Time Series Analysis

Apple Mobility Trends Data

Eldhose Poulose

24.04.2021

Introduction

Through this RMarkdown I want to create a deep understanding of the usage of R packages, and deepen my data analysis and visualisation skills.

Create an R Notebook which is then used as a Portfolio when applying for Data Science positions.

Steps

- Ingest Data
  - Apple Mobility Trends Data
- Merge datasets
- Exploratory Data Analysis and Visualizations
- Time Series Analysis and Forecasts
- Do "out of the box" time series forecast.
- Analyze fluctuations around time series trends.

Remark: The time series section is done for illustration purposes only. The forecasts there should not be taken seriously.

Motivation

Overall Process

Dataset

Apple Mobility Trends Data

- Download the data The Apple's page provides the Dataset for this data analysis.
- Import the data and summarize it
- Transform the data into long form
- Partition the data into subsets that correspond to combinations of geographical regions and transportation types
- Make contingency matrices and corresponding heat-map plots
- Make NN graphs over the contingency matrices and plot communities
- Plot the corresponding Time Series

Ingest Data

```
get_apple_target <- function(cdn_url = "https://covid19-static.cdn-apple.com",
                             json_file = "covid19-mobility-data/current/v3/index.json") {
  tf <- tempfile(fileext = ".json")
  curl::curl_download(paste0(cdn_url, "/", json_file), tf)
  json_data <- jsonlite::fromJSON(tf)
  paste0(cdn_url, json_data$basePath, json_data$regions$`en-us`$csvPath)
}
get_apple_target()
```

```
## [1] "https://covid19-static.cdn-apple.com/covid19-mobility-data/2106HotfixDev20/v3/en-us/applemobili
```

```
# get_apple_data <- function(url = get_apple_target(),
#                            fname = "applemobilitytrends-",
#                            date = stringr::str_extract(get_apple_target(), "\\d{4}-\\d{2}-\\d{2}"),
#                            ext = "csv",
#                            dest = "/Users/pouloeld/Documents/R_Projects/Time-Series-Analysis/AppleM
#                            save_file = c("n", "y")) {
#
#   save_file <- match.arg(save_file)
#   message("target: ", url)
#
#   destination <- paste0(dest,"/",fname,date,".csv")
#
#   tf <- tempfile(fileext = ext)
#   curl::curl_download(url, tf)
#
#   ## We don't save the file by default
#   switch(save_file,
#          y = fs::file_copy(tf, destination),
#          n = NULL)
#
#   read.csv(tf)
# }
```

```
#appleMobility <- get_apple_data()
```

```
fileName <- paste0("applemobilitytrends-",as.String(Sys.Date()-1),".csv")
# setwd("~/Documents/R_Projects/Time-Series-Analysis/AppleMobilityTrends")
# url <- paste0("https://covid19-static.cdn-apple.com/covid19-mobility-data/2106HotfixDev20/v3/en-us/ap
download.file(get_apple_target(),fileName)
appleMobility <- read.csv(fileName)
```

Observation: Reference date for normalization is January 13, 2020. Note the values in that column are set to 100.

Data Dimension

```
dim(appleMobility)
```

```
## [1] 4691  474
```

```
summary(as.data.frame(unclass(appleMobility[,1:3]), stringsAsFactors = TRUE))
```

```
##            geo_type                  region     transportation_type
##  city          : 790   Washington County:  27   driving:3048
##  country/region: 153   Jefferson County :  25   transit: 551
##  county        :2638   Montgomery County:  24   walking:1092
##  sub-region    :1110   Franklin County  :  22
##                        Madison County   :  21
##                        Jackson County   :  19
##                        (Other)          :4553
```

Answering basic questions about the Dataset

- Here I try to answer the basic questions which I come across from the data.
  - How many geo_types are present in the data?
  - How many unique regions are present in the data?
  - How many transportation modes are present in the data?
  - How many Countries are present in the data? **Note:** Big countries are divides into regions are provided them in the regions column.

```
lsGeo_type <- as.String(levels(appleMobility$geo_type))
lsGeo_type
```

```
## city
## country/region
## county
## sub-region
```

```
print(length(levels(appleMobility$region)))
```

```
## [1] 2325
```

```
#lsRegion <- as.String(levels(appleMobility$region))
#lsRegion
```

```
print(as.String(levels(appleMobility$transportation_type)))
```

```
## driving
## transit
## walking
```

```
print(length(levels(appleMobility$country)))
```

```
## [1] 48
```

```
#print(as.String(levels(appleMobility$country)))
```

Data transformation

Here I convert the data into narrow/long format. For this I use melt function from reshape2 package.

```
#appleMobility_melted <- melt(appleMobility,id=c("geo_type","region","transportation_type","alternative
colNames <- c("geo_type","region","transportation_type","alternative_name","sub.region","country")
appleMobility_melted <- tidyr::pivot_longer( data = appleMobility, cols = setdiff( names(appleMobility)
```

Remove empty rows

```
appleMobility_na <- appleMobility_melted[complete.cases(appleMobility_melted), ]
```

```
#colnames(appleMobility_na)[colnames(appleMobility_na) == "variable"] <- "Date"
```

Clean Date Format

```
appleMobility_na$Date <- gsub("[a-zA-Z ]", "", appleMobility_na$Date)
```

Add: DateInfo, Day

```
appleMobility_na$Date <- as.POSIXct(appleMobility_na$Date, format = "%Y.%m.%d", origin = "1970.01.01" )
#unique(appleMobility_na$Date) #To see the days
appleMobility_na$Day <- weekdays(appleMobility_na$Date)
appleMobility_na$Day <- as.factor(appleMobility_na$Day)
appleMobility_na$DateInfo <- format(appleMobility_na$Date, "%a %b %d %Y")
appleMobility_na$DateInfo <- as.factor(appleMobility_na$DateInfo)
```

Draw Random Samples

```
set.seed(123)
appleMobility_na %>% dplyr::sample_n(10)
```

```
## # A tibble: 10 x 10
##    geo_type region transportation_~ alternative_name sub.region country
##    <fct>    <fct>  <fct>            <fct>            <fct>      <fct>
##  1 county   Tangi~ driving          ""               "Louisian~ United~
##  2 city     Cardi~ walking          ""               "Wales"    United~
##  3 county   Monro~ driving          ""               "Wisconsi~ United~
##  4 county   River~ walking          ""               "Californ~ United~
##  5 sub-reg~ Cante~ driving          ""               ""         New Ze~
##  6 city     Tijua~ walking          ""               "Baja Cal~ Mexico
##  7 sub-reg~ Groni~ driving          ""               ""         Nether~
##  8 county   Deaf ~ driving          ""               "Texas"    United~
##  9 sub-reg~ Upper~ walking          "Oberösterreich" ""         Austria
## 10 county   Adams~ driving          ""               "Pennsylv~ United~
## # ... with 4 more variables: Date <dttm>, value <dbl>, Day <fct>,
## #   DateInfo <fct>
```

##Summary of the cleaned data

```
summary(as.data.frame(unclass(appleMobility_na), stringsAsFactors = TRUE))
```

```
##            geo_type                       region        transportation_type
##  city          : 367358   Washington County:  12565   driving:1399185
##  country/region:  71145   Jefferson County :  11637   transit: 256873
##  county        :1227766   Montgomery County:  11178   walking: 509254
##  sub-region    : 499043   Franklin County  :  10234
##                           Madison County   :   9771
##                           Jackson County   :   8841
##                           (Other)          :2101086
##                 alternative_name        sub.region                country
##                         :1691320             : 616231   United States:1443714
##  AB                     :   1399   Texas     : 112141   Japan        : 102953
##  ACT                    :   1399   California:  77298                :  71145
##  Andalucía              :   1399   Georgia   :  60957   France       :  41918
##  Bayern                 :   1399   Virginia  :  57237   Germany      :  40036
##  BC|Colombie-Britannique:   1399   Florida   :  56351   Thailand     :  31632
##  (Other)                : 466997   (Other)   :1185097   (Other)      : 433914
##       Date                          value                Day
##  Min.   :2020-01-13 00:00:00   Min.   :   0.44   Friday   :307032
##  1st Qu.:2020-05-08 00:00:00   1st Qu.:  85.62   Monday   :308098
##  Median :2020-09-03 00:00:00   Median : 114.69   Saturday :311684
##  Mean   :2020-09-02 11:44:01   Mean   : 121.61   Sunday   :307032
##  3rd Qu.:2020-12-28 00:00:00   3rd Qu.: 148.21   Thursday :311684
##  Max.   :2021-04-24 00:00:00   Max.   :2148.12   Tuesday  :308098
##                                                  Wednesday:311684
##             DateInfo
##  Fri Apr 02 2021:   4652
##  Fri Apr 03 2020:   4652
##  Fri Apr 09 2021:   4652
##  Fri Apr 10 2020:   4652
##  Fri Apr 16 2021:   4652
##  Fri Apr 17 2020:   4652
##  (Other)        :2137400
```

Data Partition

*I am interested in the geographical types and transportation modes. Therefore we group the data as per this requirement.

```
appleMobility_na %>%
  dplyr::group_by(geo_type, transportation_type) %>%
  dplyr::count()
```

```
## # A tibble: 12 x 3
## # Groups:   geo_type, transportation_type [12]
##    geo_type       transportation_type      n
##    <fct>          <fct>                <int>
##  1 city           driving             139035
##  2 city           transit              91609
##  3 city           walking             136714
##  4 country/region driving              29295
```

```
##  5 country/region transit              12555
##  6 country/region walking              29295
##  7 county         driving             971850
##  8 county         transit              70984
##  9 county         walking             184932
## 10 sub-region     driving             259005
## 11 sub-region     transit              81725
## 12 sub-region     walking             158313
```

```
appleMobility_part <- split(appleMobility_na, appleMobility_na[,c("geo_type","transportation_type")])
```

Heat-Map Plots

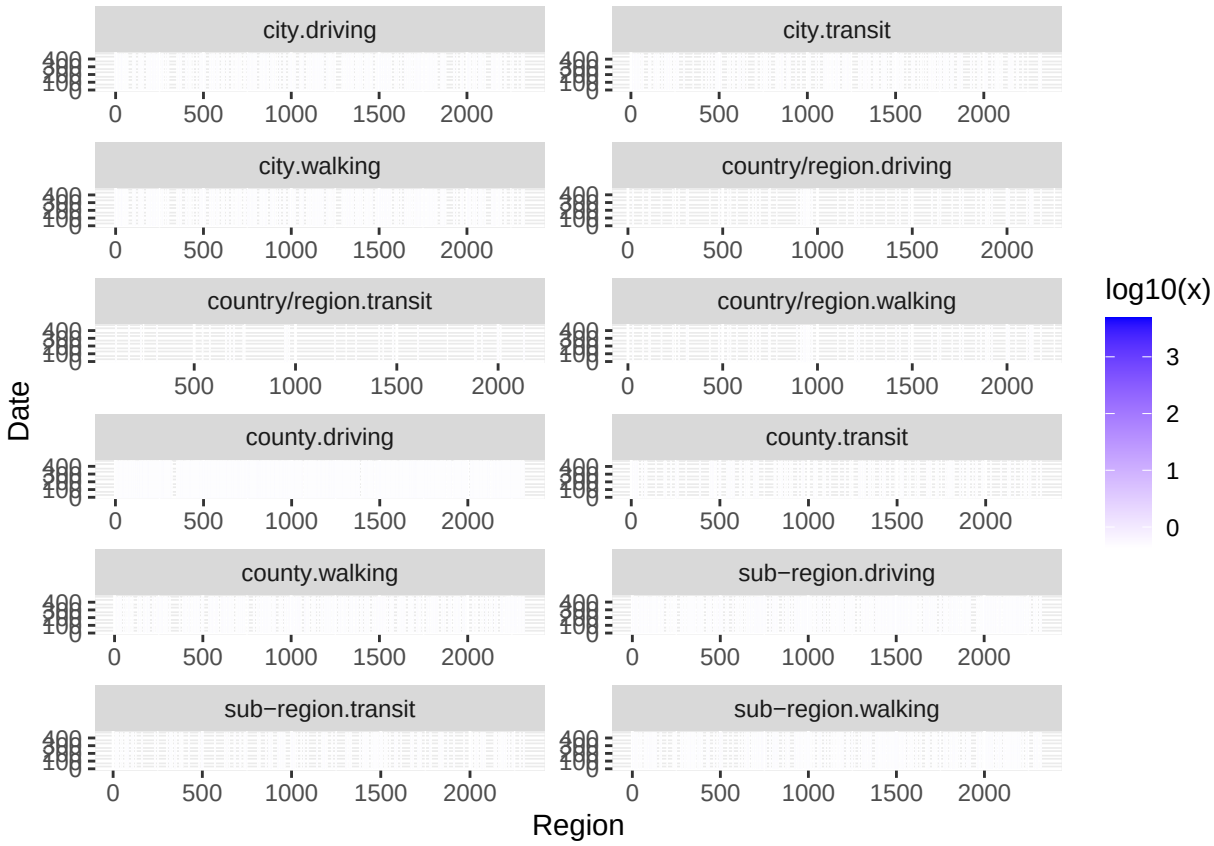
Contigency Matrix

```
aMatDateRegion <- purrr::map(appleMobility_part, function(dfX) { xtabs( formula = value ~ Date + region
aMatDateRegion <- aMatDateRegion[ purrr::map_lgl(aMatDateRegion, function(x) nrow(x) > 0 ) ]
```

```
sparseMatDateRegion <- purrr::map_df(aMatDateRegion, Matrix::summary, .id = "Type" )
head(sparseMatDateRegion)
```

```
## 465 x 2325 sparse Matrix of class "dgCMatrix", with 137175 entries
##           Type i j      x
## 1 city.driving 1 1 100.00
## 2 city.driving 2 1 100.73
## 3 city.driving 3 1 102.86
## 4 city.driving 4 1 102.65
## 5 city.driving 5 1 109.39
## 6 city.driving 6 1 109.62
```

```
ggplot2::ggplot(sparseMatDateRegion) +
  ggplot2::geom_tile( ggplot2::aes( x = j, y = i, fill = log10(x)), color = "white") +
  ggplot2::scale_fill_gradient(low = "white", high = "blue") +
  ggplot2::xlab("Region") + ggplot2::ylab("Date") +
  ggplot2::facet_wrap( ~Type, scales = "free", ncol = 2)
```




```
# d3heatmap::d3heatmap( x = aMatDateRegion[["country/region.driving"]], Rowv = FALSE )
```

```
# th <- 0.94
# aNNGraphs <-
#   purrr::map( aMatDateRegion, function(m) {
#     m2 <- cor(as.matrix(m))
#     for( i in 1:nrow(m2) ) {
#       m2[i,i] <- 0
#     }
#     m2 <- as( m2, "dgCMatrix")
#     #m2@x[ m2@x <= th ] <- 0
#     m2@x[ m2@x > th ] <- 1
#     igraph::graph_from_adjacency_matrix(Matrix::drop0(m2), weighted = TRUE, mode = "undirected")
#   })
```

```
# ind <- 3
# ceb <- cluster_edge_betweenness(aNNGraphs[[ind]])
# dendPlot(ceb, mode="hclust", main = names(aNNGraphs)[[ind]])
```

```
# plot(ceb, aNNGraphs[[ind]], vertex.size=1, vertex.label=NA, main = names(aNNGraphs)[[ind]])
```

Time Series Analysis

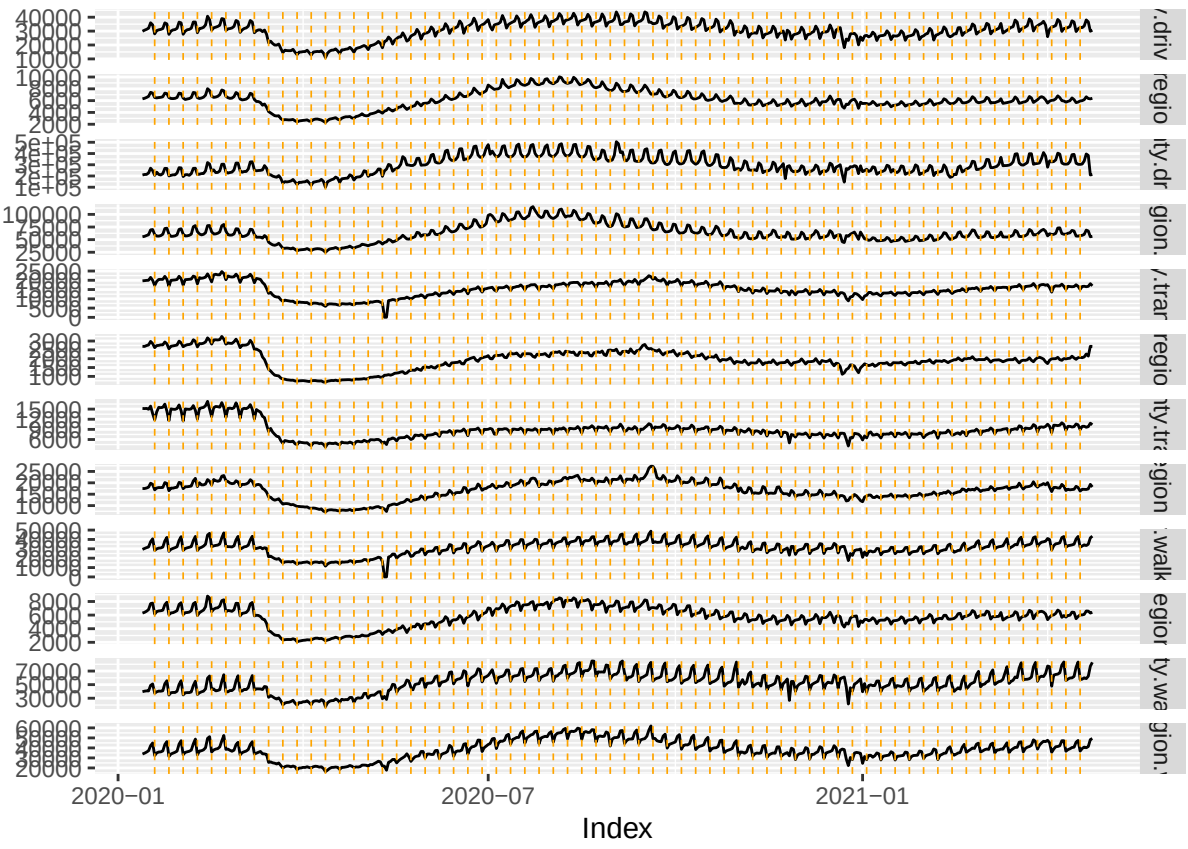
```
aDateStringToDateInfo <- unique(appleMobility_na[, c("Date", "DateInfo")] )
aDateStringToDateInfo <- setNames( aDateStringToDateInfo$DateInfo, aDateStringToDateInfo$Date )
aDateStringToDateInfo <- as.POSIXct(aDateStringToDateInfo, format = "%a %b %d %Y")
aTSDirReqByCountry <-  purrr::map( aMatDateRegion, function(m) rowSums(m) )
```

```
matTS <- do.call( cbind, aTSDirReqByCountry)
```

```
## Warning in (function (..., deparse.level = 1) : number of rows of result is not
## a multiple of vector length (arg 1)
```

```
zooObj <- zoo::zoo( x = matTS, as.POSIXct(rownames(matTS)) )
```

```
autoplot(zooObj) +
  aes(colour = NULL, linetype = NULL) +
    facet_grid(Series ~ ., scales = "free_y") +
  geom_vline( xintercept = aDateStringToDateInfo[weekdays(aDateStringToDateInfo) == "Sunday"], color =
```



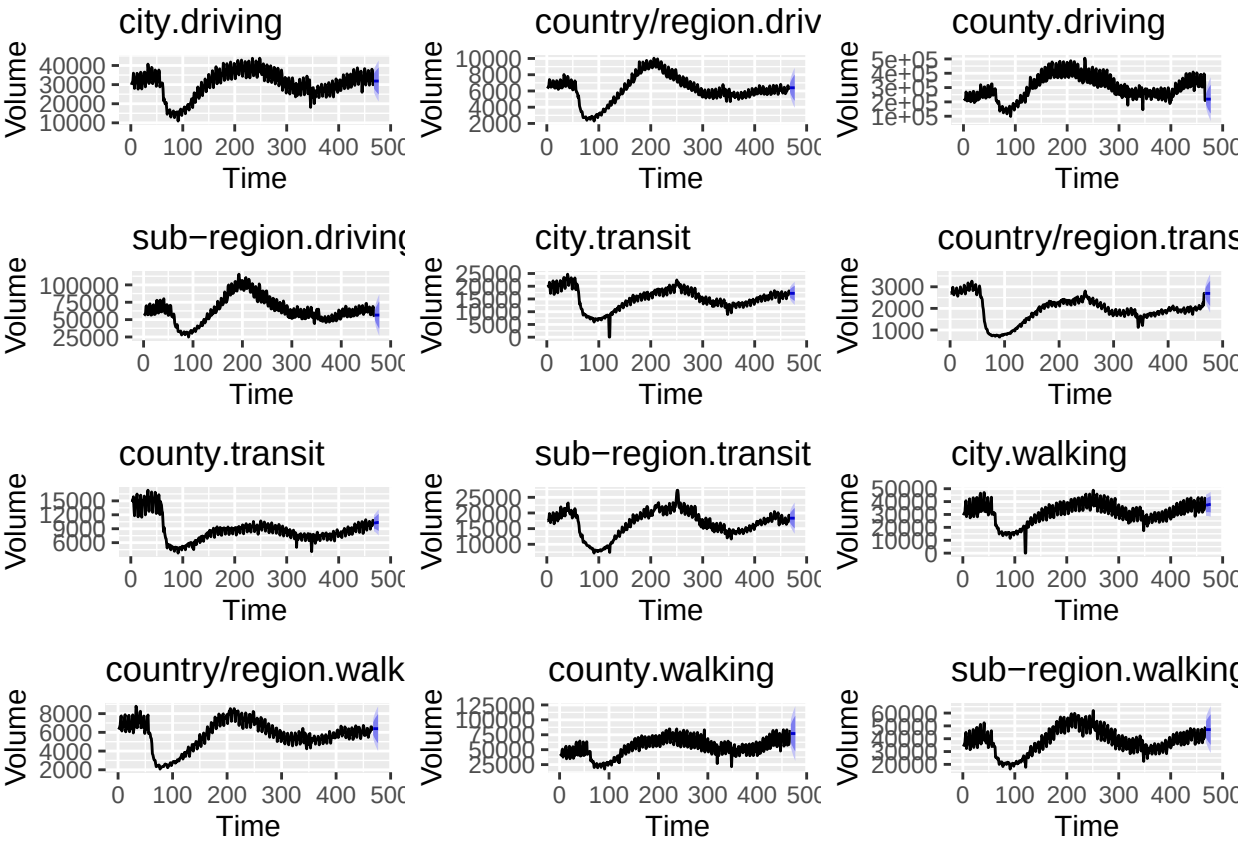
Forecasting

```
aTSModels <- purrr::map( names(zooObj), function(x) { forecast::auto.arima( zoo( x = zooObj[,x], order.
```

```
aTSModels <- purrr::map( names(zooObj), function(x) forecast::forecast( as.matrix(zooObj)[,x] ) )
names(aTSModels) <- names(zooObj)
```

```
lsPlots <- purrr::map( names(aTSModels), function(x) autoplot(aTSModels[[x]]) + ylab("Volume") + ggtitl
names(lsPlots) <- names(aTSModels)
```

```
do.call( gridExtra::grid.arrange, lsPlots )
```



Packages, Repositories, Articles