Comparing ARIMA, GLM, Neural Networks and Random Forest for macro economic forecasting

Robert F Frans, KWANTUMLINK LDA

22 April 2021

INTRODUCTION

Goal of the project

The purpose of this paper is to compare the predictability of different forecasting methods on macro-economic data from European countries.

Description of the dataset

The data were taken from two sources:
1) The FRED (Federal Reserve Bank of St Louis) for the macro-economic data
2) The ECB (European Central Bank) for the series of the Eurostoxx equity index

The following macro economic time series were used:

series	periodicity	unit	url
M3	Monthly	EUR	https://fred.stlouisfed.org/series/MABMM301EZM189S
CPI	Monthly	index	https://fred.stlouisfed.org/series/CPHPTT01EZM661N
GDP growth rate	Monthly	percentage	https://fred.stlouisfed.org/series/EA19LORSGPORGYSAM
LT interest rates	Monthly	percentage	https://fred.stlouisfed.org/series/IRLTLT01EZM156N
Residential prop. prices	Quarterly	index	https://fred.stlouisfed.org/series/QXMN628BIS
Employment / population	Yearly	percentage	https://fred.stlouisfed.org/series/SLEMPTOTLSPZSEUU
Unemployment rate	Monthly	percentage	https://fred.stlouisfed.org/series/LRHUTTTTEZM156S
Tot. balance of paym. growth	Quarterly	percentage	https://fred.stlouisfed.org/series/EA19B6BLTT02STSAQ
EUROSTOXX index	Monthly	equity index	https://sdw.ecb.europa.eu/quickview.do?SERIES_KEY=143.FM.M.U2.EUR.DS.EI.DJES50I.HSTA

Key steps performed

- downloading the data and reading the csv files into R
- preparing the data and turning them into a clean set, such as:
  - aligning the date formats
  - simplifying the variable names

    - joining all data into one data frame

- visualization of the data
- since they have different periodicity (frequencies): to interpolate missing data via "imputation"
- perform a check on the correlation among the variables
- apply a differencing technique on the series that are not stationary
- test univariate time series forecasting with ARIMA (Auto Regression Integrated Moving Average)
- perform further data preparation for the multivariate models
- test multivariate forecasting with GLM (Generalized Linear Modeling)
- test multivariate forecasting with a deep learning neural network model
- compare the suitability of the various models for macro economic time series forecasting

R code

This document only contains the R code snippets for generating the plots and some of the models. The full R code for the analysis of the data is in the file MacroForecastF.R.

METHODS / ANALYSIS

The following three forecasting methods were tested:

- Univariate forecasting with ARIMA (Auto Regression Integrated Moving Average)
- Multivariate forecasting with GLM (Generalized Linear Modeling)
- Multivariate forecasting with Deep Learning Neural Network
- Multivariate forecasting with the Machine Learning technique Random Forest

Rationale for the choice of the methods

1) Univariate versus multivariate We are aware that the univariate forecasting has limited capabilities, since this approach assumes that there is sufficient information present in the historic data for generating a meaningful prediction for the coming periods. Since one particular macro-economic time series is being influenced by many other macro-economic factors, it is expected that univariate forecasting will have a tendencey to extrapolate long term trends, and not be able to foresee cycles in these trends such as bubbles and crisis in the economy.

For sake of completion, it should be mentioned that advanced univariate forecasting techniques are able to recognize and reproduce a seasonal pattern into the predictions (to the extent that the seasonal pattern is detectable in the historic data).

Hence, in case of univariate forecasting, correlations will be sought inside the variable itself, by means of auto-regression, or "lagging". In other words, a variable will be compared to values of earlier periods of the same variable.

Nevertheless, better results are expected from multivariate forecasting. Since macro-economic data are interconnected, according to economic theory, a forecast of one time series based on several other time series is expected to take into account more of the causality among them. For instance, a low private consumption of private households will dampen the growth in Gross Domestic Product. Or, increasing interest rates will reduce the investments by companies, the acquisitions of houses, etc.

2) GLM versus deep learning with a neural network The GLM methods will provide weights to the interdependent variables and seek to maximize the explanation of the variations in the dependent variable.

The neural network methods will basically do the same, although models can be built with different hidden layers, which enables the neural model to detect more complex patterns, using them to yield relevant forecasts of a dependent variable.

3) Machine learning with Random Forest Random Forest is an ensemble of Decision Trees whereby the final node will be average result generated by the different trees.

Approach for comparing the methods:

Overall, the univariate forecasting methods will be considered as a form of benchmarking for the multivariate ones, in order to verify the extent of improvement of the forecasts.

The results of the different methods will be compared by means of R2 (R-squared, the square of the correlation) and RMSE (Root-Mean-Square Error), where appropriate.

Size and main characteristics of the dataset

After the aforementioned interpolation for missing data due to the differences in periodicity, the combined data set of 9 variables contains 264 monthly data. They span 22 years, from Jan 1999 till Nov 2020.

Visualization of the time series

Due to the different nature of the series (EUR amounts, percentages, 100-base index and equity index), they are plotted in four different graphs.

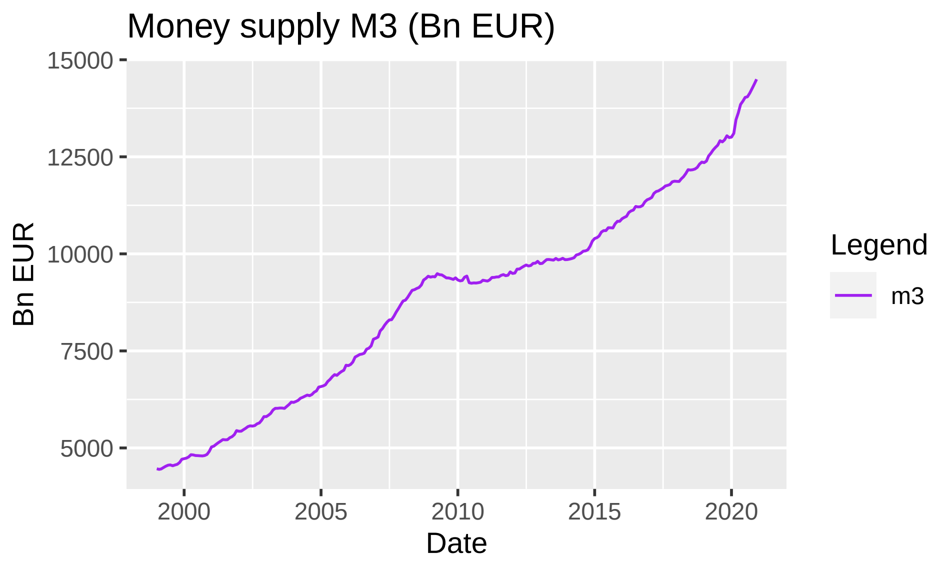
The time series that is expressed in EUR

M3 is a collection of the money supply that includes the following:
- cash and checking deposits (M1),
- 'near money', referring to savings deposits, money market securities, mutual funds, and other time deposits
- large time deposits, institutional money market funds, short-term repurchase agreements, and larger liquid funds.

The money supply normally increases following the growth of the economic activity.

Note that the M3 had a different conjecture in the crisis of 2008 than in 2020:
- in 2008 its growth slowed down due to the global economic crisis
- in 2020 it was artificially increased by "Quantitative Easing" by central banks, or in other words, by massively injecting money in the economy, in order to moderate the economic slowdown.

```
EURplot
```

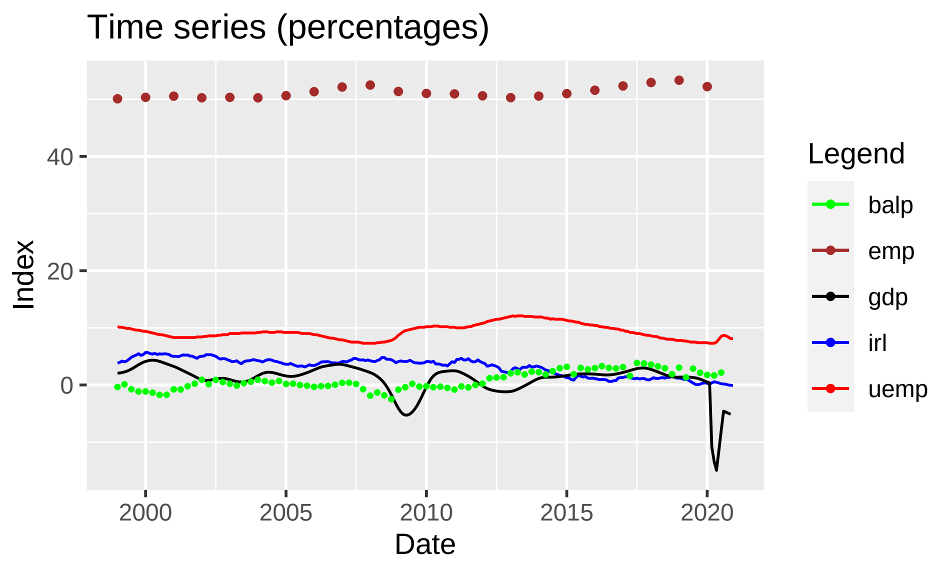


The time series that are expressed as percentages

In the following plot the following economic variables are represented :
- gdp: GDP (Gross Domestic Product) growth rate from year-to-year
- irl: the long term interest rates
- emp: the employment rate, or people employed as a percentage of total population
- uemp: the unemployment rate, or people unemployed as a percentage of the active population
- balp: the total balance of payments of the European countries towards the rest of the world. The balance of payments indicates if a country has a deficit or a surplus in assets (credits) and liabilities (debits) towards other countries.

Note that the employment times series shows up as yearly points in the plot, instead of lines, since only yearly data are available. Also, the balance of payment series shows up as quarterly points, since only quarterly data are available.

```
pctplot
```



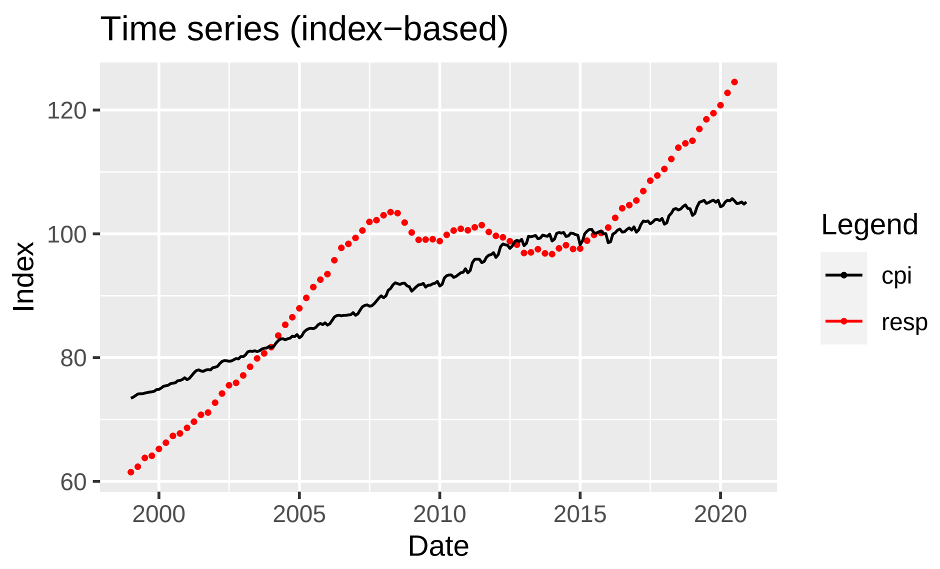
The time series that are expressed as an index

The CPI (Consumer Price Index) is expressed as an index with base 2015 = 100, and the price index of the residential houses is expressed as an index with base 2010 = 100.

It is remarkable to see how housing prices (resp) have a tendency to increase more rapidly than the general inflation (cpi), until they are corrected downwards, like in the global crisis in 2008. One can wonder if today we are not observing the same phenomenon, i.e. that the quantitative easing (increase of the money supply) would be creating the same bubble effect in the prices of residential houses, given the fact that they increase much more rapidly than the inflation in the general economy.

Note that the price index of houses shows up as quarterly points in the plot, instead of lines, since only have quarterly data available.

```
indexplot
```

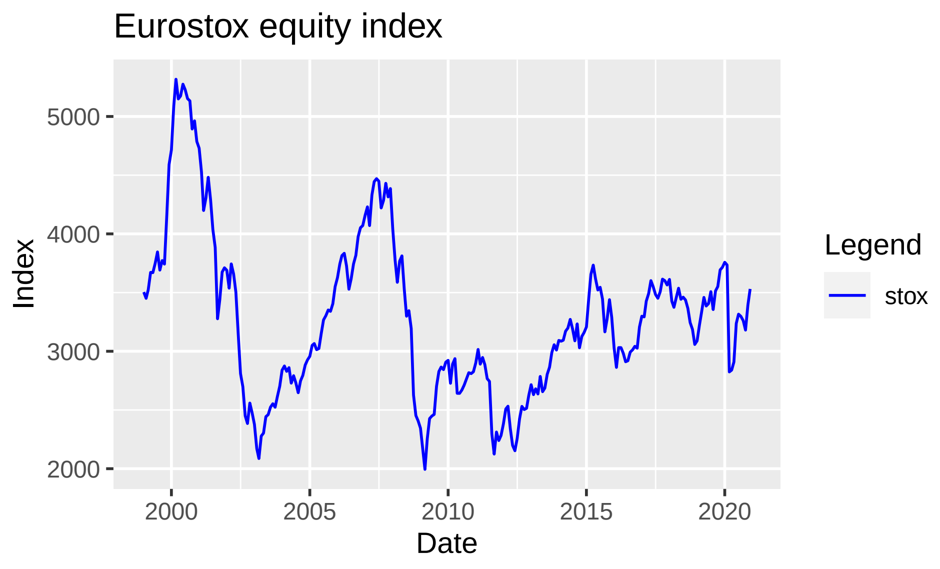


Stock exchange time series

The following plot shows the Eurstoxx equity index. This index is comparable to the Dow Jones index, but for European shares. One can identify two abrupt downfalls, one after the dot.com bubble in 2000, and after the global economic crisis (caused by the housing price bubble in the US) in 2008.

It is remarkable that the COVID crisis in 2020 just appears as a moderate and brief decline, with a recovery already starting before the economic difficulties ended. It is normal to expect from the stock exchanges that they anticipate future events, but in this case - again - the quantitative easing may have channeled too much liquidity to the stock markets instead of to the general economy, creating a “stock market inflation”, or “bubble”.

```
stoxplot
```



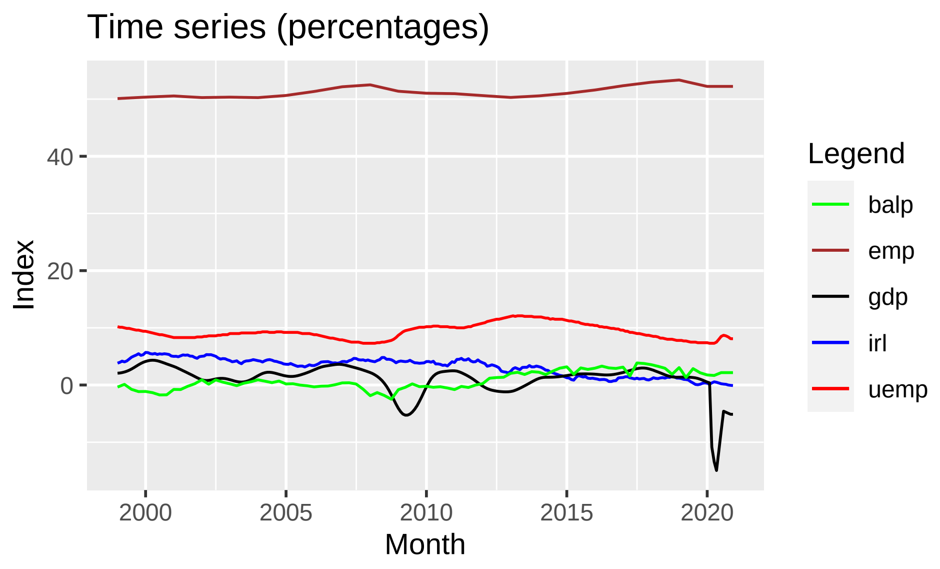
Intrapolation

Since the employment series is yearly and the prices of residential property and the balance of payments series are quarterly, we will interpolate new data via the function 'imputeTS', so that they can be treated in combination with the other series that have a monthly frequency.

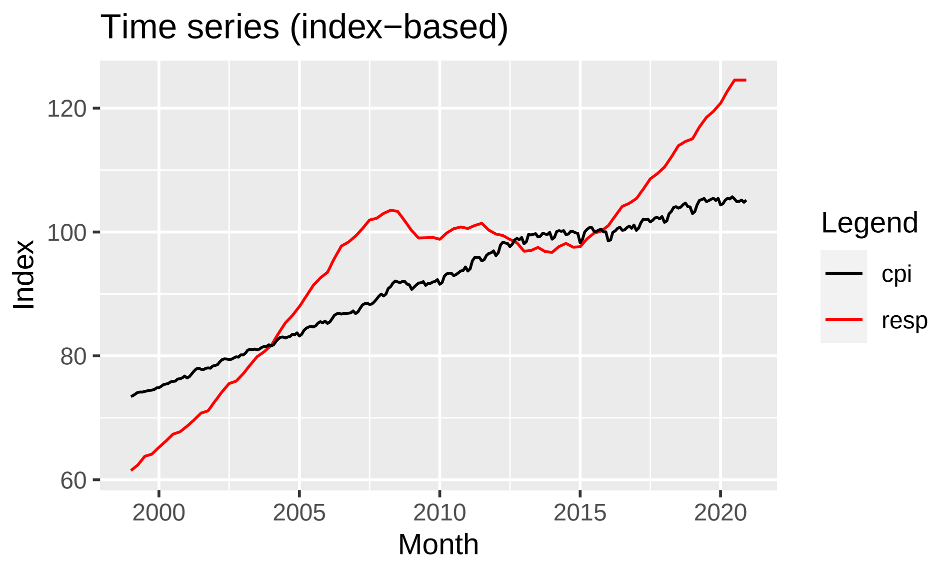
Remark: The gdp series is monthly and thus does not need interpolation, but we used the same imputation formula to complete the series with one missing value at the end. In practice, when applying the imputation formula, the model simply came up with a repetition of the previous value, which is a "naive" fill-in. Nevertheless, filling in this missing value enabled us to extend the multivariate analysis by one last period in our modeling, since all other series had a figure available for that last period.

Here follow the two plots with the data interpolated:

```
pctIplot
```



```
indexIplot
```



Applying a differencing technique on the non-stationary series

When comparing the statistical significance of different models, it is recommended to work with stationary data, i.e. data that do not have a "trend".

For series that grow over time, these trends can be taken out by replacing the series by the differences between one period and the next. This "differencing" technique will be applied to three series in our dataset: the M3 (money supply) data, the CPI (consumer price index), and the RESP (house prices index).

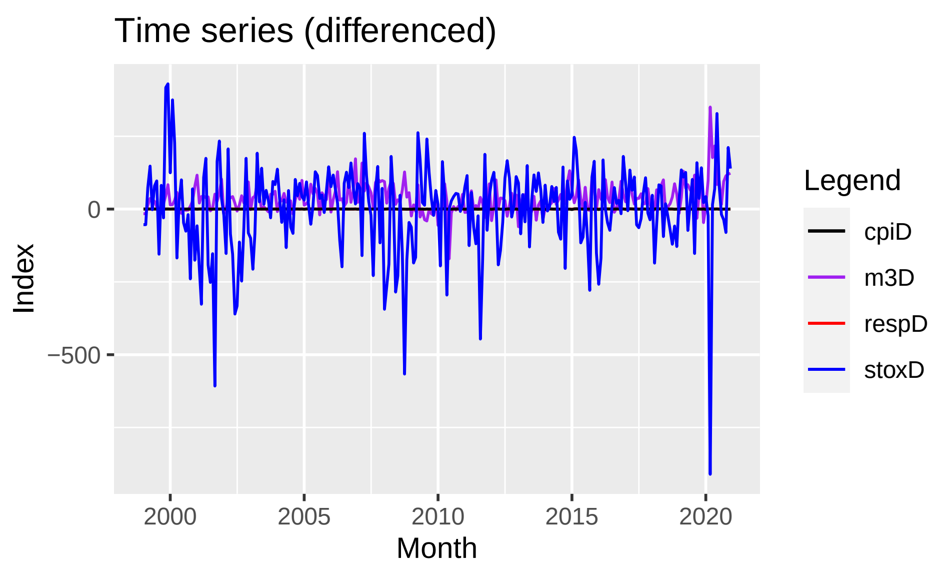
It should be noted that a model with a dependent variable that is expressed as a nominal level (CPI) will always have very high R Square and low Standard Error. However, such models are heteroskedastic by definition.

This is a problem for several reasons:
- we cannot measure accurately the statistical significance of any of our variables because the variance around any of their regression coefficients is not stable enough.
- as a result the estimated confidence interval will be incorrect.
- also, such model's residuals would not be randomly distributed.

Replacing the nominal data by the change from one period to another eliminates heteroskedasticity and autocorrelation of residuals. However, such a model will typically have a far lower R Square and a higher (relative) standard error. Yet, this approach does not break the underlying assumptions of linear regression, and facilitates the comparison between different techniques.

Here below the the series that were "differenced" are plotted again. One can see that now they remain stationary around zero.

```
differencedplot
```

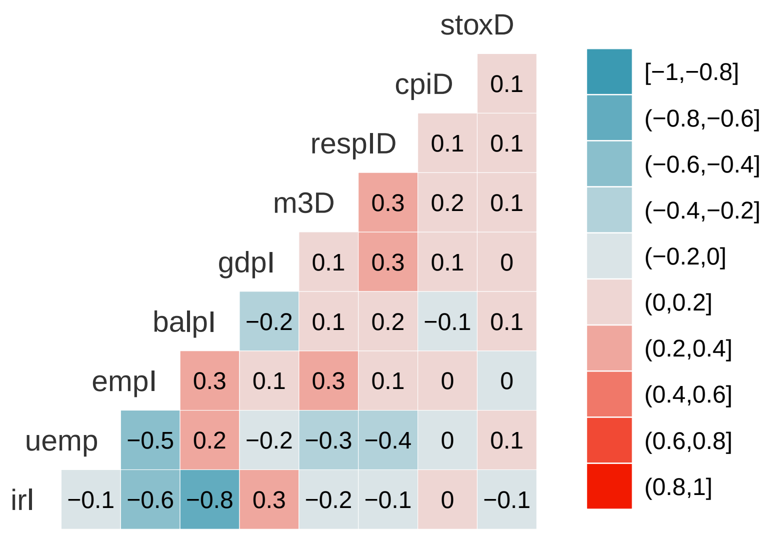


Correlation check

One of the conditions precedent for regression techniques such as GLM is that the independent variables are not significantly correlated among each other.

This can be verified by plotting a heat map, containing the coefficient of correlation computed with the Spearman method:

```
corrmap
```



The differenced times series are indicated with a “D” at the end, such as m3D, cpiD and stoxD.

Conclusion from the heat map is that there is no systematic inter-correlation between any pair of variables. The correlations are indicated by the dark colors (red for positive correlation, blue for negative correlation), and there is no entire column or entire row that is predominantly colored in dark red or dark blue.

MODELING RESULTS AND PERFORMANCE

1) UNIVARIATE TIME SERIES FORECASTING WITH ARIMA (APPROACH 1):

Explanation about the ARIMA method [1]

ARIMA (Auto Regression Integrated Moving Average) is a forecasting technique based upon the autocorrelations in the data. The term autoregression indicates that it is a regression of the variable against itself.

An ARIMA model is qualified by three parameters: p,d,q.
p = order of the autoregressive part;
d = degree of first differencing involved;
q = order of the moving average part.

Once the model order has been identified (i.e., the values of p, d and q), we need to estimate the parameters of the independent variables. When R estimates the ARIMA model, it uses maximum likelihood estimation (MLE). This technique finds the values of the parameters which maximize the probability of obtaining the data that we have observed. For ARIMA models, MLE is similar to the least squares estimates.

Optimizing via the use of the auto.arima function

The auto.arima() function in R uses a variation of the Hyndman-Khandakar algorithm (Hyndman & Khandakar, 2008), which combines unit root tests, minimization of the AICc (Akaike's Information Criterion) and MLE to obtain an ARIMA model.

Hyndman-Khandakar algorithm for automatic ARIMA modeling is as follows:
1. The number of differences 0 <= d <= 2 is determined using repeated KPSS tests.
2. The values of p and q are then chosen by minimizing the AICc after differencing the data d times. Rather than considering every possible combination of p and q, the algorithm uses a step-wise search to traverse the model space.

a. Four initial models are fitted:
   o ARIMA(0,d,0),
   o ARIMA(2,d,2),
   o ARIMA(1,d,0),
   o ARIMA(0,d,1).
   A constant is included unless d=2. If d<=1, an additional model is also fitted:
   o ARIMA(0,d,0) without a constant.

b. The best model (with the smallest AICc value) fitted in step (a) is set to be the "current model."

c. Variations on the current model are considered:
   o vary p and/or q from the current model by ± 1;
   o include/exclude step c from the current model.
   The best model considered so far (either the current model or one of these variations) becomes the new current model.

d. Repeat Step 2(c) until no lower AICc can be found.

Application of the auto.arima function

Since auto.arima already includes an optimized differencing technique, we will apply the forecast on both series, i.e. on the CPI index (cpi) and on the CPI differenced data (cpiD).

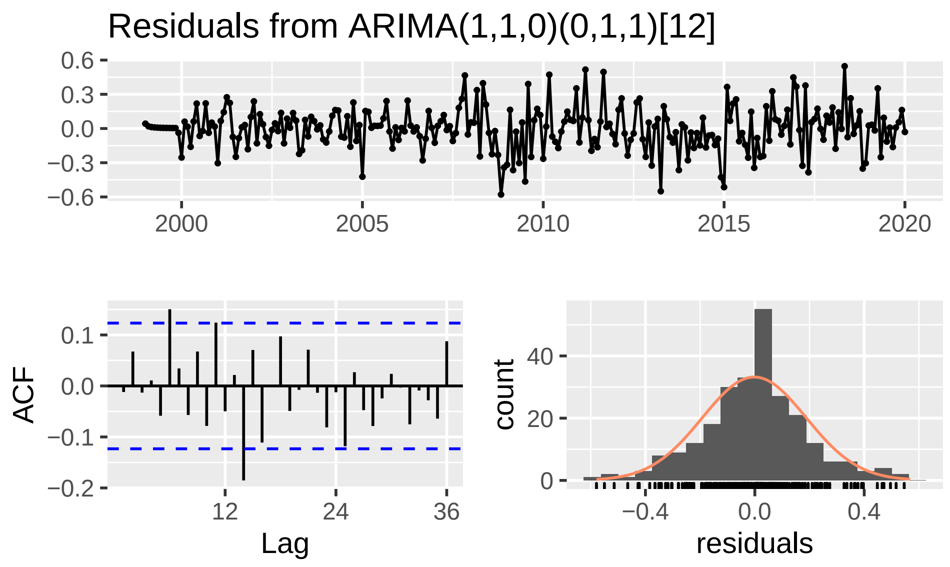
a) Application on the CPI series

The following plots from the function 'checkresiduals()' enable to verify whether the model has left any patterns in the residuals or whether these residuals are 'white noise'. It can be seen from the residuals and from the ACF (Autocorrelation plot) that there is no systematic pattern left in the residuals and that there is no need for further optimizing the model.

```
fit_cpi$aic
```

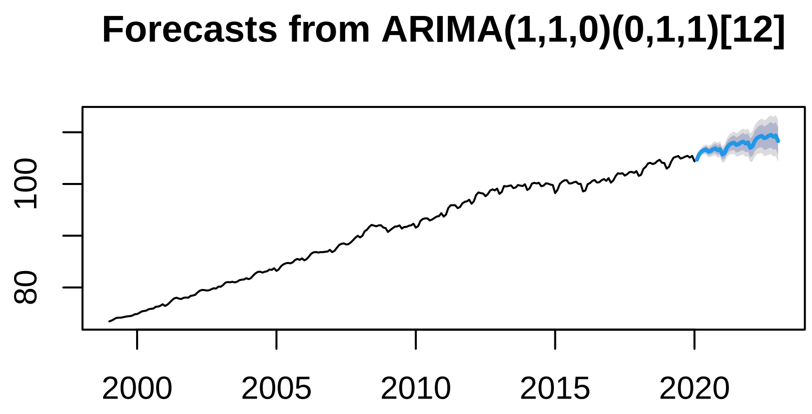
```
## [1] -95.19705
```

```
checkresiduals(fit_cpi)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(0,1,1)[12]
## Q* = 37.549, df = 22, p-value = 0.02057
##
## Model df: 2.   Total lags used: 24
```

```
plot(fcast_cpi)
```



The auto.arima model is basically extrapolating the increasing trend in the series, but on the other hand, the model demonstrates that is capability to reproduce a seasonal pattern in the data. The grey areas indicate the confidence intervals. As expected, they are expanding the further the forecasting period advances, due to the fact that the reliability of the predictions is decreasing, and thus the confidence interval increasing.

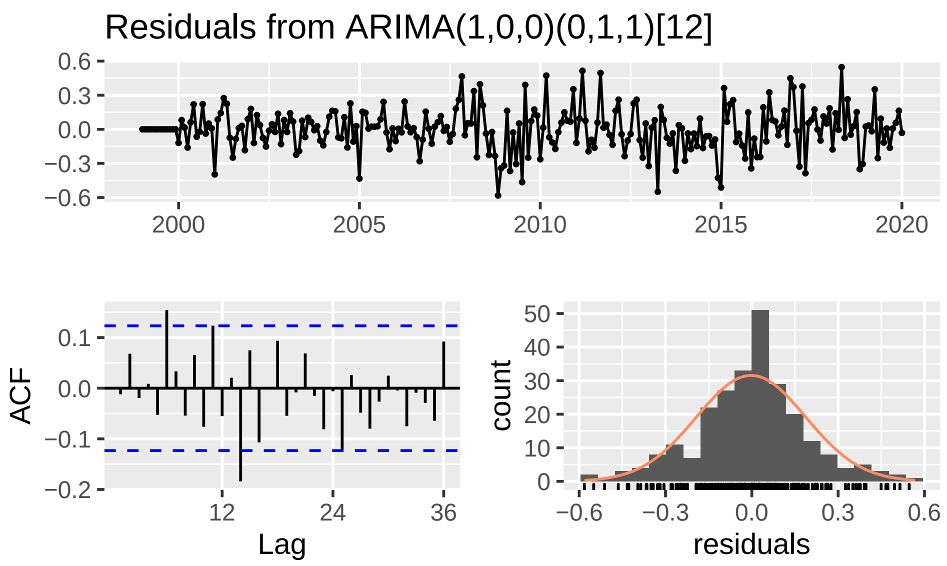
b) Application on the CPID series

As in the previous model, the plots from the 'checkresiduals()' function do not show any remaining patterns in the residuals after applying the model.

```
fit_cpiD$aic
```

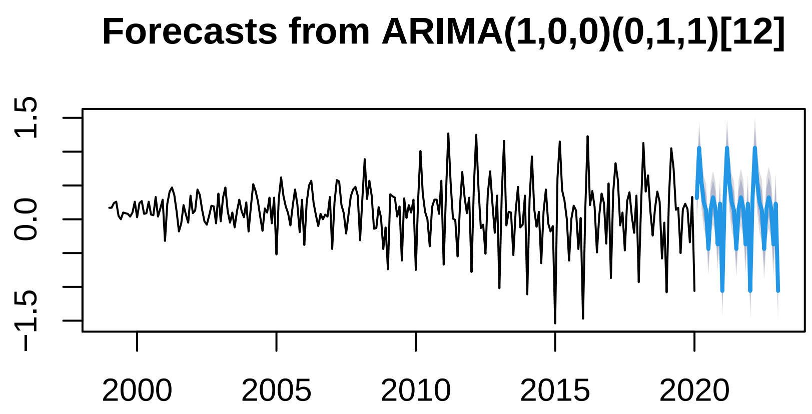
```
## [1] -94.63056
```

```
checkresiduals(fit_cpiD)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(0,1,1)[12]
## Q* = 37.201, df = 22, p-value = 0.02248
##
## Model df: 2.   Total lags used: 24
```

```
plot(fcast_cpiD)
```



Again, the auto.arima model reproduces the seasonal pattern from the historical data into the 3 year forecasts.

2) MULTIVARIATE FORECASTING APPROACH

2.1) INTRODUCTION

The CPI (Consumer Price Index) is an interesting economic indicator to forecast, because it has many side effects on the economy and the financial world. In the multivariate forecasting, the CPI will be used as a dependent variable, and all other variables will be used as independent variables.

2.2) FURTHER PREPARATION FOR THE MULTIVARIATE MODELING

a - Adding a variable for the external events not included in the series

Some preliminary tests with the GLM (not detailed here) revealed that the model had difficulty with the 2 crisis periods of 2008 and 2020.

Although the decline in GDP could have helped to identify these crisis periods, the model did not manage to handle these periods well, because they were severe deviations from the normal situation, caused by "external" events that were not included in the time series used in the model.

Therefore we will try to improve the model by adding a variable that reflects the drastically different general economic climate in these two periods of 2008 and 2020.

Adding such variable is obviously somewhat subjective, but is expected to help in distinguishing 'normal' fluctuations in the macro-economic data, and the potential fluctuations in CPI due to these special events, that were external to the economic situation in Europe.

In practice, a "negative climate" indicator is inserted for the following periods:
- 2008-08 till 2009-12: the global crisis triggered by the housing bubble in the USA
- 2020-03 till 2020-12 (=end of the series): the lockdowns due to the COVID pandemic

The negative climate is represented by a value '-1', while all other periods will receive a value '+1'.
A choice was made to consider the economic climate in 2000 as "normal" or "+1", since the crisis was limited to the "dot.com bubble" on the stock exchanges, and did not affect the global economy in the same way as the other crises did in 2008 and in 2020.

b - Creating a leading variable for CPI

Since we will test the predictive capacity of the data series on future CPI (Consumer Price Index), the simplest way to include this in a GLM model is to add variables for 'cpi of year+1', cpi of 'year+2' and 'cpi of year+3'.

Therefore, 3 new columns will be created with the cpi data, by shifting the cpi data 1, 2, and 3 years backwards. They are called cpi1, cpi2 and cpi3.

2.3) MULTIVARIATE GLM WITH ALL INDEPENDENT VARIABLES (APPROACH 2a)

a) application on the CPI series

```r
modelglm <- glm(cpi ~ m3+gdpI+irl+uemp+stox+empI+respI+balpI, data=as.data.frame(dataI))
```

The overview table below shows that the predictability of the data series on the CPI is very high, with a R2 of 0.988. Adding the climate variable helps to further improve the R2 to 0.991.

Method	Dep_var	R2	RMSE
GLM with 8 vars	cpi	0.9880912	1.0695560
GLM with 8 vars + climate variable	cpi	0.9907205	0.9441266

b) application on the CPID series

```
modelglm <- glm(cpiD ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI, data=as.data.frame(dataID))
```

The overview table indicates a much lower R2, which was expected for the "differenced series" (as explained above). It also shows that the R2 is improving somewhat when the climate variable is included.

Method	Dep_var	R2	RMSE
GLM with 8 vars	cpiD	0.1018588	0.3900162
GLM with 8 vars + climate variable	cpiD	0.1053971	0.3892472

2.4) MULTIVARIATE GLM WITH THE LEADING VARIABLES OF CPI (APPROACH 2b)

As a reminder, the leading variables of the CPI are cpi1, cpi2, cpi3, or the cpi of one year ahead, 2 years ahead and 3 years ahead.

a) application on the CPI series

```
modelglm <- glm(cpi1 ~ m3+gdpI+irl+uemp+stox+empI+respI+balpI+climate, data=as.data.frame(data2NC))
modelglm <- glm(cpi2 ~ m3+gdpI+irl+uemp+stox+empI+respI+balpI+climate, data=as.data.frame(data2NC))
modelglm <- glm(cpi3 ~ m3+gdpI+irl+uemp+stox+empI+respI+balpI+climate, data=as.data.frame(data2NC))
```

As can be seen from the overview table below, the prediction of CPI is not convincing with a rather moderate R2 of 67%, and getting weaker with a longer prediction horizon of 2 years (R2 decreases further to 53%) and three years (R2 of 46%). The RMSE increases accordingly.

Method	Dep_var	R2	RMSE
Predicting CPI 1 year ahead	cpi	0.6752185	6.410040
Predicting CPI 2 years ahead	cpi	0.5299412	8.586049
Predicting CPI 3 years ahead	cpi	0.4603342	10.046516

b) application on the CPID series

```
modelglm <- glm(cpi1D ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI+climate, data=as.data.frame(data2NCD)
modelglm <- glm(cpi2D ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI+climate, data=as.data.frame(data2NCD)
modelglm <- glm(cpi3D ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI+climate, data=as.data.frame(data2NCD)
```

The overview table shows that the use of the leading variables of CPI on the "differenced series" (cpi1D, cpi2D, cpi3D) yield a reasonably high R2.

Method	Dep_var	R2	RMSE
Predicting CPID 1 year ahead	cpiD	0.8849531	0.4881526
Predicting CPID 2 years ahead	cpiD	0.8141567	0.6671047
Predicting CPID 3 years ahead	cpiD	0.7582390	0.8101879

2.5) MULTIVARIATE DEEP LEARNING NEURAL NETWORK (APPROACH 3)

a) Preparation of the data

Numeric: All data should be numeric, which is not the case for the "date" column. Since we work with stationary data (see "differencing" above), this column is not expected to have any link with the data (other than the ordering of the data in the series). Thus we can eliminate this column.

Split in training and test set: A split was randomly sampled in a proportion of 80 % training and 20 % test.

Normalization of the data: In order to facilitate the functioning of the weightings in the neural network, it is recommended to "normalize" the data, by means of centering them at zero, and standardizing their deviation. In this way the different sizes of the data elements will not hinder the calculations of the weightings in the neural network.

b) Structuring the simplest model possible

We will start to prepare a deep neural network with the most simple structure :
- an input layer with 8 nodes, one for each independent variable
- one hidden layer with 1 node !
- an output layer with one node, corresponding to the dependent variable (cpiD)

The purpose of this "simplest" model is to benchmark it with the above multivariate GLM model. Actually, a neural network with input of 8 independent variables, without any further transformation via hidden layers, and with one output variable, should be equivalent to a linear regression. Actually, what the model does is adding weights to the input variables, as well as a separate constant, just like in a standard linear regression.

The structure of this simplest model is as follows:

```
nn <- neuralnet(cpiD ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI,
                data = training,
                hidden = c(1),
                linear.output = T,
                algorithm = "rprop+",
                threshold=0.01,
                lifesign= 'full',
                rep=10)
```

c) Structuring a base model

We will then continue to prepare a deep neural network with the following characteristics:
- an input layer with 8 nodes, one for each independent variable
- one hidden layer with 8 nodes (to be further optimized afterwards)
- an output layer with one node, corresponding to the dependent variable (cpiD)

The structure of this base model is as follows:

```
nn <- neuralnet(cpiD ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI,
                data = training,
                hidden = c(8),
                linear.output = T,
                algorithm = "rprop+",
                threshold=0.01,
                lifesign= 'full',
                rep=10)
```

d) Finetuning 1: inserting an extra layer

An additional hidden layer is inserted in the model, with 40 nodes.

```
nn <- neuralnet(cpiD ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI,
                data = training,
                hidden = c(40,8),
                linear.output = T,
                algorithm = "rprop+",
                threshold=0.01,
                lifesign= 'full',
                rep=10)
```

e) Finetuning 2: adding the variable for "economic climate" to the dataset

As mentioned above, this variable, used for identifying the exceptional circumstances in 2008 and 2020, is added to the input data of the neural model.

```
nn <- neuralnet(cpiD ~ m3D+gdpI+irl+uemp+stoxD+empI+respID+balpI+climate,
                data = training,
                hidden = c(40, 8),
                linear.output = T,
                algorithm = "rprop+",
                threshold=0.01,
                lifesign= 'full',
                rep=10)
```

f) Overview table for the neural models

The overview table for the neural models indicates that the most simple model is performing well, compared to the more advanced ones. Also that the more advanced models do not seem to improve the results significantly.

It should be noted that the outcomes of the neural network calculations are very variable between one run and another. This is due to the nature itself of the training of such models, and the absence of a strict "algorithm". The neural models also had a train/test split of 80/20, which makes that the training set varies strongly from one run to another, which can impact the variability of their internal calculations between one run and another.

Neural_Model_Structure	Dep_Var	RMSE
0. input8 + dense1 + output1	cpiD	0.9150171
1. input8 + dense8 + output1	cpiD	1.0869662
2. input8 + dense40 + dense8 + output1	cpiD	1.5608203
3. input8 + dense40 + dense8 + output1, with climate var	cpiD	1.2939652

2.6) MULTIVARIATE MACHINE LEARNING WITH RANDOM FOREST (APPROACH 4)

a) Preparation of the data We will use the same data as were prepared for the neural network approach above. No specific preparation is required.

b) Explanation about the Random Forest method [4] Random Forest generates an ensemble of Decision Trees. In a regression problem, as is the case in our model, the final outcome will be the average of the outcomes of each tree. (Alternatively, Random Forest is often used for classification problems, or categorical outcomes, whereby each tree "votes" for its output, and the majority-voted output is taken.)

A random sample of rows from the training data will be taken for each tree. From this sample, a subset of features will be taken to be used for splitting on each tree. Each tree is grown to the largest extent specified by the parameters until it reaches an outcome.

The fundamental reason to use a random forest instead of one decision tree is to combine the predictions of many decision trees into a single model. The logic is that an average, even made up of many mediocre models, will still be better than one good model. There is truth to this given the mainstream performance of random forests. Random forests are less prone to overfitting because of this.

Overfitting can occur with a flexible model like decision trees where the model with memorizing the training data and learn any noise in the data as well. This will make it unable to predict the test data.

A random forest can reduce the high variance from a flexible model like a decision tree by combining many trees into one ensemble model.

c) Application of the model From the table below it can be seen that Random Forest performed well in predicting cpiD, and that the addition of the climate variable also contributed to a lower error rate.

Random_Forest	Dep_Var	RMSE
Random Forest, n=20	cpiD	0.9646013
Random Forest, n=20, with climate variable	cpiD	0.9470953

CONCLUSION

Several conclusions can be drawn from these tests with different forecasting models. Some of them can be considered as confirmations of generally known phenomena.

- Univariate time series forecasting has the tendency to extrapolate trends. In order to detect cycles, sufficient data should be available in the series in order to reproduce them. Otherwise this type of forecasting will simply extend an existing long term trend.
- Multivariate time series forecasting have the benefit of using the correlations between series and can therefore generate more sound results. Nevertheless, abnormal swings caused by "external" events cannot be captured by such models. A possible technique is the manual addition of extra information on these events.
- When comparing GLM and neural networks for time series forecasting, the multivariate linear regressions yield a more "stable" model than the neural networks. For predictions of time series (i.e. predicting non-categorical values), different runs of the same model yield variable results, making it "unstable" to conclude. However, this does not mean that neural networks would not be capable of detecting complex patterns in the time series in a more advanced way than a linear regression technique, but at least this was not observed with the data set used in this paper.
- Several attempts to optimize the structure of the deep learning neural network, did not yield better results. At least on the used data set, the more simple models seemed to perform better than models with more layers and more nodes.
- The machine learning technique Random Forest yielded a satisfactory result, comparable to the simple neural model. However, Random Forest has the advantage of being more efficient in terms of use of calculating power.

SUGGESTIONS FOR FURTHER RESEARCH

- This comparative study could be deepened with more data, for example on a country basis, rather than on a European "aggregate" basis. It is possible that by aggregating all countries, the quality of the "patterns" in the data got diluted, and that they would be more recognizable with data per country.
- The application of the neural network was limited to sequential layer models; further sophistication via model engineering could yield improved results.

BIBLIOGRAPHY

1. 'Forecasting: Principles and Practice (August, 2018, 2nd Ed.)', Rob J Hyndman and George Athanasopoulos, Monash University, Australia, https://otexts.com/fpp2/arima-estimation.html

2. 'Time Series Analysis with Auto.Arima in R', Luis Losada, (2020), https://towardsdatascience.com/time-series-analysis-with-auto-arima-in-r-2b220b20e8ab

3. 'Neuralnet: Train and Test Neural Networks Using R', (sept 2019), Datascience+, https://datascienceplus.com/neuralnet-train-and-test-neural-networks-using-r/

4. EXXACT Blog, (2020), https://www.exxactcorp.com/blog/Deep-Learning/3-reasons-to-use-random-forest-over-a-neural-network-comparing-machine-learning-versus-deep-learning