MS4S09 Coursework 2 - 2020/21

17076749

Mark Baber

11/03/2021

This report will look at using data mining techniques on a time series dataset. This report will be broken down into several parts, from getting the data, exploring the data, looking into trend and seasonality of the data before continuing onto a more in-depth analysis. This in-depth analysis will cover ARMA and forecasting.

All of this will be done using R and Rstudio, with the use of limited packages which has been added below: - magrittr - tseries - knitr

1 Task 1 – Getting the data (10%)

Write an R script that downloads the data directly from the website for the 30 time series (3 time series for each of the 10 districts) using the "Year ordered statistics" option, and selecting the districts listed. Download up to December 2021.

Create the 30 time-series objects in R to store the data you have downloaded. Remember to specify the appropriate starting point and frequency.

The section above looked to set up the base url for the datasets, the 3 different features which changed within the url and the 10 districts which will be needed for the url. Before going further into creating a function, there was some pre-analysis on the scraped dataset to calculate the number of rows and looked into which rows & columns should be omitted.

The next step would be to create the function which will grab each dataset, from the url using the base address, features and all districts.

```
Data$Tmax$Northern_Ireland %>% head()
```

```
## [1]  8.2  7.7  8.8 10.8 14.1 16.8
```

Here is a head of the first entry within Tmax - Now that the function has managed to get all datasets for each feature (TMAX-TMEAN-TMIN) lets move on to the next step which will be to explore these datasets.

2 - Task 2 – R programming (5%)

Write an R script to identify the district and date (year and month) of the highest and the lowest max, min and mean temperature (six results in total).

This section of the report will look to calculate the max, mean and min tempertures for each subset of data whilst pointing out the district and date.

```
##         Northern_Ireland               Scotland_N               Scotland_E
##                     1340                      764                     1471
##               Scotland_W         England_E_and_NE   England_NW_and_N_Wales
##                      764                     1471                     1471
##                 Midlands              East_Anglia   England_SW_and_S_Wales
##                     1471                     1471                     1340
## England_SE_and_Central_S
##                     1471
```

For task 2, I wasn't able to figure this out so this section will be left out.

3 - Task 3 – Exploratory Data Analysis (25%)

Carry out an EDA of the data you have downloaded. In order to complete your analysis, you may find it useful to answer (but not only!) the following questions:

– Which district is the coldest/warmest? Describe used criteria. – Which district has the widest temperature range? – Are winters/summers getting colder/hotter?

```
# find max value
colMax <- function(data) sapply(data, max, na.rm = TRUE)

colMax(Data$Tmax) %>% which.max()
```

```
## East_Anglia
##           8
```

```
# East_Anglia has the highest temp within the Tmax series
colMax(Data$Tmean) %>% which.max()
```

```
## East_Anglia
##           8
```

```
# East_Anglia has the highest temp within the Tmean series.
colMax(Data$Tmin) %>% which.max()
```

```
## England_SE_and_Central_S
##                       10
```

```
# England_SE_and_Central_S has the highest temp within Tmin.
```

Above I have looked at the max temperatures whilst also finding the max of those with which.max. Now lets create a function to find the lowest temperatures.

```
# find lowest value
colMin <- function(data) sapply(data, min, na.rm = TRUE)

colMin(Data$Tmax) %>% which.min()
```

```
## Midlands
##        7

# Midlands has the lowest temp within the Tmax series.
colMin(Data$Tmean) %>% which.min()

## Scotland_E
##          3

# Scotland_E has the lowest temp within the Tmean series.
colMin(Data$Tmin) %>% which.min()

## Scotland_E
##          3

# Scotland_E has the lowest temp within the Tmin series
```

This section looked at the lowest temperatures within the dataset, with Midlands having the lowest temp within Tmax, Scotland_E having the lowest within Tmean and again Scotland_E having the lowest for Tmin.

```
colRange <- function(data) sapply(data, range, na.rm = TRUE)

colRange(Data$Tmax)

##      Northern_Ireland Scotland_N Scotland_E Scotland_W England_E_and_NE
## [1,]              1.5        0.4       -0.5        0.6             -0.1
## [2,]             22.1       20.1       21.4       21.6             24.4
##      England_NW_and_N_Wales Midlands East_Anglia England_SW_and_S_Wales
## [1,]                   -0.2     -0.6        -0.2                    0.3
## [2,]                   23.3     25.7        26.7                   24.3
##      England_SE_and_Central_S
## [1,]                     -0.1
## [2,]                     26.1

colRange(Data$Tmean)

##      Northern_Ireland Scotland_N Scotland_E Scotland_W England_E_and_NE
## [1,]             -0.7       -2.4       -3.5       -2.3             -2.0
## [2,]             17.0       15.0       16.0       16.1             18.3
##      England_NW_and_N_Wales Midlands East_Anglia England_SW_and_S_Wales
## [1,]                   -2.6     -2.8        -2.5                   -2.4
## [2,]                   17.9     19.5        20.4                   18.8
##      England_SE_and_Central_S
## [1,]                     -2.7
## [2,]                     20.2

colRange(Data$Tmin)

##      Northern_Ireland Scotland_N Scotland_E Scotland_W England_E_and_NE
## [1,]             -4.3       -6.3       -7.4       -5.6             -5.5
## [2,]             12.2       11.0       11.0       11.6             12.6
```

```
##      England_NW_and_N_Wales Midlands East_Anglia England_SW_and_S_Wales
## [1,]                   -5.8     -6.6        -5.9                   -5.4
## [2,]                   12.7     13.5        14.6                   13.9
##      England_SE_and_Central_S
## [1,]                     -5.8
## [2,]                     14.7
```

This section looked at the ranges of all the dataset features, all of these could also be done with a couple of functions on the whole dataset.

```
# The above can also be done on the full dataset.
maxTemps <- sapply(Data, colMax)
minTemps <- sapply(Data, colMin)
totRange <- sapply(Data, colRange)
```

Here are the max temperatures:

```
maxTemps
```

```
##                          Tmax Tmean Tmin
## Northern_Ireland         22.1  17.0 12.2
## Scotland_N               20.1  15.0 11.0
## Scotland_E               21.4  16.0 11.0
## Scotland_W               21.6  16.1 11.6
## England_E_and_NE         24.4  18.3 12.6
## England_NW_and_N_Wales   23.3  17.9 12.7
## Midlands                 25.7  19.5 13.5
## East_Anglia              26.7  20.4 14.6
## England_SW_and_S_Wales   24.3  18.8 13.9
## England_SE_and_Central_S 26.1  20.2 14.7
```

Here are the min temperatures:

```
minTemps
```

```
##                          Tmax Tmean Tmin
## Northern_Ireland          1.5  -0.7 -4.3
## Scotland_N                0.4  -2.4 -6.3
## Scotland_E               -0.5  -3.5 -7.4
## Scotland_W                0.6  -2.3 -5.6
## England_E_and_NE         -0.1  -2.0 -5.5
## England_NW_and_N_Wales   -0.2  -2.6 -5.8
## Midlands                 -0.6  -2.8 -6.6
## East_Anglia              -0.2  -2.5 -5.9
## England_SW_and_S_Wales    0.3  -2.4 -5.4
## England_SE_and_Central_S -0.1  -2.7 -5.8
```

And here are the ranges:

```
totRange
```

```
##       Tmax Tmean Tmin
##  [1,]  1.5  -0.7 -4.3
##  [2,] 22.1  17.0 12.2
##  [3,]  0.4  -2.4 -6.3
##  [4,] 20.1  15.0 11.0
##  [5,] -0.5  -3.5 -7.4
##  [6,] 21.4  16.0 11.0
##  [7,]  0.6  -2.3 -5.6
##  [8,] 21.6  16.1 11.6
##  [9,] -0.1  -2.0 -5.5
## [10,] 24.4  18.3 12.6
## [11,] -0.2  -2.6 -5.8
## [12,] 23.3  17.9 12.7
## [13,] -0.6  -2.8 -6.6
## [14,] 25.7  19.5 13.5
## [15,] -0.2  -2.5 -5.9
## [16,] 26.7  20.4 14.6
## [17,]  0.3  -2.4 -5.4
## [18,] 24.3  18.8 13.9
## [19,] -0.1  -2.7 -5.8
## [20,] 26.1  20.2 14.7
```

And below is a function which looks to calculate the mean temperature for each district for each feature.

```
##                              Tmax    Tmean     Tmin
## Northern_Ireland         12.09057 8.571959 5.070985
## Scotland_N               10.03461 6.844161 3.740511
## Scotland_E               10.47251 6.894161 3.373601
## Scotland_W               11.03163 7.687348 4.436557
## England_E_and_NE         12.16679 8.412713 4.691423
## England_NW_and_N_Wales   11.91837 8.468187 5.057482
## Midlands                 12.94009 8.990815 5.052798
## East_Anglia              13.69738 9.621594 5.558698
## England_SW_and_S_Wales   12.99221 9.424270 5.877251
## England_SE_and_Central_S 13.82950 9.759793 5.711010
```

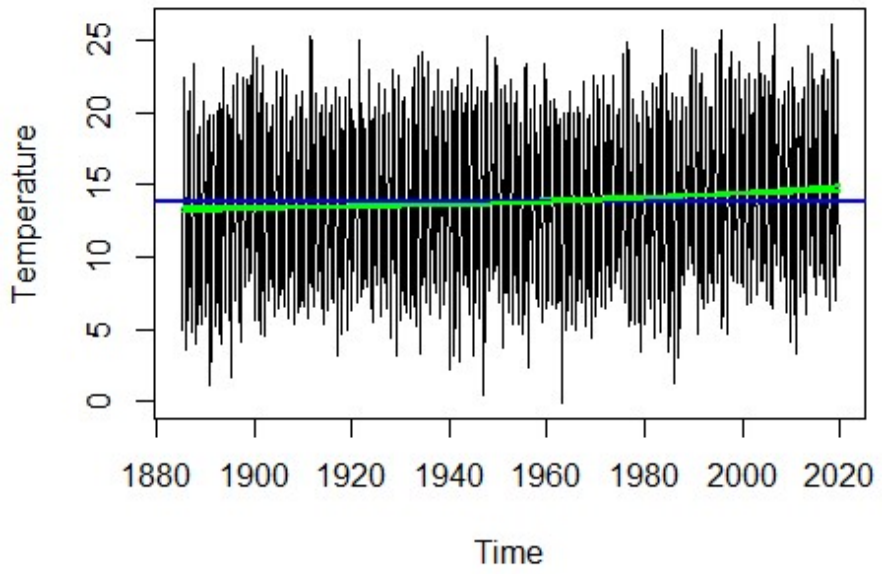
4 - Task 4 – Trend and Seasonality

4.0 - Subset

For each district, consider the 3 time series: max, mean, min. subset each of the 30 time series until December 2019.

This section plans to subset the dataset for the time series datasets.

The section above managed to slice a dataset at a time, but struggled to do this for all of the datasets within 'Data', I struggled converting the list to a matrix for each section.




```
## [1] 10136.89

## [1] 10137.68

## [1] 10136.8
```

Here is how we would check the AIC - lower is better.

The section above looks to create a linear, quadratic and cubic model for the selected sliced dataset and plots their values with a line.

Going through the trend for all of the datasets manually took a very long time (which I have omitted), but for most of them the AIC for linear.fit seemed to be the best model.

AIC's for all harmonic seasonalities

slice.har.1	slice.har.2	slice.har.3	slice.har.4	slice.har.5	slice.har.6
10134.41	10137.16	10140.49	10144.25	10147.99	10151.88

```
##   slice.har.1 slice.har.2 slice.har.3 slice.har.4 slice.har.5 slice.har.6
## 1    10134.41    10137.16    10140.49    10144.25    10147.99    10151.88
```

This section looked to estimate the seasonality for the sliced dataset whilst looking at the AIC results for each harmonic seasonality.

```
## Warning in adf.test(slicedWindow): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
```

```
##
## data:  slicedWindow
## Dickey-Fuller = -7.9003, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

5 - Task 5 – ARMA and Forecasting (20%)

```
# Using the final and the test modelestimated in the previous task,remove
trend and seasonality from each of the 30 time series. You will now have 60
residuals time series.

# Fit the residuals with an appropriate ARMA model.

# Forecast the average max, min and mean temperature for each month of 2020.
Remember that you also have to forecast the trend and seasonal components.

# Compare your forecasts with # the actual values. You may find it useful to
look at the following https://otexts.com/fpp2/accuracy.html Which model
performs better
```

6 - Reflection

This coursework seemed to be very overwhelming, with more knowledge of time series analysis I could see why creating multiple functions can speed things up within R. Whilst this is not really my area, I can see how import this could be within a business.

Many parts of this report could be improved upon especially with more time – I aim to learn more about time series over the summer.

Whilst the coursework seemed to be very straight forward working with 1 dataset, trying to create multiple functions for this timeseries analysis proved to be a lot more difficult than I thought. I was also struggling with some personal problems within my life, even with EC's I was really struggling to concentrate on this coursework.