Anomaly Detection

Dave Njoroge

2022-06-16

Importing the libraries and previewing the data

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(anomalize)
```

```
## == Use anomalize to improve your Forecasts by 50%! ============================
## Business Science offers a 1-hour course - Lab #18: Time Series Anomaly Detection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(tibbletime)
```

```
##
## Attaching package: 'tibbletime'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(dplyr)
library(magrittr)
```

```
##
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
##
##      set_names


## The following object is masked from 'package:tidyr':
##
##      extract
```

```r
path = 'http://bit.ly/CarreFourSalesDataset'
```

```r
# Grouping by server and converting to tibbletime
s = read.csv(path) %>% group_by(Date) %>%
   mutate(Date = as.Date (Date, format="%m/%d/%Y")) %>% as_tbl_time(index = Date)
head(s)
```

```
## # A time tibble: 6 x 2
## # Index: Date
##   Date        Sales
##   <date>      <dbl>
## 1 2019-01-05 549.
## 2 2019-03-08  80.2
## 3 2019-03-03 341.
## 4 2019-01-27 489.
## 5 2019-02-08 634.
## 6 2019-03-25 628.
```

```r
class(s)
```

```
## [1] "grouped_tbl_time" "tbl_time"          "grouped_df"        "tbl_df"
## [5] "tbl"               "data.frame"
```

```r
str(s)
```

```
## grouped_tbl_time [1,000 x 2] (S3: grouped_tbl_time/tbl_time/grouped_df/tbl_df/tbl/data.frame)
##  $ Date : Date[1:1000], format: "2019-01-05" "2019-03-08" ...
##  $ Sales: num [1:1000] 549 80.2 340.5 489 634.4 ...
##  - attr(*, "groups")= tibble [89 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ Date : Date[1:89], format: "2019-01-01" "2019-01-02" ...
##   ..$ .rows: list<int> [1:89]
##   .. ..$ : int [1:12] 18 246 451 485 497 524 568 697 830 840 ...
##   .. ..$ : int [1:8] 89 301 411 426 452 559 745 838
##   .. ..$ : int [1:8] 171 206 272 375 542 778 935 988
##   .. ..$ : int [1:6] 157 207 214 270 408 413
##   .. ..$ : int [1:12] 1 107 244 269 377 502 512 537 545 767 ...
##   .. ..$ : int [1:9] 71 255 283 366 385 698 704 870 956
##   .. ..$ : int [1:9] 36 68 249 273 276 277 356 548 581
##   .. ..$ : int [1:18] 128 129 268 417 444 483 507 605 686 687 ...
##   .. ..$ : int [1:8] 77 358 683 833 837 952 959 985
##   .. ..$ : int [1:9] 9 33 93 298 318 569 924 926 957
##   .. ..$ : int [1:8] 226 240 241 381 407 680 724 758
##   .. ..$ : int [1:11] 78 106 122 419 449 547 624 700 739 801 ...
```

```
##     .. ..$ : int [1:10] 76 196 433 528 574 617 886 928 954 968
##     .. ..$ : int [1:13] 203 250 376 383 578 584 671 682 742 789 ...
##     .. ..$ : int [1:13] 16 38 192 441 443 445 633 636 651 750 ...
##     .. ..$ : int [1:10] 118 167 212 511 527 649 662 688 695 807
##     .. ..$ : int [1:11] 41 325 520 566 610 623 670 710 775 790 ...
##     .. ..$ : int [1:9] 137 175 322 646 658 720 722 874 897
##     .. ..$ : int [1:16] 114 161 169 174 300 406 585 614 631 718 ...
##     .. ..$ : int [1:10] 59 120 236 348 437 556 709 763 765 934
##     .. ..$ : int [1:8] 19 81 145 294 364 427 499 708
##     .. ..$ : int [1:7] 75 80 418 518 541 652 776
##     .. ..$ : int [1:17] 83 91 141 182 248 422 471 550 601 609 ...
##     .. ..$ : int [1:13] 69 291 396 532 613 685 734 754 797 815 ...
##     .. ..$ : int [1:17] 29 54 131 142 184 211 289 299 329 372 ...
##     .. ..$ : int [1:17] 82 92 172 194 227 231 352 392 549 599 ...
##     .. ..$ : int [1:14] 4 127 274 369 389 459 498 503 513 525 ...
##     .. ..$ : int [1:14] 32 155 180 190 237 410 446 506 544 827 ...
##     .. ..$ : int [1:12] 153 201 256 374 480 508 572 659 696 738 ...
##     .. ..$ : int [1:9] 164 344 380 489 558 563 748 846 848
##     .. ..$ : int [1:14] 221 228 257 288 311 386 431 455 707 715 ...
##     .. ..$ : int [1:6] 103 115 170 313 540 582
##     .. ..$ : int [1:14] 42 70 119 245 251 278 371 473 481 500 ...
##     .. ..$ : int [1:14] 51 63 290 333 461 463 501 567 615 653 ...
##     .. ..$ : int [1:11] 347 403 420 490 557 588 666 714 725 786 ...
##     .. ..$ : int [1:12] 111 112 143 258 312 340 553 594 783 887 ...
##     .. ..$ : int [1:13] 11 35 48 95 204 379 523 533 716 726 ...
##     .. ..$ : int [1:20] 14 58 154 260 331 440 458 529 536 593 ...
##     .. ..$ : int [1:12] 5 27 43 130 168 188 423 467 606 733 ...
##     .. ..$ : int [1:13] 98 181 247 264 317 469 510 551 900 902 ...
##     .. ..$ : int [1:11] 49 52 67 133 147 189 315 341 607 694 ...
##     .. ..$ : int [1:8] 72 191 233 562 761 782 800 974
##     .. ..$ : int [1:8] 13 219 484 531 664 681 743 877
##     .. ..$ : int [1:8] 178 259 393 456 543 583 768 844
##     .. ..$ : int [1:8] 65 121 316 345 447 825 834 905
##     .. ..$ : int [1:19] 61 113 205 281 323 343 351 391 457 470 ...
##     .. ..$ : int [1:8] 139 405 625 645 655 669 856 871
##     .. ..$ : int [1:13] 24 134 200 223 308 332 361 530 598 719 ...
##     .. ..$ : int [1:7] 136 138 293 320 987 995 1000
##     .. ..$ : int [1:9] 365 677 699 713 781 787 852 944 960
##     .. ..$ : int [1:10] 10 156 165 286 388 577 667 744 861 883
##     .. ..$ : int [1:6] 218 224 234 665 891 951
##     .. ..$ : int [1:11] 144 378 404 435 552 753 798 919 923 994 ...
##     .. ..$ : int [1:8] 84 305 460 515 641 668 678 805
##     .. ..$ : int [1:9] 8 62 338 650 676 785 812 822 950
##     .. ..$ : int [1:16] 7 21 31 162 166 303 476 505 526 539 ...
##     .. ..$ : int [1:9] 229 265 275 321 495 660 771 795 978
##     .. ..$ : int [1:14] 47 90 346 350 362 429 640 657 684 703 ...
##     .. ..$ : int [1:6] 56 243 487 711 732 842
##     .. ..$ : int [1:10] 102 359 384 519 576 580 596 675 731 735
##     .. ..$ : int [1:18] 25 116 210 239 253 354 466 570 673 705 ...
##     .. ..$ : int [1:14] 3 40 110 173 177 187 296 454 462 591 ...
##     .. ..$ : int [1:12] 44 185 284 355 400 409 590 693 804 824 ...
##     .. ..$ : int [1:17] 22 73 79 86 100 117 186 238 310 342 ...
##     .. ..$ : int [1:11] 64 132 149 222 271 282 370 395 654 894 ...
##     .. ..$ : int [1:9] 55 158 262 314 363 478 538 554 858
```

```
##    .. ..$ : int [1:11] 2 96 135 146 324 357 430 432 438 439 ...
##    .. ..$ : int [1:16] 12 46 74 85 123 125 235 327 394 425 ...
##    .. ..$ : int [1:12] 28 37 126 220 390 414 448 573 736 921 ...
##    .. ..$ : int [1:11] 17 20 152 215 230 242 319 387 535 918 ...
##    .. ..$ : int [1:12] 60 94 199 232 266 285 302 326 328 402 ...
##    .. ..$ : int [1:10] 66 124 163 263 295 397 491 689 872 889
##    .. ..$ : int [1:18] 267 292 334 353 367 373 514 561 589 603 ...
##    .. ..$ : int [1:12] 23 30 34 202 398 492 595 796 799 811 ...
##    .. ..$ : int [1:9] 45 140 193 254 360 453 479 729 769
##    .. ..$ : int [1:6] 252 464 604 806 863 890
##    .. ..$ : int [1:7] 208 304 477 516 634 723 961
##    .. ..$ : int [1:16] 50 105 148 195 261 337 382 424 436 546 ...
##    .. ..$ : int [1:15] 213 225 279 280 428 496 522 564 565 616 ...
##    .. ..$ : int [1:6] 176 309 336 486 879 940
##    .. ..$ : int [1:10] 26 53 108 216 415 442 468 575 814 991
##    .. ..$ : int [1:11] 39 99 151 179 183 198 488 737 741 803 ...
##    .. ..$ : int [1:11] 109 217 287 335 368 600 663 829 906 908 ...
##    .. ..$ : int [1:9] 6 87 330 401 504 555 760 821 969
##    .. ..$ : int [1:13] 101 197 306 450 521 534 579 749 764 828 ...
##    .. ..$ : int [1:10] 57 88 150 160 587 643 702 909 911 942
##    .. ..$ : int [1:10] 104 209 339 349 421 465 608 621 717 945
##    .. ..$ : int [1:8] 15 97 297 509 517 560 706 989
##    .. ..$ : int [1:11] 159 307 474 475 644 647 672 882 884 938 ...
##    .. ..@ ptype: int(0)
##    ..- attr(*, ".drop")= logi TRUE
##  - attr(*, "index_quo")= language ~Date
##    ..- attr(*, ".Environment")=<environment: 0x000001fca173db70>
##  - attr(*, "index_time_zone")= chr "UTC"
```

```r
#Cheking for null values in our sales column
colSums(is.na(s))
```

```
##  Date Sales
##     0     0
```

```r
# Anomaly Detection
#s_anomalized = sales %>% summarise(Orders = sum(s)) %>%
#time_decompose(Orders) %>%
#    anomalize(remainder, alpha = 0.25, max_anoms = 0.2) %>%
#    time_recompose() %>%
#    plot_anomalies(time_recomposed = TRUE) +
#    ggtitle("alpha = 0.25, max_anoms = 0.2")


#
```

CONCLUSION

- Least anomalies occurred in the month of January.

- Most of the anomalies occurred in the months of February and March