University of St.Gallen

Introduction to Programming: Group Project

Niklas Halfar (20-617-361)

Florian Lerf (20-605-127)

Skill: Introduction to Programming

University of St. Gallen

Dr. Mario Silic

26. December 2021

1 Project Idea

As economists we spend a lot of time studying past economic data to gain insights, to shape a better economy in the future. In the business world there is always the past performance records to analyse, however we are usually more concerned with how we can gain a competitive advantage in the future. So for our project we thought let's try to use programming and specifically statistical programming to forecast some economic data. As we both have been learning the programming language R in this as well as in some other courses, we thought it would be best to do our project in R to deepen our skills in this language.

The problem we want to solve is forecasting future personal consumption expenditure in the US economy. We found a clean dataset which contains the past consumption as well as other features like population, unemployment and savings rate from 1967 until 2015[1]. We will use this dataset, enhance it with additional features and visualize it before trying to build different forecasting models and evaluating the results we get.

2 How to run the project

We made sure to comment all of our code so this paper is not needed to understand our project. This paper is only for providing some more background to the subject matter!

Our project is uploaded into our GitHub Repository (https://github.com/GitGetGotUp/Coding-HSG_Project) in a single R script file called economicsv6.r. Please load the file into your local RStudio or run it on Nuvolos (the online IDE provided by HSG). When you run the script please make sure to install all the libraries. This has to be done only once. They are listed in chapter 0 in our script. After installing all libraries, you have to call them into the RStudio session through the library(…) command every time you run the script. This is just below the installing of the libraries in chapter 0 in the script. You can then go through our script, with Ctrl+Enter on Windows or Cmd + Enter on Mac. This command always executes one line, so you can go through the script step-by-step. The necessary models and outputs will all be stored in the R environment and once all is run can be accessed again in the environment. There is no user input or anything, so you just have to click through and can enjoy the plots we coded and see the error metrics for our forecasting models.

3 Data Cleaning and processing

For this project we use three datasets. The first and most important one is the economics data set from the tidyverse library. We convert it to a tsibble. Tibbles are common R objects, a tsibble is the same but for a time series, having a time index, which we specified as the month because the data was recorded monthly. We then rename the columns to easier identify the features. As some of the data is recorded in billions of dollars, others in thousands, we adjust accordingly to get the normal value. We then also divide the consumption by population as the population was growing rapidly and a per capita analysis makes more sense for potential comparison with other data and countries.

Our target variable Consumption is displayed in monetary value. However as we have a time series of over 50 years, we cannot compare the value of 2000 dollars from 50 years ago with todays dollar value. We have to account for inflation. So from the tsibbledata library we took the global_economy

[1] https://ggplot2.tidyverse.org/reference/economics.html

dataset, which has the CPI (Consumer Price Index) of every country in it. We filter for the US and ,using the date variable as the key, we join the CPI column to our original economics dataset. We then adjust all monetary variables for inflation by dividing through the CPI and multiplying with 100. Now the dollar values are comparable over time because we adjusted for inflation.

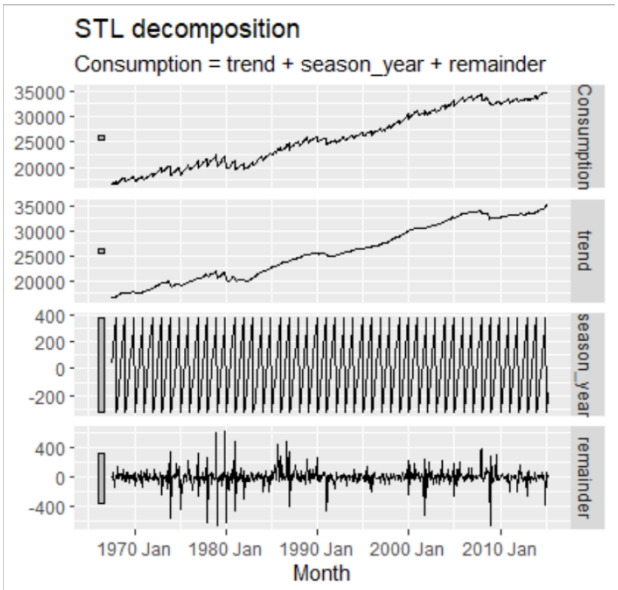
Additionally we then downloaded data from the FRED St. Louis[2] in csv format and added it into our script. This dataset is also stored in our GitHub repo and our Rscript will directly call it up from there using the URL. We then added the column where the recession was binary encoded to our dataset. Now we have our complete dataset, which we will call df_ts in the script.

Now we have prepared a dataset, which is clean and useable for the analysis. In the next step we did a exploratory data analysis. This is done to get a feel for the data and spot patters.

4 Time Series analysis

In this next step we did some exploratory data analysis. This is mostly just plotting the time series and getting familiar with it. We find that Consumption has seen a strong upwards trend, and has some seasonality to it. This might be because spending is higher in summer months. Secondly the savings rate is very volatile. We know this from economic interpretation as households try to have a steady consumption whereas investments and subsequently savings are very volatile and depend a lot on the economic conditions of the time. The unemployment follows boom and bust movement with undefined time duration.

Because our time series has so many patters which we cannot spot from the plots alone, we do a decomposition to make it easier for us to do the analysis later. We will decompose the time series in trend, season and remainder. The Decomposition is made up of these three components. The season is mostly over a month or a year, the trend is the underlying direction and the remainder is the part which cannot be explained by the other two factors. We are doing a decomposition using LOESS (STL), which is a method especially good for estimating non-linear relationships.

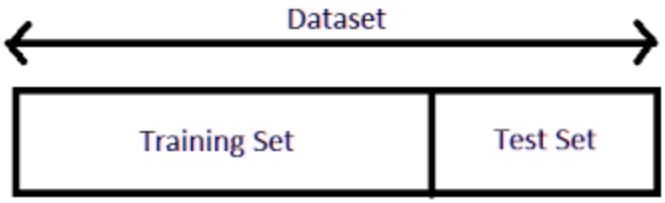


Here you can see our decomposition of the Consumption time series.

[2] https://fred.stlouisfed.org/series/USREC

5 Forecasting using models

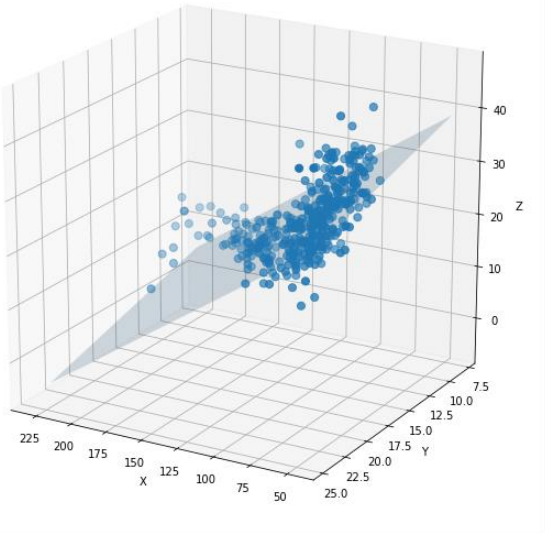
In the forecasting part the first thing we do is to split the data as seen in the picture.



We split it into Training Set from 1967-01-01 till 2012-12-01 and 2013-01-01 till 2015-04-01. So we can train the models on the Training Set and then test it on the Testing Set. Our model takes the features of the Testing Set and predict a value. Then we take the difference to the actual value in the Testing Set. This difference is the error of our model. A perfect model would have an error of 0. We try to find the one model which has the lowest error. The error itself is calculated as the Mean Squared Error (MAE). This is the absolute difference between predicted value and actual value and then the average over all errors from 2013-01-01 till 2015-04-01.

First we use very «dumb» models. Why are they dumb? Because for these models you would not need a data scientist. A mean model predicts that the value of the Consumption is equal to the average value of the Training set. Intuitively this does make sense, as the average is generally speaking a good prediction. A Naïve model is when we predict that the consumption value is just the same as last month. And a seasonal Naïve is when we predict that the consumption value is the same as in the last season. The mean and the Naïve model have a very large error in our script and we will not discuss them further. The seasonal Naïve is the best with a MAE of 189 dollars when predicting the remainder. But how good is an error of 189 dollars? This remainder has a mean value of -17. So our MAE of 189 is very large in comparison and the model is not able to forecast the values in a useable manner. When we use the whole time series by adding the trend and season to the consumption, our model does better and has a MAE of 538 dollars. The nominal error is larger but because we are predicting values that are around 34000 dollars, our model is only a little bit off in comparison with 538 dollars on average.

Then we moved on to linear regression. In linear regression the model uses the features (unemployment, recession, savings rate etc.) we have in our data set and tries to predict consumption with it. In a 3D space it would look like the picture on the right:



The features are points in a coordinate system. We try to fit a plane into the space. This plane is fitted using the minimization of a loss function. This loss function in our case is the mean squared error function. The plane is constructed so that the squared distance of the plane to each datapoint is minimized. It uses coefficients which it puts in front of each feature to fit the plane and get the prediction. Generally speaking for the i-th feature our linear equation looks like this (for three features X_1 to X_3):

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \beta_3 X_{i3}$$

Now in our case we have 7 features therefore we have seven coefficients (beta). In higher dimensions the visualization is not possible but the mathematics remain the same. The model trains the coefficients beta 1 to 7 with Machine Learning on the training dataset so that the generated line has the minimal distance to the actual value points. We then take the first row in the testing dataset and multiply each feature with the corresponding coefficient and the resulting Y is our prediction. The difference is our error and the absolute average over all of the errors in the Testing Set is our MAE. As features we give our model Population, Savings rate, Unemployment Duration, Unemployed, recession (yes or no), trend and season. Note that our model knows the past trend and how the seasons behave. When we test it we then have an average value of 25785 and an absolute error of 94. This means our linear regression model had the smallest error, as being off 94 on average values of 34000 is much better than being off 538 (the error is a smaller percentage of the to be predicted value, which is better).

The rest of the models we will not explain in detail as they are more for completeness of the evaluation. The concept however is almost the same as for the linear regression.

We also do a polynomial model, which uses the same equation as the linear regression but with polynomial terms (of degree 2). And we also fit an ARIMA model, which is commonly used in econometrics. We do not have to understand how it works, we just needed the error for comparison with our model. In a last step we also deployed Facebook's Prophet model, which is a package model, meaning we did not code it ourselves. We also just use it for comparison to the linear model.

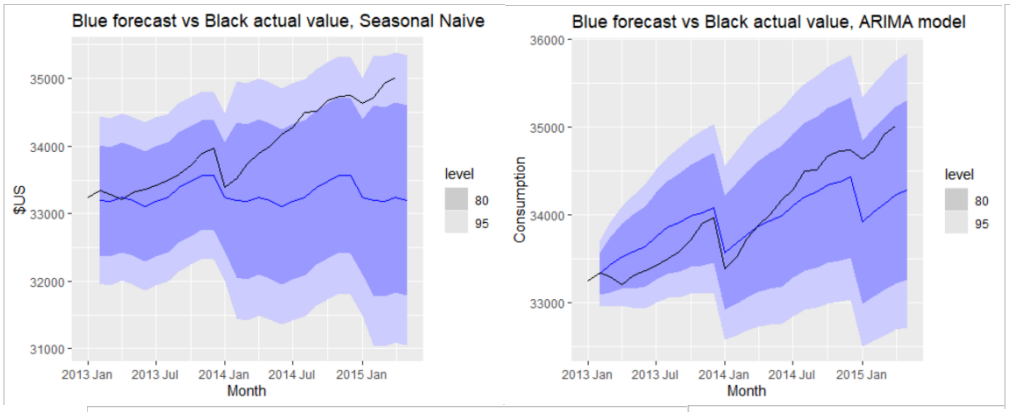
6 Evaluation of the models

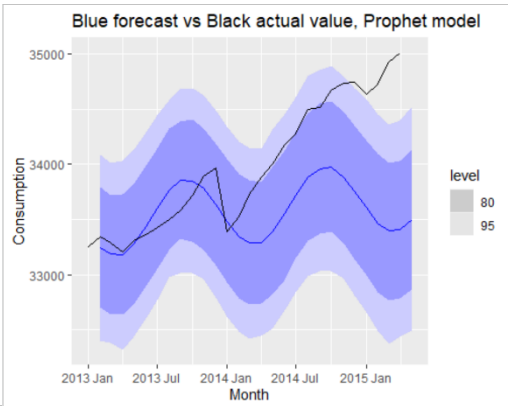
So for comparison we put all the MAE (rounded) into a table. Note that the MAE is not the only error metric used in Machine Learning. However the MAE is very easy to interpret, which is why we used it in our project.

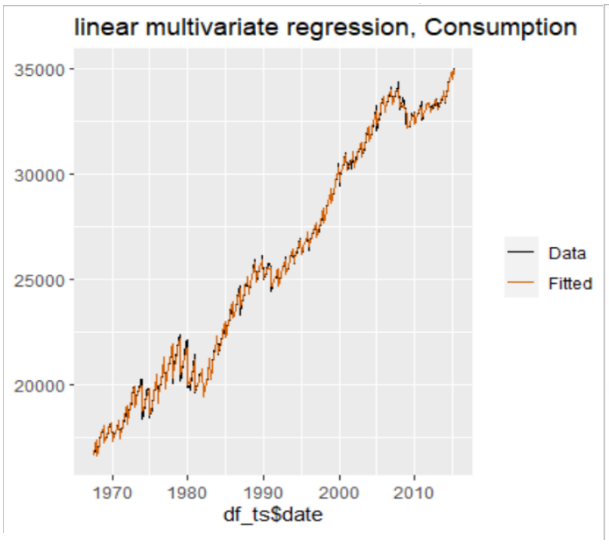
So all the models tried to predict the seasonally adjusted series. Below are the results:

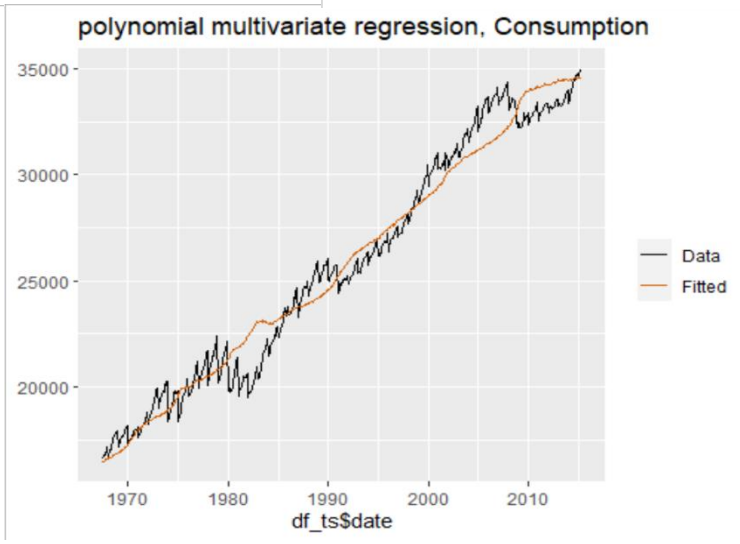
Model	MAE (error)	Mean value of the to be predicted value Consumption
Seasonal Naïve	731$	33976.22$
Linear Regression	94$	33976.22$
Polynomial Regression	848$	33976.22$
ARIMA	236$	33976.22$
Prophet	515$	33976.22$

Below you can see the graphs of our models. The blue graphs are the univariate models seasonal Naïve, ARIMA and Prophet on our Testing Set. Here the black line is what the real consumption was and the blue line is the prediction of our model. The blue and light blue interval are prediction intervals. The model thinks the value will be in those intervals with 80% confidence (blue) and 95% confidence (light blue). For the regression we plotted the whole time series not just the Testing set. The error was only evaluated on the testing set. But we wanted to plot the whole time series to show the difference between the linear model and the polynomial one. As you can see the linear model is able to fit the model very tightly and therefore decrease the error.









7 Conclusion

In conclusion we can say that our linear model performed best. It can predict the Consumption of an US Citizen per month with an error of only 94 dollars. The polynomial model was probably worse because it overfit on the training data. Because the polynomial uses quadratic terms it can fit the line better to the different feature points of the Training data. But when we give this model the Testing data which it has not seen yet, this line doesn't fit so well and has often larger errors than a straight linear line. The seasonal Naïve, ARIMA and prophet model are only univariate forecasting models. So they use only the past value of the consumption and cannot use the other features like unemployment to better their predictions. So comparisons between the models have to be made with caution.

Further there is a limitation as we did not cross-validate our models. We used a validation set approach (Training and Testing data) which is sufficient for this project. However for the results to be used further cross validation would be needed. This is challenging for time series as evaluation on a rolling forecasting origin is needed. However this would exceed the scope of this project.

With all the extra features we gave our linear model it was able to produce a really good forecast. The polynomial model which uses the exact same features was a lot worse. The linear regression in practice can often beat the more sophisticated models. In our case it is clearly superior to univariate models. We could test the model also against tree methods or neural networks, however we will leave this step for a future paper, as it otherwise would exceed the scope of this project.

8 Bibliography

Hyndman, R.J., & Athanasopoulos, G. (2021) *Forecasting: principles and practice*, 3rd edition, OTexts: Melbourne, Australia. OTexts.com/fpp3. Accessed on 25.12.2021.