vendeu.

Portanto, rearrumei o arquivo csv para ficar apenas com as colunas que decidi levar em conta para a aplicação dos algoritmos de agrupamento. Percebendo que dos 179149 pedidos, apenas 154443 foram processados, todos os números no dataset seguinte partiram de 154443 pedidos. O dataset ficou da seguinte maneira:

```
library(readxl)
cluster <- read_excel("~/Desktop/cluster.xlsx", sheet = "cluster")
head(cluster)
```

```
## # A tibble: 6 x 6
##                              Code `Sum of quantity` `Sum of price`
##                             <chr>             <dbl>          <dbl>
## 1 c27a276b623c751a6b0a8ad6243d681d                47        4786.41
## 2 5bcebbc4f704cffaf2e6ccaf1d1fb5c7                76        6241.81
## 3 d709ff164dc53eb7b8470e84c2b60974               140       11094.43
## 4 a36f5dfa4f08cdfe64594061ba76f30e                85       11502.30
## 5 e6017ce062eb324ab446e9983afba369               138       11920.50
## 6 213005fe5d815c539812614f1f6a7768               195       16728.06
## # ... with 3 more variables: liquid_cost <dbl>, category <chr>,
## #   category1 <dbl>
```

```
#View(cluster)
```

O arquivo cluster.xlsx está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Clustering/cluster.xlsx. Como uma análise de cluster se baseia em medir distâncias entre linhas e pontos, a variável "category" foi transformada em número. A legenda dessa transformação pode ser encontrada no arquivo cluster.xlsx na aba "legend".
O código do produto não é numérico, então ele será retirado da seguinte forma, juntamente com a coluna original "category". A "category1" representará "category".

```
cluster1 <- cluster[,-c(1,5)]
head(cluster1)
```

```
## # A tibble: 6 x 4
##   `Sum of quantity` `Sum of price` liquid_cost category1
##               <dbl>          <dbl>       <dbl>     <dbl>
## 1                47        4786.41     46.7958         1
## 2                76        6241.81     37.8157         1
## 3               140       11094.43     37.6516         1
```

```
## 4                85       11502.30     60.0668         1
## 5               138       11920.50     42.5440         1
## 6               195       16728.06     41.6587         1
```

```
#View(cluster1)
```

Para termos uma ideia das variáveis neste novo dataset, utilizamos a função "summary()" novamente

```
summary(cluster1)
```

```
##  Sum of quantity    Sum of price      liquid_cost        category1
##  Min.   :    2.0   Min.   :    145   Min.   :  4.114   Min.   : 1
##  1st Qu.:  197.5   1st Qu.:  36599   1st Qu.: 55.500   1st Qu.: 1
##  Median :  477.0   Median :  83372   Median : 90.923   Median : 1
##  Mean   : 1245.6   Mean   : 276287   Mean   :134.417   Mean   : 2
##  3rd Qu.: 1053.0   3rd Qu.: 209993   3rd Qu.:141.881   3rd Qu.: 1
##  Max.   :19654.0   Max.   :6531395   Max.   :896.681   Max.   :11
```

Para uma análise de cluster, antes de mais nada, precisamos verificar se os dados das variáveis estão em uma escala relativamente parecida, ou se o range dos números varia muito de uma variável para outra. Baseado no summary acima, vemos que o range das variáveis é bem diferente. Nesse caso, precisamos realizar um processo de scaling das colunas, para que o efeito de dominância das variáveis com números maiores ou ranges maiores seja diminuito sobre variáveis com ranges menores. Ao deixar todas as colunas em uma escala parecida, a dispersão dos data-points é reduzida ao calcular distâncias, resultando num melhor desempenho do algoritmo de agrupamento. Utilizamos a função scale, da seguinte maneira:

```
cluster2 <- scale(cluster1) # [xi - mean(x)]/[sd(x)]
head(cluster2)
```

```
##      Sum of quantity Sum of price liquid_cost  category1
## [1,]      -0.4683029   -0.3874411  -0.6140806 -0.4197044
## [2,]      -0.4569719   -0.3853642  -0.6770164 -0.4197044
## [3,]      -0.4319657   -0.3784393  -0.6781664 -0.4197044
## [4,]      -0.4534554   -0.3778573  -0.5210727 -0.4197044
## [5,]      -0.4327471   -0.3772605  -0.6438787 -0.4197044
## [6,]      -0.4104760   -0.3703999  -0.6500832 -0.4197044
```

```
#View(cluster2)
```

Agora, calculamos as distâncias entre as linhas. Há vários métodos para calcular distâncias entre linhas como o Manhattan Distance, Max-Coordinate Distance, Jaccard Distance, etc. Mas o mais comum é a distância euclidiana, que é a que será utilizada nessa análise. Curta explanação sobre Distância Euclidiana: Imaginemos que temos uma linha A, com colunas a1, a2,..., ak. Imaginemos uma linha B, com colunas b1, b2,..., bk. A distância euclidiana nada mais é do que d(A,B) = SQRT(|a1 - b1|^2 + ... + |ak - bk|^2).

```
euclid <- dist(cluster2, method = "euclidean")
```

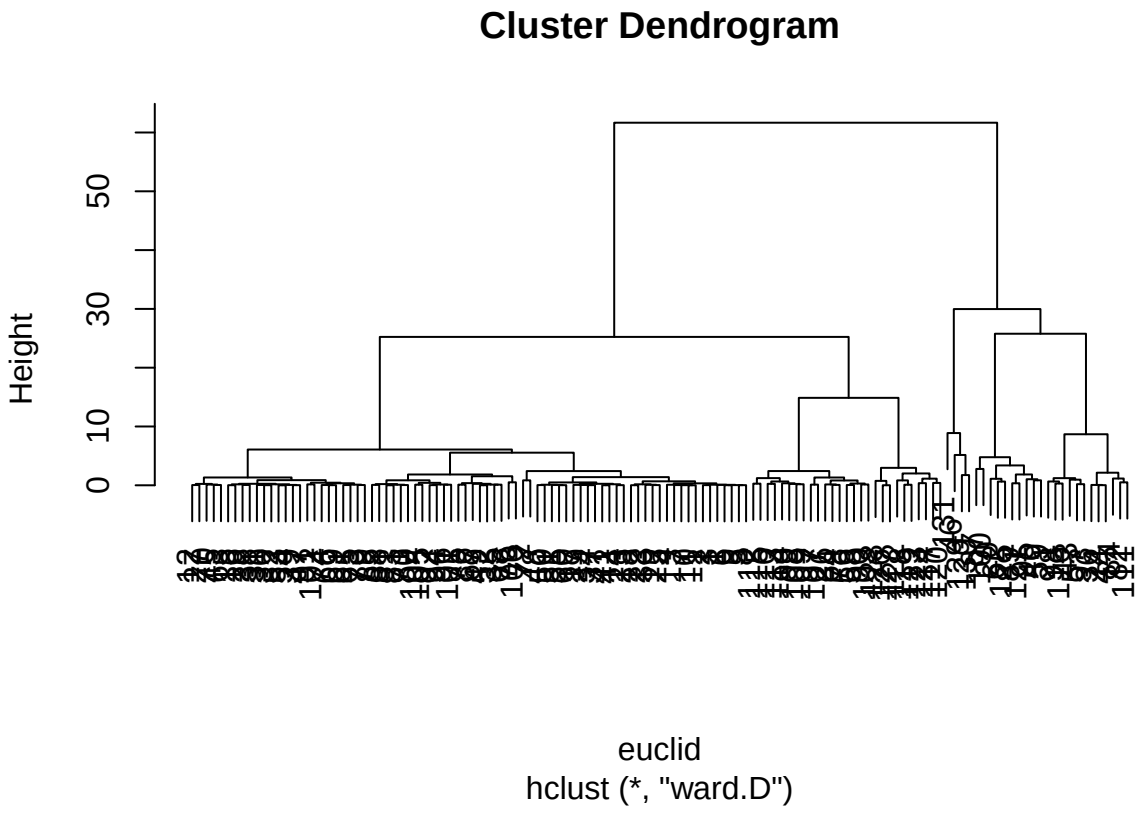
Finalmente, vamos usar o primeiro algoritmo de agrupamento não supervisionado chamado Hierarchical Clustering. Esse algoritmo começa com um cluster por linha e vai juntando uma linha a outra em grupos baseado no quão próximo é uma linha da outra, ou quão parecida é uma linha a outra, baseado na distância calculada, nesse caso euclidiana. Isso se repete até que todas as linhas sejam agrupadas em um único cluster. É um algoritmo computacionalmente pesado, mas produz um diagrama útil para visualização e útil posteriormente para um bom guess de cluster no algoritmo K-Means, que será abordado na sequência.

Começamos usando a função "hclust()", que leva como primeiro argumento a distância, nesse caso "euclid", e como segundo argumento o método, nesse caso será o Ward.D, que é um método comumente utilizado.

```
hierarchical <- hclust(euclid, method = "ward.D")
```

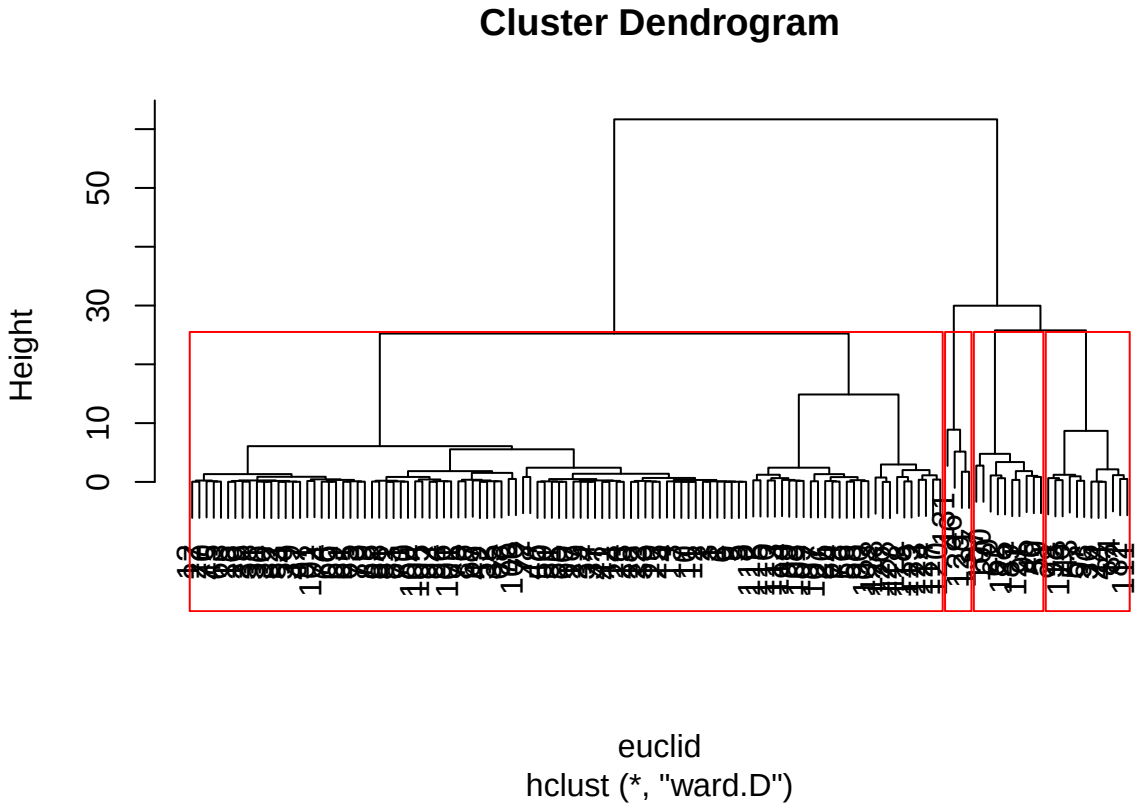
Agora vizualisamos o diagrama, que é chamado de dendograma:

```r
plot(hierarchical)
```



Notamos que pode-se existir 3 ou 4 clusters. Utilizando a função “cutree()”, podemos especificar a quantidade de grupos que queremos ver no dendograma. Em seguida, podemos visualizar esses grupos em vermelho da seguinte maneira:

```r
plot(hierarchical)
groups <- cutree(hierarchical, k = 4)
rect.hclust(hierarchical, k = 4, border = "red")
```



Bem, vemos que 4 grupos de produtos foram separados, mas quais as suas características? Esse é um defeito do algoritmo de hierarchical clustering, pois essas características não são reveladas. Mas podemos utilizar um algoritmo bastante famoso chamado K-Means para chegar nessas informações.

K-Means

O algoritmo K-Means funciona da seguinte maneira:

1) k data-points são escolhidos randomicamente para serem os centroids ou centros dos k clusters.

2) Atribuem-se outros data points para serem agrupados com o cluster com centroid mais próximo.

3) Recalcula-se o centroid de cada cluster.

4) Volta-se ao passo 2 até que os centroids atinjam convergência.

O objetivo é que as distâncias entre os data-points que pertençam a um mesmo cluster (within distance) sejam as menores possíveis, enquanto que as distâncias entre clusters diferentes (between distance) sejam as maiores possíveis.

Começamos criando dois vetores vazios, onde vamos guardar os valores WSS(within sum of squares) para cada loop do algoritmo, com k (número de clusters) entre 1 e o máximo número de clusters que os dados permitam. K=1 significa que teremos um grande cluster com todos os produtos, o que não faz muito sentido, enquanto k=131 siginifica que cada produto é um cluster, o que também não faz muito sentido. Então vamos definir k = 1 até 130. Vimos pelo Hierarchical clustering que o número de clusters deve ser em torno de 3, 4 ou 5 mas pode variar um pouco.

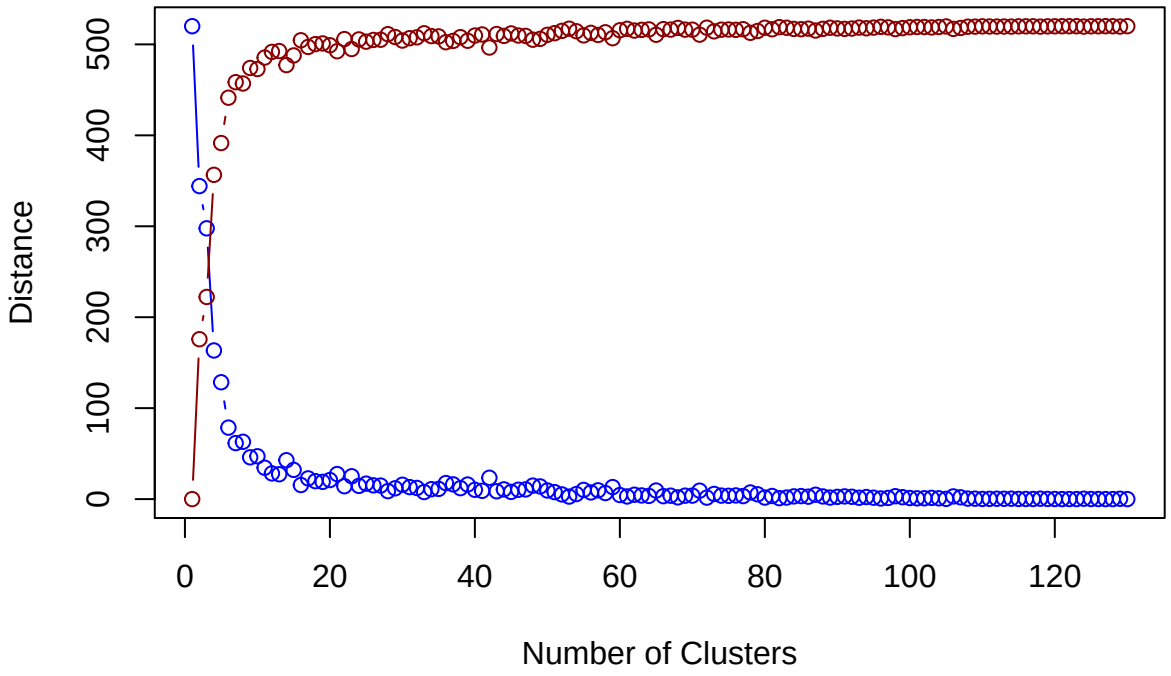
```
wss <- c(rep(0, nrow(cluster2)-1))
bss <- c(rep(0,nrow(cluster2)-1))
```

Agora vamos usar um loop para passar por cada célula dos nossos vetores “wss” e “bss”, aplicar o algoritmo k-means e guardar a soma total dos squared errors para o resultado. Então vamos guardar esse valor nos vetores wss e bss e passar para o próximo valor de k.

```
for (i in 1:(nrow(cluster2)-1)){
  kmeans_temp <- kmeans(cluster2, centers = i)
  wss[i] <- kmeans_temp$tot.withinss
  bss[i] <- kmeans_temp$betweenss
}
```

Plotamos os valores WSS vs k(número de clusters). Essa linha será em azul. O momento em que a curva se estabiliza, sugere que aumentar o valor de k além daquele ponto não agrega muito ao resultado, pois a distância within não diminui mais significativamente. Plotamos por cima, em vermelho, o gráfico do bss. Nesse caso, quanto maior o bss, melhor.

```
plot(1:(nrow(cluster2)-1), wss, type = "b",xlab = "Number of Clusters",
     ylab = "Distance", col = "blue")
par(new = T) #deixa o gráfico estático
plot(1:(nrow(cluster2)-1), bss, type = "b", axes = F, xlab = "", ylab = "",
    col = "dark red")
```




```
par(new = F)
```

Em ambos os casos, vemos que as curvas se estabilizam por volta de 4 a 6 clusters. Assim, vamos executar novamente o algoritmo escolhendo 4 clusters como solução.

```
partition <- kmeans(cluster2, 4)
centers <- partition$centers
centers
```

```
##   Sum of quantity Sum of price liquid_cost  category1
## 1     -0.36097148  -0.04707057   2.9862264  1.2591131
## 2     -0.35938906  -0.30520308  -0.4269323  2.4133002
## 3      5.24848304   5.42554863   0.3167626  0.4197044
## 4     -0.07380251  -0.11456135  -0.2423525 -0.4038665
```

Para fazerem sentido, os resultados precisam ser deescalados. Para tanto, criei a seguinte função:

```
#DESCALING COLUMNS
# x = column number, y = column reference name
descale <- function(x,y){
  for (i in 1:nrow(centers)){
    centers[i,x] <<- centers[i,x] * sd(y) + mean(y)
  }
}
```

Usando a função para deescalar cada variável:

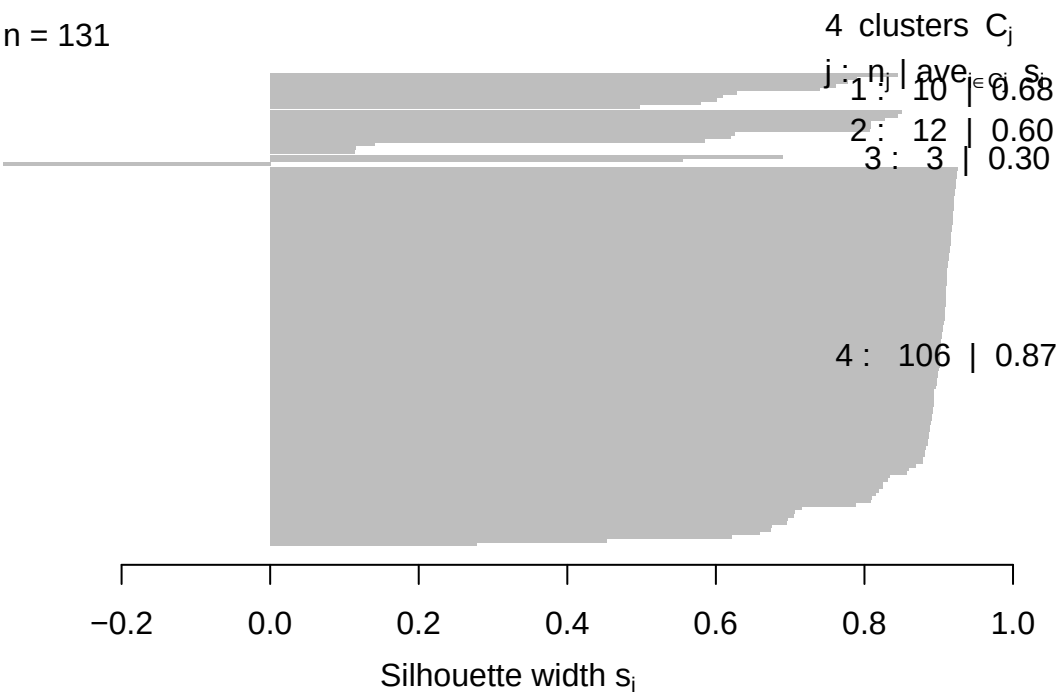
```
descale(1, cluster1$`Sum of quantity`)
descale(2, cluster1$`Sum of price`)
descale(3, cluster1$liquid_cost)
descale(4, cluster1$category1)
centers
```

```
##   Sum of quantity Sum of price liquid_cost category1
## 1          321.70    243302.27   560.51196  5.000000
## 2          325.75     62414.99    73.49938  7.750000
## 3        14678.33   4078259.71   179.61480  3.000000
## 4         1056.67    196007.86    99.83648  1.037736
```

Portanto, por exemplo, na média, os produtos vindos do grupo de category1 = 1.037 tiveram 1.056,67 unidades vendidas, geraram 196.007,86 de lucro e possuem um custo líquido individual de 99,83 e fazem parte da category1 1, que corresponde à category "388128822cef4b4f102ae881e040a64b". Já os produtos vindos do grupo de category1 = 5.0, tiveram, em média, 321,70 unidades vendidas, geraram 243.302,27 de lucro, possuem um custo líquido individual de 560,51 e pertencem à category1 número 5, que corresponde à category "90cc5bdd050bcd7cf0d50d079d0fda66". O fluxo de interpretação se repete para os dois clusters restantes, e a categoria deles pode ser consultada pela aba "legend" no arquivo cluster.xlsx .

Quanto à qualidade do agrupamento, podemos usar um gráfico de silhouette, chamado de silhouette plot, através do seguinte código:

```
library(cluster)
euclidean_sq <- euclid * euclid
plot(silhouette(partition$cluster, euclidean_sq))
```



Interpretamos o silhouette plot da seguinte maneira:

- $0 < 0.25$: Nenhuma estrutura significativa foi identificada no grupo
- 0.26 - 0.50: A estrutura identificada é fraca e pode ser artificial
- 0.51 - 0.70: Uma estrutura razoável foi encontrada
- 0.71 - 1.00: Uma estrutura forte foi identificada

Assim, vemos que uma estrutura forte foi identificada para 106 produtos, pois possuem um silhouette width de 0,87 . Para o grupo com 10 e o grupo com 12 produtos, uma estrutura razoável foi encontrada, com silhouette width de 0,68 e 0,60 respectivamente. Já para o grupo com 3 produtos, a estrutura identificada é fraca e pode ser artificial, pois seu silhouette width é 0,30. De maneira geral, obtemos uma silhouette width de 0.82, que é um número bastante satisfatório.

Para finalizar a parte de agrupamento, vamos anexar a coluna membership ao dataframe “cluster”, que vai dizer a que grupo cada coluna pertence, colocando os números de 1 a 4 que correspondem aos grupos 1 a 4.

```
cluster <- cbind(cluster, membership = partition$cluster)
head(cluster)
```

```
##                               Code Sum of quantity Sum of price
## 1 c27a276b623c751a6b0a8ad6243d681d              47      4786.41
## 2 5bcebbc4f704cffaf2e6ccaf1d1fb5c7              76      6241.81
## 3 d709ff164dc53eb7b8470e84c2b60974             140     11094.43
## 4 a36f5dfa4f08cdfe64594061ba76f30e              85     11502.30
## 5 e6017ce062eb324ab446e9983afba369             138     11920.50
## 6 213005fe5d815c539812614f1f6a7768             195     16728.06
##   liquid_cost                          category category1 membership
```

```
## 1     46.7958 388128822cef4b4f102ae881e040a64b         1         4
## 2     37.8157 388128822cef4b4f102ae881e040a64b         1         4
## 3     37.6516 388128822cef4b4f102ae881e040a64b         1         4
## 4     60.0668 388128822cef4b4f102ae881e040a64b         1         4
## 5     42.5440 388128822cef4b4f102ae881e040a64b         1         4
## 6     41.6587 388128822cef4b4f102ae881e040a64b         1         4
```

Agora exportei esse arquivo com o código abaixo. O código está comentado pois o arquivo ClusterMembership.csv está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Clustering/ProductMembership.xlsx.

```
#write.csv(cluster,file="ProductMembership.csv")
```

O código esta em comentário pois o arquivo já está disponível.
O source code, além de estar disponível neste documento, também está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Clustering/clustering%20part.r

Previsão de venda para junho, julho e agosto de 2017

A partir dos dados fornecidos pelo desafio, o meu primeiro passo foi preparar um dataset em que na sua primeira coluna estivessem as datas de meses, enquanto que o resto das colunas seria composto por cada um dos 131 produtos ou "codes". As quantidades vendidas por cada produto por cada mês popularia esse dataset. Então agreguei todas as vendas por mês por produto através de uma tabela dinâmica no excel. Nomeei esse dataset de "TodosOsProdutos", que está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Forecast/TodosOsProdutos.xlsx. Verifiquei que, como já havia citado antes, o mês mais antigo que aparece nos dados fornecidos é junho/2016, enquanto que o mês mais recente é julho/2017. Porém, apenas trouxe os dados de junho/2016 até maio/2017, pois o target é prever junho, julho e agosto de 2017. Vamos ver as 2 primeiras colunas e as 12 primeiras linhas deste dataset:

```
library(readxl)
TodosOsProdutos <- read_excel("~/Desktop/PrevisaoTodosProdutos/TodosOsProdutos.xlsx")
TodosOsProdutos[1:12,1:2]
```

```
## # A tibble: 12 x 2
##    `Row Labels` `0671c2b9132a3f5215a4212ce0691694`
##           <dttm>                              <dbl>
##  1   2016-06-01                                 NA
##  2   2016-07-01                                 NA
##  3   2016-08-01                                 NA
##  4   2016-09-01                                 NA
##  5   2016-10-01                                 NA
##  6   2016-11-01                                717
##  7   2016-12-01                                249
##  8   2017-01-01                                252
##  9   2017-02-01                                668
## 10   2017-03-01                               1481
## 11   2017-04-01                                536
## 12   2017-05-01                                722
```

```
#View(TodosOsProdutos)
```

Percebi que, para alguns produtos, não havia dados para alguns meses. Ou seja, ou unidades não haviam sido pedidas para aquele mês, ou simplesmente esses dados não foram fornecidos. Fazer um modelo de previsão para cada produto seria impossível. Por isso, foi preciso pensar em uma maneira de fazer um modelo que fosse se auto-adaptar a cada série temporal, e seria imprescindível também que esse modelo fosse robusto o suficiente para lidar com células vazias.

Há dois pacotes famosos que conheço para séries temporais que poderiam ser utilizados: O "prophet", desenvolvido pelo Facebook, ou o "forecast", desenvolvido por Rob Hyndman, grande estatístico australiano especialista em séries temporais. Decidi utilizar o "forecast", pois tinha um pouco mais de familiaridade. Assim, vamos carregar essa library.

```
library(forecast)
```

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

Tendo o dataset "TodosOsProdutos" já carregado no R, o primeiro passo é transformar esse dataset na classe ts (timeseries). Tiramos a primeira coluna com as datas, selecionamos a frequência = 12, que correspondem a 12 meses, e selecionamos a start date como junho de 2016. Então verificamos as duas primeiras colunas novamente:

```
TodosOsProdutos1 <- ts(TodosOsProdutos[,-1],f=12,s=2016+6/12)
TodosOsProdutos1[1:12,1:2]
```

```
##       0671c2b9132a3f5215a4212ce0691694 09f544ec2a74c89abeec7b0590fc2d11
##  [1,]                               NA                               NA
##  [2,]                               NA                               NA
##  [3,]                               NA                               57
##  [4,]                               NA                              141
##  [5,]                               NA                               25
##  [6,]                              717                              196
##  [7,]                              249                              180
##  [8,]                              252                              128
##  [9,]                              668                                2
## [10,]                             1481                               50
## [11,]                              536                               62
## [12,]                              722                               97
```

```
#View(TodosOsProdutos1)
```

Se a frequência fosse maior que 24, deveríamos usar a função tbats(), que é o modelo "Exponential smoothing state space model with Box-Cox transformation, ARMA errors, Trend and Seasonal components" com performance um tanto lenta. Como os dados fornecidos são de baixa frequência (f=12), segundo o próprio Rob Hyndman (https://robjhyndman.com/hyndsight/forecast6/), é recomendado que se use o modelo ets() "error, trend and seasonality" ou "Exponential Smoothing State Space Model". Podemos aplicar diretamente a função forecast() para cada time series, já que o parâmetro "model" usa o modelo "ets" como padrão.

Antes, guardamos o número de colunas na variável ncols, e definimos quantos períodos futuros queremos prever determinando o valor de "h".

```
ncols <- ncol(TodosOsProdutos1)
ncols
```

```
## [1] 131
```

```
h <- 3 #períodos futuros para o forecast
```

Criamos uma matriz 3x131 para receber todos os valores das 393 previsões.

```
fcastMean <- matrix(NA,nrow=h,ncol=ncols)
```

Então fazemos um for loop, passando pela time series de cada produto em suas respectivas colunas, fazemos a previsão e vemos as primeiras 3 previsões.

```
fcastMean <- matrix(NA,nrow=h,ncol=ncols)
  for (i in 1:ncols) {
    fcastMean[,i] <- forecast(TodosOsProdutos1[,i],h=h)$mean
```

```
  }
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
```

```
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
```

```
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
fcastMean[1:3,1:3] #Vendo as primeiras previsões
```

```
##          [,1]     [,2]     [,3]
## [1,] 660.7836 93.79542 72.79111
## [2,] 660.7836 93.79542 72.79111
## [3,] 660.7836 93.79542 72.79111
```

Adicionanamos a coluna data:

```
fcastMean1 <- cbind(c('06/2017', '07/2017', '08/2017'), fcastMean)
fcastMean1[,1:4]
```

```
##      [,1]      [,2]               [,3]               [,4]
## [1,] "06/2017" "660.783564803514" "93.7954157832027" "72.791110136131"
## [2,] "07/2017" "660.783564803514" "93.7954157832027" "72.791110136131"
## [3,] "08/2017" "660.783564803514" "93.7954157832027" "72.791110136131"
```

Guardamos o nome de todos os produtos em um vetor "produtos"

```
produtos <- colnames(TodosOsProdutos[2:ncol(TodosOsProdutos)])
head(produtos)
```

```
## [1] "0671c2b9132a3f5215a4212ce0691694" "09f544ec2a74c89abeec7b0590fc2d11"
## [3] "0ad316f6b5cb5e81ebff73ae2490ccfe" "0bbe09e34a11e8e31cf49d6f8df2992d"
## [5] "0dca7ec6ba9b6e8f17f04f713a6be727" "0f38be2df6854b4374f06cae1bc38482"
```

Adicionamos uma linha em cima colocando o nome da cada produto na coluna das suas previsões.

```
fcastMean2 <- rbind(union("PredictionDates", produtos), fcastMean1)
fcastMean2[,1:4]
```

```
##      [,1]              [,2]
## [1,] "PredictionDates" "0671c2b9132a3f5215a4212ce0691694"
## [2,] "06/2017"         "660.783564803514"
```

```
## [3,] "07/2017"         "660.783564803514"
## [4,] "08/2017"         "660.783564803514"
##      [,3]                               [,4]
## [1,] "09f544ec2a74c89abeec7b0590fc2d11" "0ad316f6b5cb5e81ebff73ae2490ccfe"
## [2,] "93.7954157832027"                 "72.791110136131"
## [3,] "93.7954157832027"                 "72.791110136131"
## [4,] "93.7954157832027"                 "72.791110136131"
```

Exportamos esse arquivo, vide código abaixo. O código está comentado, pois arquivo já está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Forecast/PrevisaoJunhoJulhoAgosto2017.xlsx

```
#write(t(fcastMean2),file="Desktop/PrevisaoTodosProdutos/PrevisaoJunhoJulhoAgosto2017.csv",sep=","
#,ncol=ncol(fcastMean2))
```

Agora podemos verificar qual modelo foi usado em cada série temporal através da construção da seguinte matriz

```
fcastMethod <- matrix(NA,nrow=h,ncol=ncols)
  for (i in 1:ncols) {
   fcastMethod[,i] <- forecast(TodosOsProdutos1[,i],h=h)$method
  }
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
```

```
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
```

```
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```
## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
```

```
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series

## Warning in ets(object, lambda = lambda, allow.multiplicative.trend =
## allow.multiplicative.trend, : Missing values encountered. Using longest
## contiguous portion of time series
```

```r
fcastMethod [1:3,1:3]
```

```
##      [,1]         [,2]         [,3]
## [1,] "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
## [2,] "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
## [3,] "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
```

Adicionamos a coluna data

```r
fcastMethod1 <- cbind(c('06/2017', '07/2017', '08/2017'), fcastMethod)
fcastMethod1[,1:4]
```

```
##      [,1]      [,2]         [,3]         [,4]
## [1,] "06/2017" "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
```

```
## [2,] "07/2017" "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
## [3,] "08/2017" "ETS(A,N,N)" "ETS(M,N,N)" "ETS(M,N,N)"
```

Adicionamos uma linha em cima colocando o nome da cada produto na coluna das suas previsões e verificamos cada modelo utilizado para cada série temporal.

```
fcastMethod2 <- rbind(union("PredictionDates", produtos), fcastMethod1)
fcastMethod2
```

```
##      [,1]              [,2]
## [1,] "PredictionDates" "0671c2b9132a3f5215a4212ce0691694"
## [2,] "06/2017"         "ETS(A,N,N)"
## [3,] "07/2017"         "ETS(A,N,N)"
## [4,] "08/2017"         "ETS(A,N,N)"
##      [,3]                                [,4]
## [1,] "09f544ec2a74c89abeec7b0590fc2d11" "0ad316f6b5cb5e81ebff73ae2490ccfe"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,5]                                [,6]
## [1,] "0bbe09e34a11e8e31cf49d6f8df2992d" "0dca7ec6ba9b6e8f17f04f713a6be727"
## [2,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(A,N,N)"
##      [,7]                                [,8]
## [1,] "0f38be2df6854b4374f06cae1bc38482" "118484c270085e811fbbc81978a269b2"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,9]                                [,10]
## [1,] "13b69fd4bf80b95756e3b138c9169a7f" "174ef25d9556d516a813e2972f3b8351"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,11]                               [,12]
## [1,] "193628b6634713730d3c506f2da0ff58" "1a225367d52023424b6f4b2aa8632615"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,13]                               [,14]
## [1,] "1b3980ee40dc5d60ecae3b19cd41f49a" "1c234775cae774823f38abe6721e61a4"
## [2,] "ETS(M,A,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,A,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,A,N)"                       "ETS(M,N,N)"
##      [,15]                               [,16]
## [1,] "1c619f92929dbaa41df54608dc70a7ce" "1dbe25b2fd344aed0c444fd6b715525b"
## [2,] "ETS(A,A,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,A,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,A,N)"                       "ETS(M,N,N)"
##      [,17]                               [,18]
## [1,] "1f12f1e1b9e7a20d4ad9dd549ec072c0" "213005fe5d815c539812614f1f6a7768"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,19]                               [,20]
```

```
## [1,] "23056d85a1c9115cd021a6ad35c84aff" "24549e47832f72f647d40a86b43b6925"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,21]                              [,22]
## [1,] "270dcdb08fc6470a6976f43b8169b0f6" "29424aaf6e27a8dbe4b7273a0a39131d"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,23]                              [,24]
## [1,] "29f175b6bccc264e8141481dcda8102d" "2ab0e87dbce6ac45502aa1d2a8265933"
## [2,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(A,N,N)"
##      [,25]                              [,26]
## [1,] "2bc9e1d807d8f9187b8650acd35a14e5" "2e35421c34fb588ba40a0c57b3971d24"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,27]                              [,28]
## [1,] "30b6a9b1621f558d788ffb3d07a19281" "32ceebf3efea1d04ace4183d20d4da5b"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,29]                              [,30]
## [1,] "3454ea52396a4cfd3fc37414d30c7b9c" "3657af9de7395eaba0dbcbcaa6fd90be"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,31]                              [,32]
## [1,] "373cc1cfc10a45488be6b97bd5e94c44" "374e1947dcb8f4848f4ada6f04921edd"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,33]                              [,34]
## [1,] "37a62c0ad48679cee5554655de294721" "37e2a39e829495d13b6cbb5320413e48"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,35]                              [,36]
## [1,] "390943ce05959ac98c702d250c2ebb54" "3a36dc63a58442085d0ccd98c4f9c64a"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,37]                              [,38]
## [1,] "3b4407288e2983a514a241c9b84b7094" "3bc993e0f0c636e9aaaefa0356bdecc0"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,39]                              [,40]
## [1,] "3d21b63892749e921e3ff5818753bd67" "3d3d13446c52ecaaed5d0bf55a933d4d"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
```

```
##      [,41]                              [,42]
## [1,] "3da22f1b88a20ea8efc3d83fcb872e21" "40bddb00475d65eddb68e9aeb6fab0de"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,43]                              [,44]
## [1,] "40d98a2375332cb635d4cb28ab68e087" "411e1404e183718207628232e91ce5a9"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,45]                              [,46]
## [1,] "41e2bee39c1d3ef52fcedd69d0ab8c8c" "42920a6f15855662ee9a272fbe44cbd1"
## [2,] "ETS(A,N,N)"                       "ETS(M,A,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,A,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,A,N)"
##      [,47]                              [,48]
## [1,] "4534ea61b50410b3b6243e02b40c8cd1" "453d427550d3816e446d4882bf67a75f"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,49]                              [,50]
## [1,] "4557c7e5af70efd2e3ca2befd59ccdc3" "498556cd1dade09f21fea97d3c916875"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,51]                              [,52]
## [1,] "49d1447a5d1a218169eef2ea58cdc149" "4b58730cdc153f4eeda0d1321e630ba4"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,53]                              [,54]
## [1,] "4ceedf57303e127d31a164c7ae5791d8" "4ef261b089aa567ef24778fae254bfe2"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,55]                              [,56]
## [1,] "51016cd454b391dd0b0e23b2b16b6fc3" "525848b647262de5fd7be193b17cdaac"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,57]                              [,58]
## [1,] "54209126056016c7c391c0c8fd8e6eff" "547bbdaa191bdb83d5b5b376bf2402fb"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,59]                              [,60]
## [1,] "5490e2abafb28c022e53b55ba6641122" "55447a73ff140176f4210347854c71f1"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,61]                              [,62]
## [1,] "5837d4e32e5382af65d33b4dfbd9f561" "585eead66aaf3b2e140d480979184ae6"
## [2,] "ETS(M,N,N)"                       "ETS(A,A,N)"
## [3,] "ETS(M,N,N)"                       "ETS(A,A,N)"
```

```
## [4,] "ETS(M,N,N)"                       "ETS(A,A,N)"
##      [,63]                              [,64]
## [1,] "59af761735fbd646f8a8f98dc88f6fab" "5b7a30a9e6a43b170ad4d9e00d8d9359"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,65]                              [,66]
## [1,] "5bcebbc4f704cffaf2e6ccaf1d1fb5c7" "5e39201e582b1bb89cae7f650e4330c8"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,67]                              [,68]
## [1,] "5e4c14883e5a606fac56b2630da5dca7" "5fb59bee157577ee04a269e52af88598"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,69]                              [,70]
## [1,] "5ff2d52d31bf9ad42924a35c4729ffdb" "600c486af5e3f2d1624d2fc872e20e16"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,71]                              [,72]
## [1,] "60424117a2618c7184687046fa5693c4" "618ea23457368cf423b03d6b8fede992"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,73]                              [,74]
## [1,] "630b051b117d509fc00a7c72328b1ddf" "6411de56554a998e78ae1317fc956e19"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,75]                              [,76]
## [1,] "6ace63708a91744bea8a68311bb3506c" "6c82ad0e791258434fd42c51409b0239"
## [2,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(A,N,N)"
##      [,77]                              [,78]
## [1,] "6e6ae6bc648c6a00b4a4b7a8e976a41a" "7162b3182cd93dd197257b0339627554"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,79]                              [,80]
## [1,] "723f73c85e91fc31d147dfade389d4f9" "727673fa3e457bc596532b3eb26b23a0"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,81]                              [,82]
## [1,] "75caad099f6f8c205e22e93826732c22" "760693745e10b0c5e68c42214c729b0d"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,83]                              [,84]
## [1,] "7c829a5a8a0e4408b682f4394783483e" "7da116bd1d42f3475803402e710253cf"
## [2,] "ETS(M,N,N)"                       "ETS(M,N,N)"
```

```
## [3,] "ETS(M,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(M,N,N)"
##      [,85]                              [,86]
## [1,] "7e3713530b46887cff58a2e2ac433ac5" "8bdd60f700ed8368e66080eb6bb6d313"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,87]                              [,88]
## [1,] "8ca06ab9bced028b2f180d4fa0527588" "8e5d2c3f2476cb5c507dd0f00f6eabda"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,89]                              [,90]
## [1,] "9cdf9e07e226869d8da4a3e10ac65c62" "9e5dd3c1d252136c4351b84589dae2b5"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,91]                              [,92]
## [1,] "a0c96bb1b4117eeeae27b77a1381f55f" "a2018dae10d736a66eea5a0a349ef9ee"
## [2,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                       "ETS(A,N,N)"
##      [,93]                              [,94]
## [1,] "a36f5dfa4f08cdfe64594061ba76f30e" "a586c072b4a3958ac72a6fff71730cb3"
## [2,] "ETS(A,A,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,A,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,A,N)"                       "ETS(M,N,N)"
##      [,95]                              [,96]
## [1,] "a7772a34be22f0fd09f0ef36b6eb337f" "abf2d3cb446492ee7897087db9a0b2a0"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,97]                              [,98]
## [1,] "ae5a740ca930b9149f590179b0dde3f3" "b08b7321c4db8f45a1a97a79d1e44dd8"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,99]                              [,100]
## [1,] "b272ba3f4adb1dd16eaac1b53940629e" "b367e71967b4701550ffce69e729ee6f"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,101]                             [,102]
## [1,] "b3fca69c356bec270dc3a90f8b498883" "b763aecfcfc8cbac2c3a51488d33480a"
## [2,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(A,N,N)"
##      [,103]                             [,104]
## [1,] "b9d929195dcd4e6a36e5e65891746b5e" "bc4d7296a37f5dfa0944b3274229b0a0"
## [2,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                       "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                       "ETS(M,N,N)"
##      [,105]                             [,106]
## [1,] "bc97d7cb7c13c2a7ff4558bb12a3f047" "c254dc11afbcc091678f0ab49a02e7ad"
```

```
## [2,] "ETS(A,A,N)"                      "ETS(M,N,N)"
## [3,] "ETS(A,A,N)"                      "ETS(M,N,N)"
## [4,] "ETS(A,A,N)"                      "ETS(M,N,N)"
##      [,107]                            [,108]
## [1,] "c27a276b623c751a6b0a8ad6243d681d" "c32ada18ec4f2992e8c003ffeb73b97b"
## [2,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(A,N,N)"
##      [,109]                            [,110]
## [1,] "c348d9bc6a152d8d34489a79ab9452a7" "c443d252c048280160fc427766d9f1f4"
## [2,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(A,N,N)"
##      [,111]                            [,112]
## [1,] "c80a47e4c53432d9ee8cd98c9ce13769" "c85bd735a203c905716bc8fba284d02a"
## [2,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(A,N,N)"
##      [,113]                            [,114]
## [1,] "c85f192a81fa83780e5c012175749eee" "cbeb3a98c1c9b01522db6ee2128ed805"
## [2,] "ETS(M,A,N)"                      "ETS(A,N,N)"
## [3,] "ETS(M,A,N)"                      "ETS(A,N,N)"
## [4,] "ETS(M,A,N)"                      "ETS(A,N,N)"
##      [,115]                            [,116]
## [1,] "ce4baabfbcb1d43e22f7ba44b49f2714" "d32bc6c4069b86ba9e9d7d651dbf1a1e"
## [2,] "ETS(A,N,N)"                      "ETS(M,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(M,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(M,N,N)"
##      [,117]                            [,118]
## [1,] "d408e1b5e841dde4e15a4cfa182e3812" "d4592ab52cb9cd5af0510943a4c8e28c"
## [2,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(A,N,N)"
##      [,119]                            [,120]
## [1,] "d57911cca4b08f7b46417d952c0ca1dc" "d5bc9e14d090330cd07e6ccbcb3c3e4e"
## [2,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(A,N,N)"
##      [,121]                            [,122]
## [1,] "d709ff164dc53eb7b8470e84c2b60974" "dd1935ffd0ee2b6ec159ba7867d11e57"
## [2,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(M,N,N)"
##      [,123]                            [,124]
## [1,] "e0b4fdba39475c05a3ac9e769cce9bfa" "e13f7f001fe2b1af072a3d50d3058284"
## [2,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(A,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(A,N,N)"                      "ETS(A,N,N)"
##      [,125]                            [,126]
## [1,] "e6017ce062eb324ab446e9983afba369" "e6762ba2ffbca07ab6cee7551caeaad5"
## [2,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(A,N,N)"
##      [,127]                            [,128]
```

```
## [1,] "ee2b4e97025f5ca840570265c4288c99" "f08984b2adcbf33ba61fe13fcfa5b957"
## [2,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(A,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(A,N,N)"
##      [,129]                            [,130]
## [1,] "f5f92c2a12f182115c45288a6ef28e94" "f9a023f31c8087fd0c169b3bedd351d1"
## [2,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(M,N,N)"
##      [,131]                            [,132]
## [1,] "fd84644da59504bd9e9dcb4b6db63bea" "ffdad3ddbaf6c76c9bba1b48c51e03c6"
## [2,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [3,] "ETS(M,N,N)"                      "ETS(M,N,N)"
## [4,] "ETS(M,N,N)"                      "ETS(M,N,N)"
```

Verificamos que há 4 variações de modelos utilizados: ETS(A,N,N), ETS(M,N,N), ETS(A,A,N) e ETS(MAN). Os valores passados para a função são em ordem: erro, tendência e sazonalidade. "A" siginifica additive, N significa none e M significa multiplicative.

Ao tentar utilizar o for loop para criação de novas matrizes com upper limit, lower limit, level(intervalo de confiança), residuals e fitted, um erro é gerado. Isso ocorre pelo fato de haver valores vazios nas matrizes iniciais.

O source code desta parte de previsão, além de estar disponível neste documento, também está disponível neste link: https://github.com/hlispector/Desafio/blob/master/Forecast/PrevisaoTodosProdutos.r

Conclusões

Referente à análise de cluster, acredito que mais colunas que descrevessem características específicas dos produtos facilitiariam na hora de fazer uma diferenciação e agrupamento de produtos mais refinado. As informações de impostos, por exemplo, não foram ricas para poder levar em consideração no agrupamento. Entretanto, os resultados me pareceram bastante razoáveis e creio que podem sim ajudar a ter uma ideia sobre os produtos.

Em relação a time series, caso dados de mais meses fossem fornecidos, aumentariam as possibilidades de as previsões serem mais próximas dos números reais. Com um histórico maior e sem tantos campos vazios, também seria possível dividir o dataset em training set e test set e calcular indicadores de erro como MAPE (Mean Absolute Percentage Error) onde se compara o valor real com o valor previsto. De toda maneira, ser capaz de prever números em séries temporais para 131 produtos com alguns dados ausentes é bastante impressionante. As previsões feitas fazem sentido e ficaria curioso em comparar com os dados reais e calcular os erros.

Agradeço a oportunidade de ter trabalhado neste problema interessante que certamente agregou à minha aprendizagem nessa carreira em Data Science.