Data Cleaning and Preparation

Rade Bajic, Eddie Morrissey, Stijn Jorissen, Nurly Kuzdikbay, David Basler

2023-07-20

Load general libraries

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.3.1
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.1
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```
## Warning: package 'tibble' was built under R version 4.3.1
```

```
## Warning: package 'tidyr' was built under R version 4.3.1
```

```
## Warning: package 'readr' was built under R version 4.3.1
```

```
## Warning: package 'purrr' was built under R version 4.3.1
```

```
## Warning: package 'stringr' was built under R version 4.3.1
```

```
## Warning: package 'forcats' was built under R version 4.3.1
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats 1.0.0     ✔ stringr 1.5.0
## ✔ ggplot2 3.4.2     ✔ tibble  3.2.1
## ✔ purrr   1.0.1     ✔ tidyr   1.3.0
## ✔ readr   2.1.4
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```r
library(ggplot2)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.3.1
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.3.1
```

```
## 
## Attaching package: 'reshape2'
## 
## The following object is masked from 'package:tidyr':
## 
##     smiths
```

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.1
```

```
## corrplot 0.92 loaded
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.1
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:ggplot2':
##
##     margin
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(tseries) # for ADF test
```

```
## Warning: package 'tseries' was built under R version 4.3.1
```

———— START OF DATA LOADING ————

Load datasets Load commodity price data

```r
data_commodity <- read.csv("./data/commodity-prices-2016.csv")
head(data_commodity)
```

```
##         Date All.Commodity.Price.Index Non.Fuel.Price.Index
## 1 1980-01-01                        NA                   NA
## 2 1980-02-01                        NA                   NA
## 3 1980-03-01                        NA                   NA
## 4 1980-04-01                        NA                   NA
## 5 1980-05-01                        NA                   NA
## 6 1980-06-01                        NA                   NA
##   Food.and.Beverage.Price.Index Food.Price.Index Beverage.Price.Index
## 1                            NA               NA             189.3100
## 2                            NA               NA             190.3879
## 3                            NA               NA             194.0604
## 4                            NA               NA             186.1379
## 5                            NA               NA             185.0702
## 6                            NA               NA             179.3753
##   Industrial.Inputs.Price.Index Agricultural.Raw.Materials.Index
## 1                      81.88965                         78.90015
## 2                      83.04837                         75.71515
## 3                      75.22890                         69.00247
## 4                      72.47125                         67.87711
## 5                      69.58810                         65.87967
## 6                      68.85570                         67.81455
##   Metals.Price.Index Fuel.Energy.Index Crude.Oil.petroleum Aluminum  Bananas
## 1           84.04900                NA            72.08931 2054.860 401.9608
## 2           88.34523                NA            69.83942 2131.009 372.1860
## 3           79.72631                NA            70.98153 1978.379 422.9135
## 4           75.78966                NA            70.40037 1932.456 395.8956
## 5           72.26675                NA            71.02119 1775.804 444.9690
## 6           69.60773                NA            70.98600 1668.960 342.4111
##     Barley   Beef     Coal Cocoa.beans Coffee.Other.Mild.Arabicas
## 1 66.58454 136.36 39.69663    3167.157                     168.67
## 2 66.58454 134.55 40.25813    3236.823                     164.83
## 3 69.89784 118.00 40.82757    3091.098                     184.38
## 4 69.89784 114.51 41.40507    2910.098                     180.81
## 5 68.24119 110.50 41.99073    2585.799                     190.54
## 6 64.92789 113.89 42.58469    2498.055                     181.41
##   Coffee.Robusta Rapeseed.oil   Copper   Cotton  Fishmeal Groundnuts.peanuts
## 1         162.56       591.59 2592.633 88.72000  986.5551           980.0752
## 2         162.00       596.60 2916.712 97.20999 1040.8056          1000.3759
## 3         169.89       561.51 2303.828 93.53000  960.4345          1009.3985
## 4         162.90       541.45 2074.548 90.56000  944.3603          1015.0376
## 5         174.06       536.44 2076.752 88.39999 1014.6850          1035.3383
## 6         169.01       546.46 2006.204 84.14001  972.4902          1043.2331
##   Hides China.import.Iron.Ore.Fines.62..FE.spot   Lamb      Lead Soft.Logs
## 1  59.1                                   12.15 117.23 1111.1284  84.39110
## 2  48.7                                   12.15 122.01 1166.2439  80.24920
## 3  39.4                                   12.15 119.32 1117.7424  80.24920
## 4  38.1                                   12.15 132.72  970.0327  87.49752
## 5  33.8                                   12.15 142.07  793.6631  87.49752
## 6  38.2                                   12.15 141.63  736.3430  87.49752
##   Hard.Logs Maize.corn
## 1  146.0755   105.5068
## 2  159.5655   114.1678
```

```
## 3  155.2755   109.8373
## 4  152.7855   108.2626
## 5  162.7355   109.8373
## 6  164.4255   113.3805
##   Natural.Gas...Russian.Natural.Gas.border.price.in.Germany
## 1                                                        NA
## 2                                                        NA
## 3                                                        NA
## 4                                                        NA
## 5                                                        NA
## 6                                                        NA
##   Natural.Gas...Indonesian.Liquefied.Natural.Gas.in.Japan
## 1                                                      NA
## 2                                                      NA
## 3                                                      NA
## 4                                                      NA
## 5                                                      NA
## 6                                                      NA
##   Natural.Gas...Spot.price.at.the.Henry.Hub.terminal.in.Louisiana   Nickel
## 1                                                              NA 6584.801
## 2                                                              NA 6978.928
## 3                                                              NA 6733.787
## 4                                                              NA 6233.369
## 5                                                              NA 6000.770
## 6                                                              NA 6294.834
##   Crude.Oil...petroleum.simple.average.of.three.spot.prices
## 1                                                     35.64
## 2                                                     35.09
## 3                                                     36.01
## 4                                                     35.09
## 5                                                     35.72
## 6                                                     35.54
##   Crude.Oil...petroleum...Dated.Brent.light.blend Oil.Dubai
## 1                                           40.00     38.00
## 2                                           38.50     36.00
## 3                                           38.25     35.75
## 4                                           38.15     35.00
## 5                                           38.50     35.60
## 6                                           38.00     36.00
##   Crude.Oil.petroleum...West.Texas.Intermediate.40.API Olive.Oil Oranges
## 1                                                37.00  2271.723   347.0
## 2                                                37.04  2256.483   350.0
## 3                                                39.52  2188.114   338.0
## 4                                                39.50  2081.168   377.0
## 5                                                39.50  2044.541   442.2
## 6                                                39.50  2053.294   480.0
##   Palm.oil Swine...pork Poultry.chicken Rice   Rubber Fish.salmon Hard.Sawnwood
## 1 547.0539     72.41854        33.90030  395 68.82001    7.452902      297.6097
## 2 555.3176     65.52780        32.55447  399 75.31000    7.604658      308.2893
## 3 518.1311     65.00207        31.82699  415 66.35001    7.400278      304.9222
## 4 504.9093     51.85896        30.99040  419 60.55000    7.426402      302.9500
## 5 482.5974     54.90065        30.66304  433 60.39000    7.693488      310.7206
```

```
## 6 458.6328     62.46733        31.28139  442 61.87000    8.016716      312.0266
##   Soft.Sawnwood   Shrimp Soybean.Meal Soybean.Oil Soybeans
## 1      138.0042 13.33795     201.7560    525.5814 238.7660
## 2      131.2310 12.67656     198.2617    518.7471 241.3613
## 3      131.2310 12.78680     186.1032    486.7801 227.0758
## 4      143.0841 12.34587     181.1979    451.0653 218.2105
## 5      143.0841 12.01518     187.3929    463.1907 225.9045
## 6      143.0841 11.46402     190.0605    485.0164 232.6338
##   Sugar.European.import.price Sugar.Free.Market Sugar.U.S..import.price
## 1                          NA             17.30                   19.66
## 2                          NA             22.75                   24.69
## 3                          NA             19.63                   21.18
## 4                          NA             21.25                   22.67
## 5                          NA             30.94                   31.89
## 6                          NA             30.80                   32.10
##   Sunflower.oil      Tea      Tin Uranium    Wheat Wool.coarse Wool.fine
## 1      566.9270 225.1799 16973.59    40.0 175.6348    553.1209  684.2774
## 2      573.9586 233.0945 17090.21    38.0 172.6952    568.1548  722.5671
## 3      535.2845 226.8333 17460.59    35.0 163.5093    552.7451  695.9569
## 4      486.0630 221.8068 17041.71    32.0 156.5280    510.6503  688.1304
## 5      502.7631 229.6112 17180.60    32.0 161.3047    524.9324  720.7610
## 6      493.9736 235.4093 17211.47    31.5 157.6303    532.9505  737.2568
##       Zinc
## 1 773.8215
## 2 868.6204
## 3 740.7524
## 4 707.6831
## 5 701.0691
## 6 676.8184
```

Load weather data

```
data_weather <- read.csv("./data/average_monthly_temperature_by_state_1950-2022.csv")
head(data_weather)
```

```
##   X month year       state average_temp monthly_mean_from_1901_to_2000
## 1 0     1 1950     Alabama         53.8                           45.9
## 2 1     1 1950     Arizona         39.6                           41.1
## 3 2     1 1950    Arkansas         45.6                           40.4
## 4 3     1 1950  California         39.4                           42.7
## 5 4     1 1950    Colorado         25.2                           24.5
## 6 5     1 1950 Connecticut         32.5                           27.3
##   centroid_lon centroid_lat
## 1    -86.82837     32.78983
## 2   -111.66442     34.29311
## 3    -92.43927     34.89975
## 4   -119.61070     37.24607
## 5   -105.54782     38.99855
## 6    -72.72571     41.62029
```

Load macroeconomic data

```
data_macro <- read.csv("./data/US_macroeconomics.csv")
head(data_macro)
```

```
##         date  CPI Mortgage_rate Unemp_rate   NASDAQ disposable_income
## 1 1980-11-01 85.6       14.2050        7.5 200.6856            4976.5
## 2 1980-12-01 86.4       14.7900        7.2 198.3986            4999.8
## 3 1981-01-01 87.2       14.9040        7.5 198.8176            4980.4
## 4 1981-02-01 88.0       15.1325        7.4 194.8521            4965.0
## 5 1981-03-01 88.6       15.4000        7.4 203.5932            4979.0
## 6 1981-04-01 89.1       15.5800        7.2 215.1200            4965.1
##   Personal_consumption_expenditure personal_savings
## 1                           1826.8             11.6
## 2                           1851.7             11.4
## 3                           1870.0             10.9
## 4                           1884.2             10.8
## 5                           1902.9             10.8
## 6                           1904.4             10.9
```

———————— END OF DATA LOADING ——————————

———————— START OF GENERAL DATA CLEANING AND PREPROCESSING ——————————

Change to date type

```
data_commodity$Date <- as.Date(data_commodity$Date)
data_macro$Date <- as.Date(data_macro$date)
data_weather$Date <- as.Date(paste(data_weather$year, data_weather$month, "01", sep='-'))
```

Clean Commodity dataset

```
na_columns <- colSums(is.na(data_commodity)) >0 # there are 5-6 columns with NA values. As these seem not to be very important, we delete these columns
data_commodity <- data_commodity[, !na_columns]
data_commodity_filt <- data_commodity %>%
  filter(Date >= as.Date('1980-11-01') & Date <= as.Date('2016-02-01'))
```

Clean Weather dataset

```r
na_columns <- colSums(is.na(data_weather)) >0 # there are no NA values in the dataset
data_weather <- data_weather[c('Date', 'state', 'average_temp')] #only keep relevant columns for analysis
data_weather <- pivot_wider(data_weather, names_from = state, values_from = average_temp) # pivot states in columns
data_weather_filt <- data_weather %>%
  filter(Date >= as.Date('1980-11-01') & Date <= as.Date('2016-02-01'))
```

Clean Macro dataset

```r
na_columns <- colSums(is.na(data_macro)) >0 # there are no NA values in the dataset
data_macro <- data_macro[, -which(names(data_macro) == 'date')] #only keep relevant columns for analysis
data_macro_filt <- data_macro %>%
  filter(Date >= as.Date('1980-11-01') & Date <= as.Date('2016-02-01'))
```

Merge datasets

```r
df <- merge(data_commodity_filt, data_macro_filt, by='Date')
df <- merge(df, data_weather_filt, by='Date') # merged dataset for analysis
colnames(df) <- gsub("\\.", "_", colnames(df))
column_names <- names(df)
print(column_names) # 1 = Date, 2:55 = Commodity Prices, 56:62 = Macro inputs, 63:110 = Average temperatures by state
```

```
##  [1] "Date"
##  [2] "Beverage_Price_Index"
##  [3] "Industrial_Inputs_Price_Index"
##  [4] "Agricultural_Raw_Materials_Index"
##  [5] "Metals_Price_Index"
##  [6] "Crude_Oil_petroleum"
##  [7] "Aluminum"
##  [8] "Bananas"
##  [9] "Barley"
## [10] "Beef"
## [11] "Coal"
## [12] "Cocoa_beans"
## [13] "Coffee_Other_Mild_Arabicas"
## [14] "Coffee_Robusta"
## [15] "Rapeseed_oil"
## [16] "Copper"
## [17] "Cotton"
## [18] "Fishmeal"
## [19] "Groundnuts_peanuts"
## [20] "Hides"
## [21] "China_import_Iron_Ore_Fines_62__FE_spot"
## [22] "Lamb"
## [23] "Lead"
## [24] "Soft_Logs"
## [25] "Hard_Logs"
## [26] "Maize_corn"
## [27] "Nickel"
## [28] "Crude_Oil___petroleum_simple_average_of_three_spot_prices"
## [29] "Crude_Oil___petroleum___Dated_Brent_light_blend"
## [30] "Oil_Dubai"
## [31] "Crude_Oil_petroleum___West_Texas_Intermediate_40_API"
## [32] "Olive_Oil"
## [33] "Oranges"
## [34] "Palm_oil"
## [35] "Swine___pork"
## [36] "Poultry_chicken"
## [37] "Rice"
## [38] "Rubber"
## [39] "Fish_salmon"
## [40] "Hard_Sawnwood"
## [41] "Soft_Sawnwood"
## [42] "Shrimp"
## [43] "Soybean_Meal"
## [44] "Soybean_Oil"
## [45] "Soybeans"
## [46] "Sugar_Free_Market"
## [47] "Sugar_U_S__import_price"
## [48] "Sunflower_oil"
## [49] "Tea"
## [50] "Tin"
## [51] "Uranium"
## [52] "Wheat"
```

```
##  [53] "Wool_coarse"
##  [54] "Wool_fine"
##  [55] "Zinc"
##  [56] "CPI"
##  [57] "Mortgage_rate"
##  [58] "Unemp_rate"
##  [59] "NASDAQ"
##  [60] "disposable_income"
##  [61] "Personal_consumption_expenditure"
##  [62] "personal_savings"
##  [63] "Alabama"
##  [64] "Arizona"
##  [65] "Arkansas"
##  [66] "California"
##  [67] "Colorado"
##  [68] "Connecticut"
##  [69] "Delaware"
##  [70] "Florida"
##  [71] "Georgia"
##  [72] "Idaho"
##  [73] "Illinois"
##  [74] "Indiana"
##  [75] "Iowa"
##  [76] "Kansas"
##  [77] "Kentucky"
##  [78] "Louisiana"
##  [79] "Maine"
##  [80] "Maryland"
##  [81] "Massachusetts"
##  [82] "Michigan"
##  [83] "Minnesota"
##  [84] "Mississippi"
##  [85] "Missouri"
##  [86] "Montana"
##  [87] "Nebraska"
##  [88] "Nevada"
##  [89] "New Hampshire"
##  [90] "New Jersey"
##  [91] "New Mexico"
##  [92] "New York"
##  [93] "North Carolina"
##  [94] "North Dakota"
##  [95] "Ohio"
##  [96] "Oklahoma"
##  [97] "Oregon"
##  [98] "Pennsylvania"
##  [99] "Rhode Island"
## [100] "South Carolina"
## [101] "South Dakota"
## [102] "Tennessee"
## [103] "Texas"
## [104] "Utah"
```

```
## [105] "Vermont"
## [106] "Virginia"
## [107] "Washington"
## [108] "West Virginia"
## [109] "Wisconsin"
## [110] "Wyoming"
```

Average US Temperature

```
df$USA_Avg_Temp <- rowMeans(df[, 63:110], na.rm=TRUE) #average all states together
df <- df[, -c(63:110)]
head(df)
```

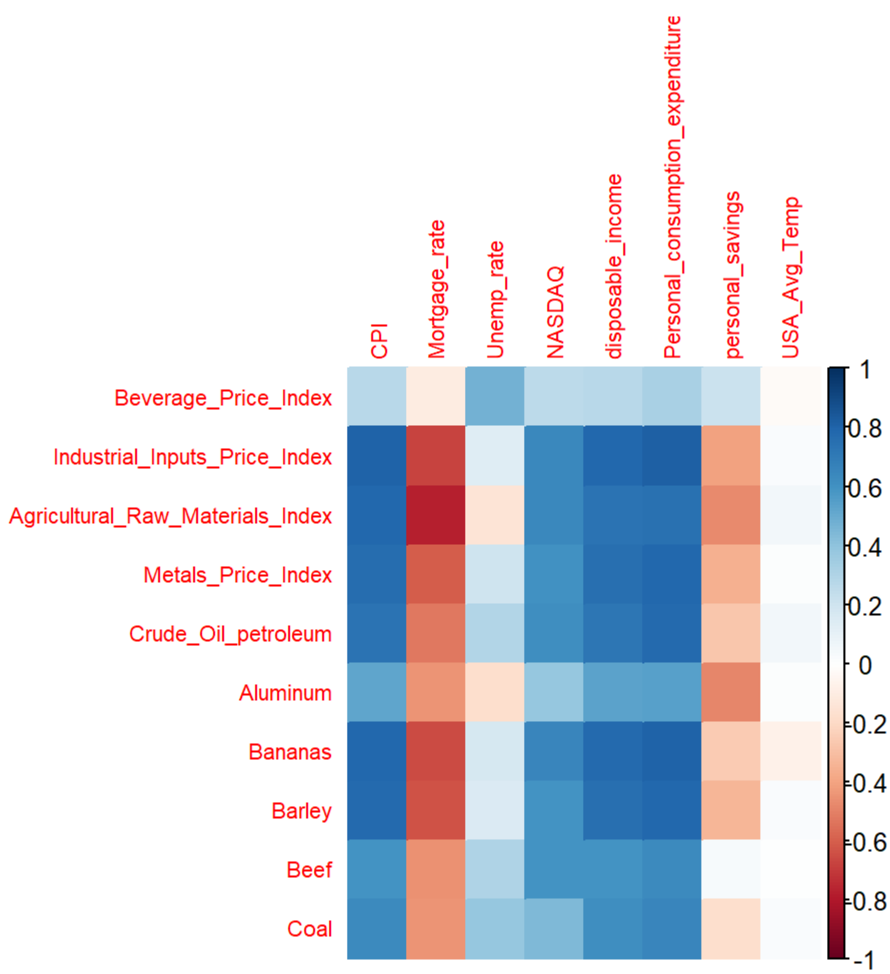
```
##         Date Beverage_Price_Index Industrial_Inputs_Price_Index
## 1 1980-11-01             136.3164                      69.68354
## 2 1980-12-01             136.5980                      67.13575
## 3 1981-01-01             139.2010                      65.61433
## 4 1981-02-01             135.4352                      63.80699
## 5 1981-03-01             136.8588                      63.79768
## 6 1981-04-01             136.5669                      63.27183
##   Agricultural_Raw_Materials_Index Metals_Price_Index Crude_Oil_petroleum
## 1                         74.29385           66.35346            73.19304
## 2                         72.35484           63.36594            73.08986
## 3                         69.44860           62.84479            73.71042
## 4                         65.82446           62.34975            71.27406
## 5                         65.75739           62.38215            70.89036
## 6                         65.96707           61.32503            69.29339
##   Aluminum  Bananas    Barley   Beef     Coal Cocoa_beans
## 1 1503.910 379.3540 103.22196 133.25 45.68286    2173.755
## 2 1430.657 369.9804  94.87500 124.77 46.32904    2109.821
## 3 1430.830 387.6248  92.51746 121.73 46.98435    2093.066
## 4 1452.381 435.5954  94.17411 116.75 47.64894    2040.817
## 5 1442.952 459.3051  89.20417 113.30 48.32292    2115.774
## 6 1369.235 425.6705  82.57757 114.80 49.00644    2109.601
##   Coffee_Other_Mild_Arabicas Coffee_Robusta Rapeseed_oil   Copper   Cotton
## 1                     114.86         116.36       595.60 2010.614 98.03000
## 2                     121.21         118.54       576.55 1878.337 99.16000
## 3                     127.98         122.13       519.39 1876.132 99.51001
## 4                     125.11         115.48       492.32 1803.379 95.85001
## 5                     125.93         113.92       504.35 1816.607 91.72000
## 6                     128.20         112.93       516.38 1823.221 88.64999
##   Fishmeal Groundnuts_peanuts Hides China_import_Iron_Ore_Fines_62__FE_spot
## 1 1141.269           2706.767  54.3                                   12.15
## 2 1119.167           2622.180  50.1                                   12.15
## 3 1087.019           2255.639  44.8                                   12.15
## 4 1040.806           1691.729  40.7                                   12.15
## 5 1018.704           1578.947  42.4                                   12.15
## 6 1006.648           1590.226  45.9                                   12.15
##     Lamb     Lead Soft_Logs Hard_Logs Maize_corn   Nickel
## 1 124.20 813.5049  71.96541  107.9755   146.8434 6452.666
## 2 120.62 742.9570  73.00089  109.4355   145.2687 6390.912
## 3 133.28 703.2737  70.41220  112.3455   151.9613 6403.773
## 4 134.12 690.0461  67.82352  109.1155   143.3003 6370.748
## 5 124.75 727.5247  69.37673  106.7755   142.1193 6292.434
## 6 136.17 758.3892  66.78805  102.2855   144.4814 6307.061
##   Crude_Oil___petroleum_simple_average_of_three_spot_prices
## 1                                                     40.97
## 2                                                     39.05
## 3                                                     38.69
## 4                                                     36.75
## 5                                                     36.44
## 6                                                     35.70
##   Crude_Oil___petroleum___Dated_Brent_light_blend Oil_Dubai
## 1                                           40.85     39.75
## 2                                           40.15     39.35
```

```
## 3                                          40.30     39.25
## 4                                          38.70     37.10
## 5                                          38.35     36.85
## 6                                          37.19     35.55
##   Crude_Oil_petroleum___West_Texas_Intermediate_40_API Olive_Oil Oranges
## 1                                                35.98  2267.237   316.5
## 2                                                36.99  2210.018   321.6
## 3                                                38.00  2218.428   348.8
## 4                                                38.00  2058.948   325.2
## 5                                                38.00  2033.036   386.5
## 6                                                37.99  1991.441   403.0
##   Palm_oil Swine___pork Poultry_chicken Rice Rubber Fish_salmon Hard_Sawnwood
## 1 483.4238    112.69279        37.02843  463  64.22    8.352088      265.5024
## 2 513.9993    102.32851        36.70107  470  62.66    7.975751      266.7920
## 3 516.4784     92.77159        35.90085  470  62.18    7.747213      269.3469
## 4 528.8739     93.33486        36.70107  480  59.81    7.366256      266.5098
## 5 512.3466     76.68067        36.00997  505  57.28    7.385256      264.4383
## 6 485.9029     92.33974        35.13700  515  52.12    7.271337      260.4243
##   Soft_Sawnwood   Shrimp Soybean_Meal Soybean_Oil Soybeans Sugar_Free_Market
## 1      117.6846 10.36171     302.6726    615.0890 334.9063             37.81
## 2      119.3779 10.25148     256.9929    547.4071 293.2899             28.79
## 3      115.1446 10.25148     247.5020    532.1953 283.4830             27.78
## 4      110.9114 10.58218     237.8457    525.1405 275.6742             24.09
## 5      113.4513 11.24356     231.4413    531.0930 272.4817             21.81
## 6      109.2181 11.57426     247.4689    539.6910 288.2602             17.83
##   Sugar_U_S__import_price Sunflower_oil      Tea      Tin Uranium    Wheat
## 1                   39.28      631.0908 223.5705 15599.89      28 195.1089
## 2                   30.29      632.8487 216.2512 14698.20      27 182.6160
## 3                   29.57      606.4800 217.3755 14360.89      25 189.5973
## 4                   26.07      571.3217 222.8430 13679.67      25 181.5137
## 5                   23.81      571.3217 221.7186 13624.55      25 175.2673
## 6                   19.91      573.0797 216.1409 13344.57      25 180.4114
##   Wool_coarse Wool_fine     Zinc  CPI Mortgage_rate Unemp_rate   NASDAQ
## 1    525.9347  735.9322 800.2771 85.6       14.2050        7.5 200.6856
## 2    528.1898  736.0527 782.6401 86.4       14.7900        7.2 198.3986
## 3    527.3128  758.6894 776.0261 87.2       14.9040        7.5 198.8176
## 4    512.0284  740.9892 731.9338 88.0       15.1325        7.4 194.8521
## 5    502.7575  719.4364 756.1846 88.6       15.4000        7.4 203.5932
## 6    496.6187  721.2425 824.5278 89.1       15.5800        7.2 215.1200
##   disposable_income Personal_consumption_expenditure personal_savings
## 1            4976.5                           1826.8             11.6
## 2            4999.8                           1851.7             11.4
## 3            4980.4                           1870.0             10.9
## 4            4965.0                           1884.2             10.8
## 5            4979.0                           1902.9             10.8
## 6            4965.1                           1904.4             10.9
##   USA_Avg_Temp
## 1     46.51667
## 2     37.00000
## 3     30.50208
## 4     31.94167
```

```
## 5      38.37917
## 6      47.88542
```

Correlation Analysis

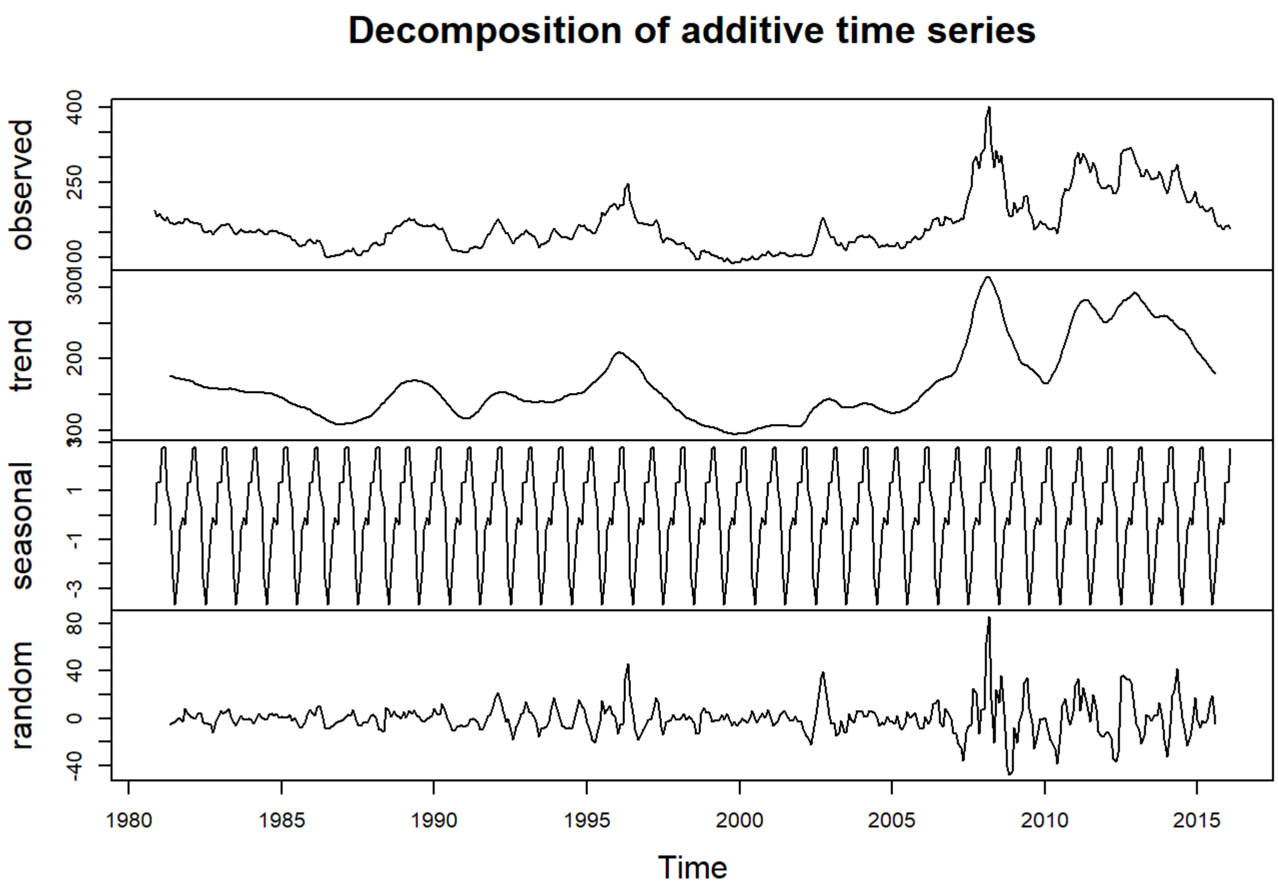
```
dependent_vars <- df[, -c(56:63)] # exclude the predictors
predictor_vars <- df[, c(56:63)] # exclude the dependent variables
dependent_vars <- dependent_vars[, !colnames(dependent_vars) %in% "Date"] #for correlation analy
sis exclude the date
dependent_vars_10 <- dependent_vars[, c(1:10)] # create several parts, as it is easier to look a
t
dependent_vars_20 <- dependent_vars[, c(11:20)]
dependent_vars_30 <- dependent_vars[, c(21:30)]
dependent_vars_40 <- dependent_vars[, c(31:40)]
dependent_vars_50 <- dependent_vars[, c(41:50)]
dependent_vars_60 <- dependent_vars[, c(51:54)]
cor_matrix_10 <- cor(dependent_vars_10, predictor_vars, use = "complete.obs")
corrplot(cor_matrix_10, method = "color", type = "full", tl.cex = 0.7)
```



```
#cor_matrix_20 <- cor(dependent_vars_20, predictor_vars, use = "complete.obs")
#corrplot(cor_matrix_20, method = "color", type = "full", tl.cex = 0.7)

#cor_matrix_30 <- cor(dependent_vars_30, predictor_vars, use = "complete.obs")
#corrplot(cor_matrix_30, method = "color", type = "full", tl.cex = 0.7)

#cor_matrix_40 <- cor(dependent_vars_40, predictor_vars, use = "complete.obs")
#corrplot(cor_matrix_40, method = "color", type = "full", tl.cex = 0.7)

#cor_matrix_50 <- cor(dependent_vars_50, predictor_vars, use = "complete.obs")
#corrplot(cor_matrix_50, method = "color", type = "full", tl.cex = 0.7)

#cor_matrix_60 <- cor(dependent_vars_60, predictor_vars, use = "complete.obs")
#corrplot(cor_matrix_60, method = "color", type = "full", tl.cex = 0.7)
```

Create triple exponential smoothing Holt-Winters method ##

```
dfts <- ts(df$Wheat, frequency=12, start= c(1980,11))
components_dfts <- decompose(dfts)
plot(components_dfts)
```



———— END OF GENERAL DATA CLEANING AND PREPROCESSING ————

———— DATA INSPECTION AND ADF TEST ————

```
any_negative <- any(df < 0) #check for null or negative values which would distort log
any_null <- any(is.null(df) | is.na(df)) #chekc for null or negative values which would distort
log
print(any_negative) #any negative values in the dataset?
```

```
## [1] FALSE
```

```
print(any_null) #any NULL values in the dataset?
```

```
## [1] FALSE
```

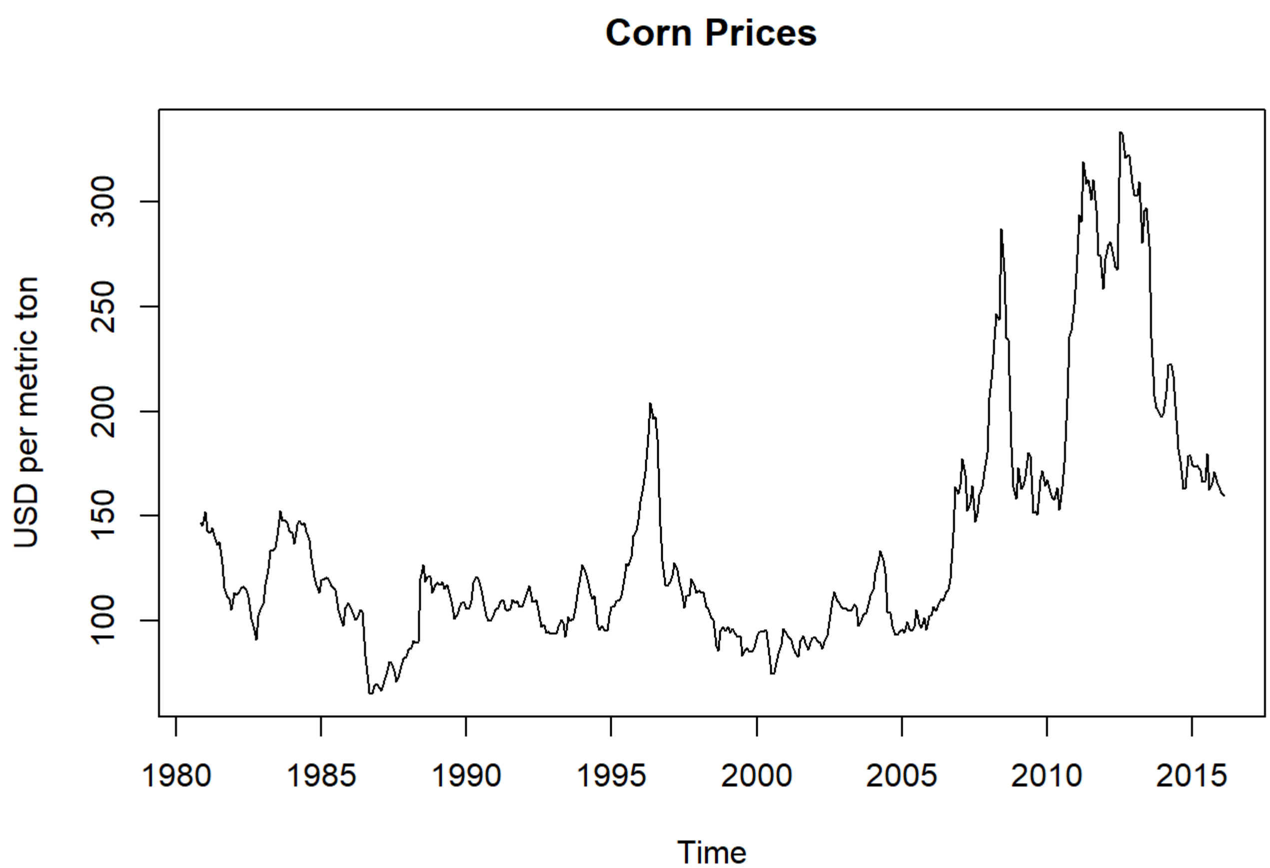
Start with the cleaned and prepared df

```
dependent_vars <- data.frame(soybeans = df$Soybeans, corn = df$Maize_corn)
predictor_vars <- df[, c(56:63)] # exclude the dependent variables

vars <- predictor_vars
vars <- cbind(date = df$Date, corn = dependent_vars$corn, vars)
head(vars)
```

```
##         date     corn  CPI Mortgage_rate Unemp_rate   NASDAQ disposable_income
## 1 1980-11-01 146.8434 85.6       14.2050        7.5 200.6856            4976.5
## 2 1980-12-01 145.2687 86.4       14.7900        7.2 198.3986            4999.8
## 3 1981-01-01 151.9613 87.2       14.9040        7.5 198.8176            4980.4
## 4 1981-02-01 143.3003 88.0       15.1325        7.4 194.8521            4965.0
## 5 1981-03-01 142.1193 88.6       15.4000        7.4 203.5932            4979.0
## 6 1981-04-01 144.4814 89.1       15.5800        7.2 215.1200            4965.1
##   Personal_consumption_expenditure personal_savings USA_Avg_Temp
## 1                           1826.8             11.6     46.51667
## 2                           1851.7             11.4     37.00000
## 3                           1870.0             10.9     30.50208
## 4                           1884.2             10.8     31.94167
## 5                           1902.9             10.8     38.37917
## 6                           1904.4             10.9     47.88542
```

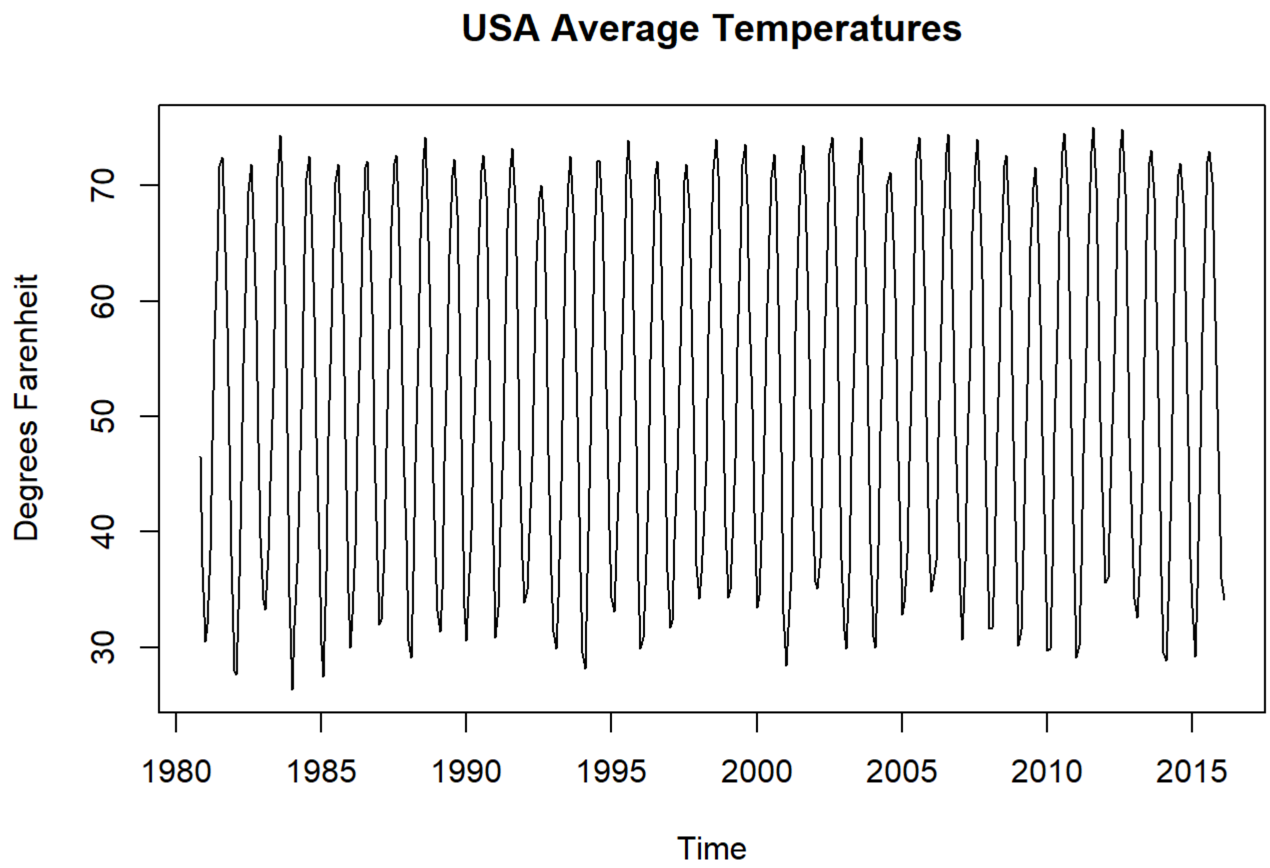
Convert Y to a time series object

```
ts_corn <- ts(vars$corn,  start = c(1980, 11), frequency = 12)
plot(ts_corn, ylab = 'USD per metric ton', main = 'Corn Prices') #
```



Convert Weather to a time series object

```
ts <- ts(vars$USA_Avg_Temp,  start = c(1980, 11), frequency = 12)
plot(ts, ylab = 'Degrees Farenheit', main='USA Average Temperatures')
```



Perform the ADF test8 for corn

```
adf_result <- adf.test(ts_corn, k = trunc((length(ts_corn)-3)^(1/4)))
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_corn
## Dickey-Fuller = -3.1223, Lag order = 4, p-value = 0.1033
## alternative hypothesis: stationary
```

```
adf_result$p.value
```

```
## [1] 0.1032673
```

Perform the ADF test8 for all relevant variables

```
# Create an empty data frame to store the results
adf_results <- data.frame(Variable = character(), TestStatistic = numeric(), p_value = numeric
(), stringsAsFactors = FALSE)
# Loop through each variable in the 'vars' dataset
for (col in colnames(vars)) {
  # Convert the variable into a time series object
  ts_data <- ts(vars[[col]],start = c(1980, 11), frequency = 12)

  # Perform the ADF test
  adf_result <- adf.test(ts_data, alternative = "stationary")

  # Extract the test statistics and p-value
  test_statistic <- adf_result$statistic
  p_value <- adf_result$p.value

  # Add the results to the data frame
  adf_results <- adf_results %>%
    add_row(Variable = col, TestStatistic = test_statistic, p_value = p_value)
}
```

```
## Warning in adf.test(ts_data, alternative = "stationary"): p-value smaller than
## printed p-value

## Warning in adf.test(ts_data, alternative = "stationary"): p-value smaller than
## printed p-value
```

```
# Print the table of results
print(adf_results)
```

```
##                          Variable TestStatistic    p_value
## 1                            date     -8.773426 0.01000000
## 2                            corn     -2.770877 0.25176236
## 3                             CPI     -2.127983 0.52344424
## 4                   Mortgage_rate     -3.450241 0.04738837
## 5                      Unemp_rate     -3.166638 0.09374952
## 6                          NASDAQ     -2.899076 0.19758658
## 7               disposable_income     -1.897668 0.62077328
## 8  Personal_consumption_expenditure     -1.773082 0.67342254
## 9                 personal_savings     -1.806196 0.65942879
## 10                   USA_Avg_Temp    -11.616001 0.01000000
```

Create table for report

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.3.1
```

```
library(kableExtra)
```

```
## Warning: package 'kableExtra' was built under R version 4.3.1
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
# Convert p-values to formatted character strings
adf_results$p_value <- sprintf("%.8f", adf_results$p_value)

# Create the table using kable()
table_output <- kable(adf_results, align = "c") %>%
  kable_styling(bootstrap_options = "striped", full_width = FALSE) %>%
  add_header_above(c('Augmented Dickey-Fuller-Test'=3))

# Print the table
print(table_output)
```

Augmented Dickey-Fuller-Test		
Variable	**TestStatistic**	**p_value**
date	-8.773426	0.01000000
corn	-2.770877	0.25176236
CPI	-2.127983	0.52344424
Mortgage_rate	-3.450241	0.04738837
Unemp_rate	-3.166638	0.09374952
NASDAQ	-2.899076	0.19758658
disposable_income	-1.897668	0.62077328
Personal_consumption_expenditure	-1.773082	0.67342254
personal_savings	-1.806196	0.65942879
USA_Avg_Temp	-11.616001	0.01000000

———— END DATA INSPECTION AND ADF TEST —————

———— TRANSORM DATA FOR DEEP LEARNING APPROACH —————

```
library(dplyr)
head(vars)
```

```
##         date     corn  CPI Mortgage_rate Unemp_rate   NASDAQ disposable_income
## 1 1980-11-01 146.8434 85.6       14.2050        7.5 200.6856            4976.5
## 2 1980-12-01 145.2687 86.4       14.7900        7.2 198.3986            4999.8
## 3 1981-01-01 151.9613 87.2       14.9040        7.5 198.8176            4980.4
## 4 1981-02-01 143.3003 88.0       15.1325        7.4 194.8521            4965.0
## 5 1981-03-01 142.1193 88.6       15.4000        7.4 203.5932            4979.0
## 6 1981-04-01 144.4814 89.1       15.5800        7.2 215.1200            4965.1
##   Personal_consumption_expenditure personal_savings USA_Avg_Temp
## 1                           1826.8             11.6     46.51667
## 2                           1851.7             11.4     37.00000
## 3                           1870.0             10.9     30.50208
## 4                           1884.2             10.8     31.94167
## 5                           1902.9             10.8     38.37917
## 6                           1904.4             10.9     47.88542
```

```
vars_nonstationary <-  select(vars, corn, CPI, Unemp_rate, NASDAQ, disposable_income, Personal_c
onsumption_expenditure, personal_savings) # those have to be transformed
vars_stationary <- select(vars, Mortgage_rate, USA_Avg_Temp) # vars ready for model, but need ti
ming adjustment because of the non-stationary variables which will lose their first row in the p
rocess
```

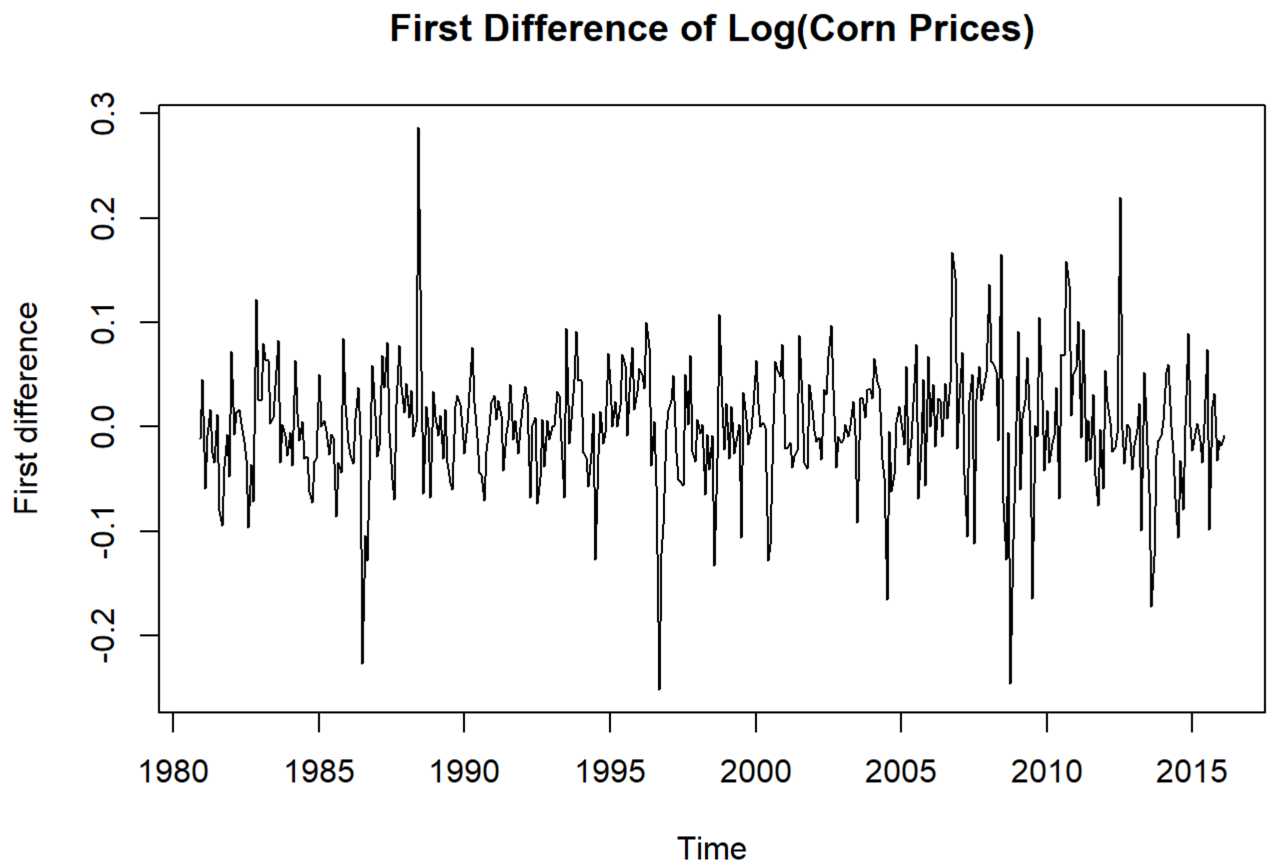
create the first difference of the logarithmized series

```r
# Create an empty dataframe to store the first differences
vars_diff <- data.frame(matrix(ncol = ncol(vars_nonstationary), nrow = nrow(vars_nonstationary)-
1))
# Loop through each variable in vars_nonstationary
for (col_name in colnames(vars_nonstationary)) {
  # Convert column to a time series object
  ts_col <- ts(vars_nonstationary[[col_name]], start = c(1980, 11), frequency = 12)

  # Calculate the first difference of the logarithmized series
  diff_series <- diff(log(ts_col))

  # Assign the differenced series to the new dataframe, excluding the first row
  vars_diff[[paste0(col_name, "_diff")]] <- diff_series
}
vars_diff <- vars_diff %>%
  select(contains('_diff'))
head(vars_diff)
```

```
##      corn_diff    CPI_diff Unemp_rate_diff  NASDAQ_diff disposable_income_diff
## 1 -0.010781732 0.009302393     -0.04082199 -0.011460961           0.004671079
## 2  0.045040722 0.009216655      0.04082199  0.002109596          -0.003887703
## 3 -0.058683522 0.009132484     -0.01342302 -0.020147081          -0.003096912
## 4 -0.008275903 0.006795043      0.00000000  0.043882960           0.002815770
## 5  0.016483876 0.005627477     -0.02739897  0.055072216          -0.002795629
## 6 -0.024828841 0.006711435      0.04082199  0.006595439           0.001951731
##   Personal_consumption_expenditure_diff personal_savings_diff
## 1                           0.013538334          -0.017391743
## 2                           0.009834295          -0.044850566
## 3                           0.007564897          -0.009216655
## 4                           0.009875711           0.000000000
## 5                           0.000787960           0.009216655
## 6                           0.004923796           0.009132484
```

Check if the diff is correctly calculated by taking one example

```r
#log_ts_corn <- log(ts_corn)
#diff_log_ts_corn <- diff(log_ts_corn)
#plot(diff_log_ts_corn, ylab = 'First difference', main = 'First Difference of Log(Corn Price
s)')
plot(vars_diff$corn_diff, ylab = 'First difference', main = 'First Difference of Log(Corn Price
s)')
```



Transform stationary data to time series

```
# Create an empty dataframe to store the transformed time series
ts_vars_stationary <- data.frame(matrix(ncol = ncol(vars_stationary), nrow = nrow(vars_stationar
y)-1))
# Set the column names of vars_ts
colnames(ts_vars_stationary) <- colnames(vars_stationary)

# Loop through each column in vars_stationary
for (i in 1:ncol(vars_stationary)) {
  # Convert column to a time series object
  ts_vars_stationary[, i] <- ts(vars_stationary[-1, i],start = c(1980, 11), frequency = 12)
}

# View the resulting dataframe
str(ts_vars_stationary)
```

```
## 'data.frame':    423 obs. of  2 variables:
##  $ Mortgage_rate: Time-Series  from 1981 to 2016: 14.8 14.9 15.1 15.4 15.6 ...
##  $ USA_Avg_Temp : Time-Series  from 1981 to 2016: 37 30.5 31.9 38.4 47.9 ...
```

———— END TRANSORM DATA FOR DEEP LEARNING APPROACH —————

———— START TEST FOR CAUSALITY —————

```
#head(vars_diff)
#head(ts_vars_stationary)
nrow(ts_vars_stationary) # this has one row more, because no transformation happened, hence we h
ave to delete the first row
```

```
## [1] 423
```

```
nrow(vars_diff) # this has one row less, because of the transformation
```

```
## [1] 423
```

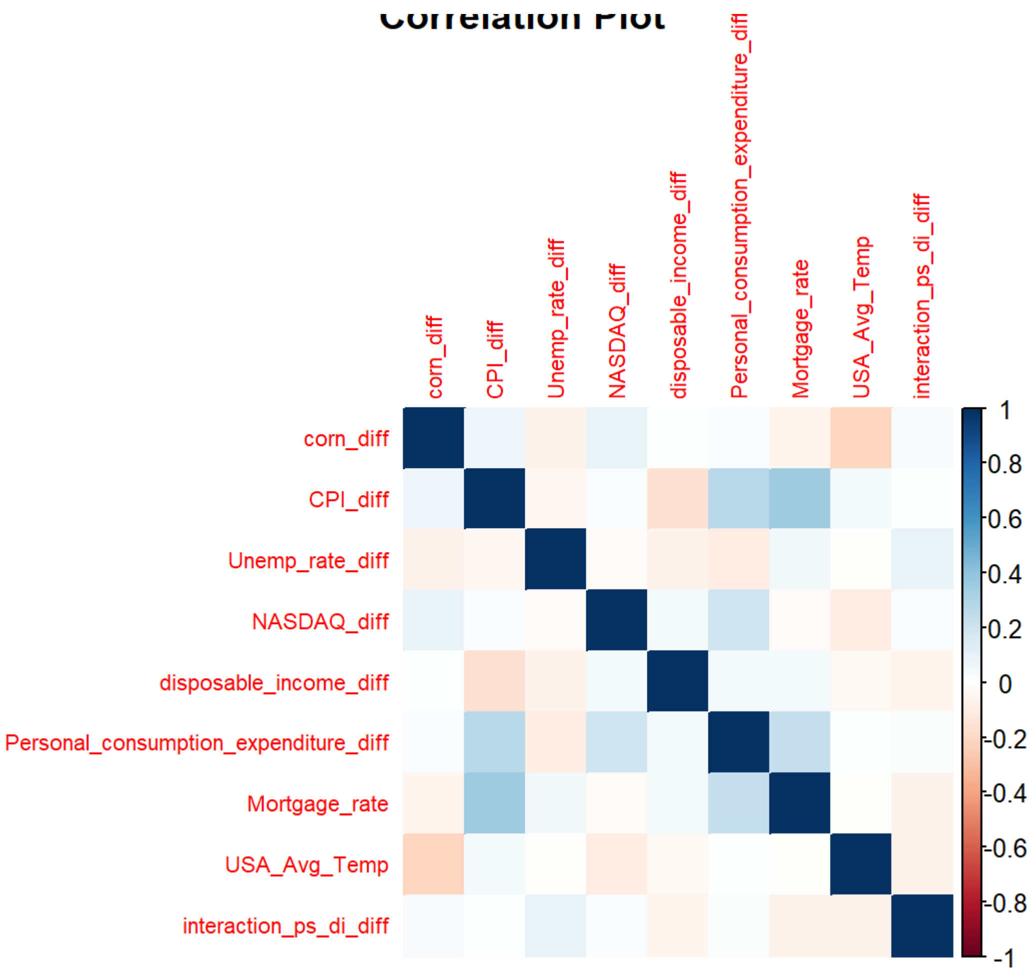
```
vars_stationary_cleaned <- ts_vars_stationary[-1, ]
df_clean <- cbind(vars_diff, ts_vars_stationary) #merge the transformed data with the stationary
data
str(df_clean)
```

```
## 'data.frame':    423 obs. of  9 variables:
##  $ corn_diff                          : Time-Series  from 1981 to 2016: -0.01078 0.04504 -
0.05868 -0.00828 0.01648 ...
##  $ CPI_diff                           : Time-Series  from 1981 to 2016: 0.0093 0.00922 0.00
913 0.0068 0.00563 ...
##  $ Unemp_rate_diff                    : Time-Series  from 1981 to 2016: -0.0408 0.0408 -0.0
134 0 -0.0274 ...
##  $ NASDAQ_diff                        : Time-Series  from 1981 to 2016: -0.01146 0.00211 -
0.02015 0.04388 0.05507 ...
##  $ disposable_income_diff             : Time-Series  from 1981 to 2016: 0.00467 -0.00389 -
0.0031 0.00282 -0.0028 ...
##  $ Personal_consumption_expenditure_diff: Time-Series  from 1981 to 2016: 0.013538 0.009834
0.007565 0.009876 0.000788 ...
##  $ personal_savings_diff              : Time-Series  from 1981 to 2016: -0.01739 -0.04485 -
0.00922 0 0.00922 ...
##  $ Mortgage_rate                      : Time-Series  from 1981 to 2016: 14.8 14.9 15.1 15.4
15.6 ...
##  $ USA_Avg_Temp                       : Time-Series  from 1981 to 2016: 37 30.5 31.9 38.4 4
7.9 ...
```

```
#head(df_clean)
```

Check correlation of data

```r
df_clean$interaction_ps_di_diff <- df_clean$personal_savings_diff * df_clean$disposable_income_d
iff # create interaction term
df_clean <- df_clean[, -which(names(df_clean) == 'personal_savings_diff')]
library(corrplot)
cor_matrix <- cor(df_clean)
corrplot(cor_matrix, method = "color", type = "full", tl.cex = 0.7, main= 'Correlation Plot')
```



Granger test

```r
library(lmtest) # load for granger test
```

```
## Warning: package 'lmtest' was built under R version 4.3.1
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.3.1
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
# Specify the lag orders to consider (months of lag)
lag_orders <- c(1,3,6) # test for months of lag between predictors and dependent variable

# Create empty vectors to store the outputs
Variable <- character()
Correlation <- numeric()
p_value <- numeric()


# Exclude the dependent variable from the loop
vars_to_test <- df_clean[, colnames(df_clean) != "corn_diff"]
# Loop through each variable in vars_to_test
for (col_name in colnames(vars_to_test)) {
  for (lag in lag_orders) {
  # Perform the Granger causality test
  granger_test <- grangertest(vars_to_test[, col_name], df_clean$corn_diff, order = lag)

  # Extract the p-value from the test result
  p_value <- c(p_value, granger_test$`Pr(>F)`[2])

  # Calculate the correlation with corn_diff
  Correlation <- c(Correlation, cor(vars_to_test[lag:nrow(df_clean), col_name], df_clean[1:(nrow
(df_clean)-lag+1), "corn_diff"]))

  # Store the variable name
  Variable <- c(Variable, paste(col_name, lag, sep = "_"))

  }
}


# Combine the outputs into a dataframe
granger_results <- data.frame(Variable, Correlation, p_value)
granger_results$p_value <- round(granger_results$p_value, digits = 3)
granger_results$Correlation <- round(granger_results$Correlation, digits = 3)

# View the resulting dataframe
granger_results
```

```
##                                    Variable Correlation p_value
## 1                                CPI_diff_1       0.063   0.036
## 2                                CPI_diff_3       0.098   0.117
## 3                                CPI_diff_6       0.057   0.405
## 4                         Unemp_rate_diff_1      -0.061   0.665
## 5                         Unemp_rate_diff_3      -0.082   0.686
## 6                         Unemp_rate_diff_6      -0.026   0.842
## 7                             NASDAQ_diff_1       0.096   0.331
## 8                             NASDAQ_diff_3      -0.009   0.535
## 9                             NASDAQ_diff_6      -0.051   0.826
## 10                disposable_income_diff_1       0.008   0.174
## 11                disposable_income_diff_3       0.007   0.394
## 12                disposable_income_diff_6      -0.054   0.620
## 13 Personal_consumption_expenditure_diff_1       0.024   0.559
## 14 Personal_consumption_expenditure_diff_3       0.024   0.538
## 15 Personal_consumption_expenditure_diff_6      -0.017   0.476
## 16                         Mortgage_rate_1      -0.056   0.376
## 17                         Mortgage_rate_3      -0.043   0.176
## 18                         Mortgage_rate_6      -0.024   0.045
## 19                          USA_Avg_Temp_1      -0.214   0.000
## 20                          USA_Avg_Temp_3      -0.107   0.002
## 21                          USA_Avg_Temp_6       0.190   0.001
## 22                interaction_ps_di_diff_1       0.033   0.148
## 23                interaction_ps_di_diff_3       0.044   0.005
## 24                interaction_ps_di_diff_6       0.033   0.007
```

Create the table using kable

```r
library(knitr)
library(kableExtra)
table_output <- kable(granger_results, format = "html", align = "c") %>%
  kable_styling(bootstrap_options = "striped", full_width = FALSE) %>%
  add_header_above(c('Granger Causality Test between Corn Prices and Predictors for Lage 1, 3 an
d 6 Months' =3))

# Print the table
print(table_output)
```

Granger Causality Test between Corn Prices and Predictors for Lage 1, 3 and 6 Months

Variable	Correlation	p_value
CPI_diff_1	0.063	0.036
CPI_diff_3	0.098	0.117
CPI_diff_6	0.057	0.405
Unemp_rate_diff_1	-0.061	0.665
Unemp_rate_diff_3	-0.082	0.686

Granger Causality Test between Corn Prices and Predictors for Lage 1, 3 and 6 Months

Variable	Correlation	p_value
Unemp_rate_diff_6	-0.026	0.842
NASDAQ_diff_1	0.096	0.331
NASDAQ_diff_3	-0.009	0.535
NASDAQ_diff_6	-0.051	0.826
disposable_income_diff_1	0.008	0.174
disposable_income_diff_3	0.007	0.394
disposable_income_diff_6	-0.054	0.620
Personal_consumption_expenditure_diff_1	0.024	0.559
Personal_consumption_expenditure_diff_3	0.024	0.538
Personal_consumption_expenditure_diff_6	-0.017	0.476
Mortgage_rate_1	-0.056	0.376
Mortgage_rate_3	-0.043	0.176
Mortgage_rate_6	-0.024	0.045
USA_Avg_Temp_1	-0.214	0.000
USA_Avg_Temp_3	-0.107	0.002
USA_Avg_Temp_6	0.190	0.001
interaction_ps_di_diff_1	0.033	0.148
interaction_ps_di_diff_3	0.044	0.005
interaction_ps_di_diff_6	0.033	0.007

Data Output

head(df_clean)

```
date <- vars[2:nrow(vars), 'date']
df_clean$date <- date
library(writexl)
```

```
## Warning: package 'writexl' was built under R version 4.3.1
```

```
write_xlsx(df_clean, "df_clean.xlsx")
```

END TEST FOR CAUSALITY