exercise_part05

Forecasting: ARIMAX Model

Original Source

The standard ARIMA (autoregressive integrated moving average) model allows to make forecasts based only on the past values of the forecast variabe. The model assumes that future values of a variable linearly depend on its past values, as well as on the values of past (stochastic) shocks. The ARIMAX model is an extended version of the ARIMA model. It includes also other indepenedent (predictor) variables. The model is also referred to as the vector ARIMA or the dynamic regression model.

The ARIMAX model is similar to a multivariate regression model, but allows to take advantage of autocorrelation that may be present in residuals of the regression to improve the accuracy of a forecast.

Here use the **auto.arima** function from the **forecast** package to make forecasts withthe ARIMAX model. A function from the **lmtest** package is also used to check the statistical significance of regression coefficients.

Exercise 1

Load the dataset, and plot the variables cons (ice cream consumption), temp(temperature), and income.

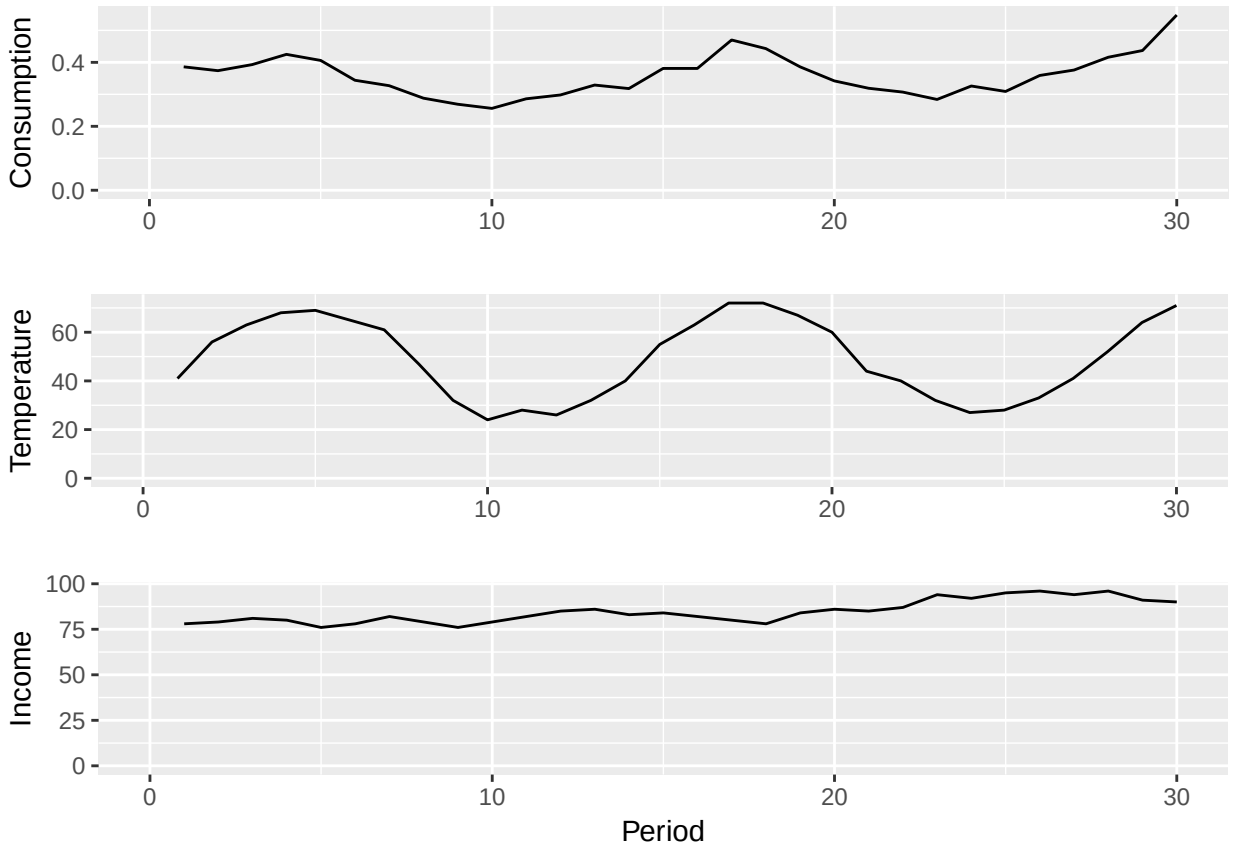
```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(gridExtra)
```

```
## Loading required package: gridExtra
```

```
df <- read.csv("Icecream.csv")
p1 <- ggplot(df, aes(x=X, y=cons)) + ylab("Consumption") + xlab("") + geom_line() + expand_limits(x=0, y
p2 <- ggplot(df, aes(x=X, y=temp)) + ylab("Temperature") + xlab("") + geom_line() + expand_limits(x=0, y
p3 <- ggplot(df, aes(x=X, y=income)) + ylab("Income") + xlab("Period") + geom_line() + expand_limits(x=0
grid.arrange(p1, p2, p3, ncol=1, nrow=3)
```



Exercise 2

Estimate an ARIMA model for the data on ice cream consumption using the **auto.arima** function. Then pass the model as input to the **forecast** function to get a forecast for the next 6 periods (both functions are from the forecast package).

```
require(forecast)
```

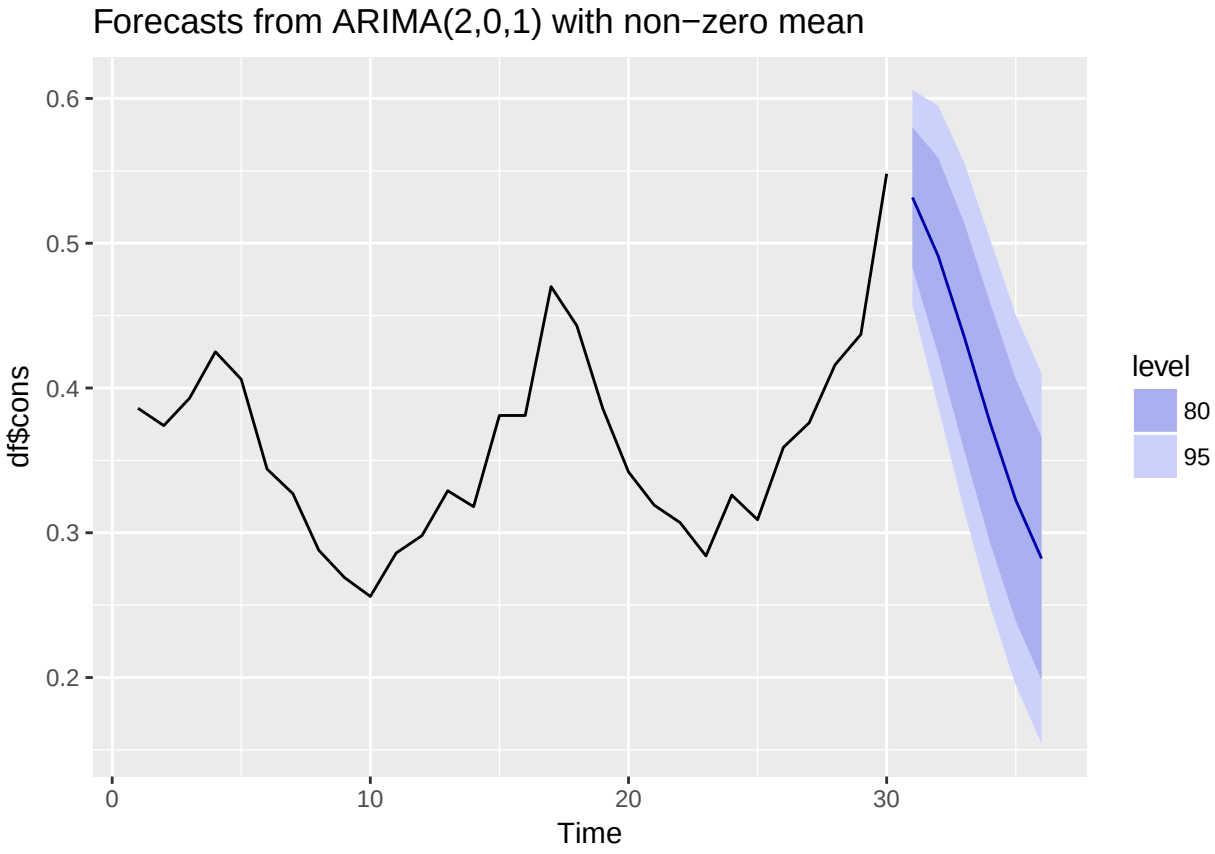
```
## Loading required package: forecast
```

```
fit_cons <- auto.arima(df$cons)
fcast_cons <- forecast(fit_cons, h=6)
```

Exercise 3

Plot the obtained forecast with the **autoplot.forecast** function from the **forecast** package.

```
autoplot.forecast(fcast_cons)
```



Exercise 4

Use the **accuracy** function from the **forecast** package to find the mean absolute scaled error (MASE) of the fitted ARIMA model.

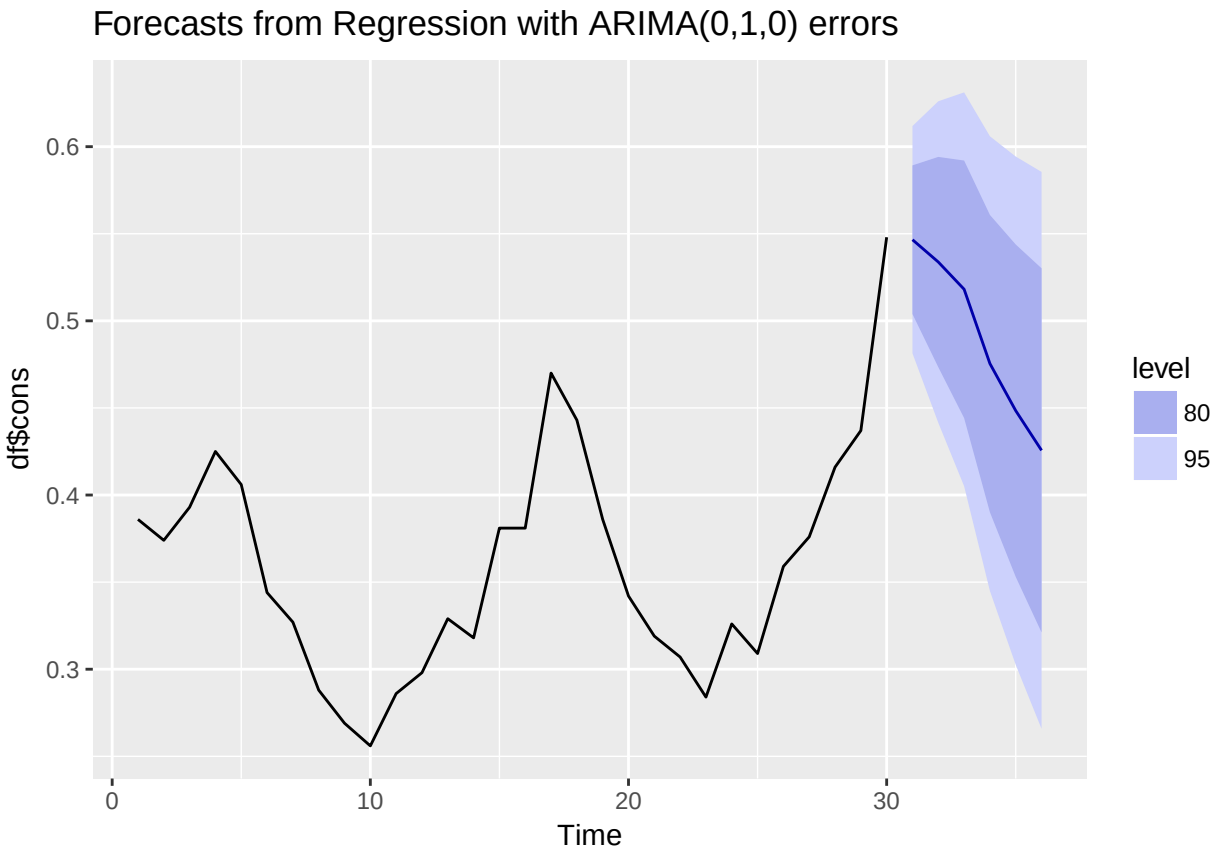
```
accuracy(fit_cons)
```

```
##                       ME       RMSE        MAE        MPE     MAPE
## Training set 0.0001020265 0.03525269 0.02692058 -0.9289076 7.203054
##                    MASE       ACF1
## Training set 0.8200598 -0.1002999
```

Exercise 5

Estimate an extended ARIMA model for the consumption data with the temperature variable as an additional regressor (using the **auto.arima** function). Then make a forecast for the next 6 periods (note that this forecast requires an assumption about the expected temperature; assume that the temperature for the next 6 periods will be represented by the following vector: **fcast_temp <- c(70.5, 66, 60.5, 45.5, 36, 28)**). Plot the obtained forecast.

```
fit_cons_temp <- auto.arima(df$cons, xreg = df$temp)
fcast_temp <- c(70.5, 66, 60.5, 45.5, 36, 28)
fcast_cons_temp <- forecast(fit_cons_temp, xreg=fcast_temp, h=6)
autoplot.forecast(fcast_cons_temp)
```



Exercise 6

Print summary of the obtained forecast. Find the coefficient for the temperature variable, its standard error, and the MASE of the forecast. Compare the MASE with the one of the initial forecast.

```
summary(fcast_cons_temp)
```

```
##
## Forecast method: Regression with ARIMA(0,1,0) errors
##
## Model Information:
## Series: df$cons
## Regression with ARIMA(0,1,0) errors
##
## Coefficients:
##       df$temp
##        0.0028
## s.e.   0.0007
##
## sigma^2 estimated as 0.001108:  log likelihood=58.03
## AIC=-112.06   AICc=-111.6   BIC=-109.32
##
## Error measures:
##                       ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 0.002563685 0.03216453 0.02414157 0.564013 6.478971 0.7354048
##                     ACF1
```

```
## Training set -0.1457977
##
## Forecasts:
##    Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## 31      0.5465774 0.5039101 0.5892446 0.4813234 0.6118313
## 32      0.5337735 0.4734329 0.5941142 0.4414905 0.6260566
## 33      0.5181244 0.4442225 0.5920263 0.4051012 0.6311476
## 34      0.4754450 0.3901105 0.5607796 0.3449371 0.6059529
## 35      0.4484147 0.3530078 0.5438217 0.3025024 0.5943270
## 36      0.4256524 0.3211393 0.5301654 0.2658135 0.5854913
```

Exercise 7

Check the statistical significance of the temperature variable coefficient using the **coeftest** function from the **lmtest** package. Is the coefficient statistically significant at 5% level?

```
require(lmtest)
```

```
## Loading required package: lmtest
```

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

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

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

```
coeftest(fit_cons_temp)
```

```
##
## z test of coefficients:
##
##          Estimate Std. Error z value  Pr(>|z|)
## df$temp 0.0028453  0.0007302  3.8966 9.756e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Exercise 8

The funciton that estimates the ARIMA model can input more additional regressors, but only in the form of a matrix. Create a matrix with the following columns: 1. values of the temperature variable, 2. values of the income variable, 3. values of the income variable lagged one period, 4. values fo the income variable lagged two period.

Print matrix. Note: the last three columns can be created by prepending two NA's to the vector of values of the income variable, and using the obtained vector as an input to the **embed** function (with the **dimension** parameter equal to the number of columns to be created).

```
temp_column <- matrix(df$temp, ncol=1)
income <- c(NA, NA, df$income)
income_matrix <- embed(income, 3)
vars_matrix <- cbind(temp_column, income_matrix)
print(vars_matrix)
```

```
##       [,1] [,2] [,3] [,4]
##  [1,]   41   78   NA   NA
##  [2,]   56   79   78   NA
##  [3,]   63   81   79   78
##  [4,]   68   80   81   79
##  [5,]   69   76   80   81
##  [6,]   65   78   76   80
##  [7,]   61   82   78   76
##  [8,]   47   79   82   78
##  [9,]   32   76   79   82
## [10,]   24   79   76   79
## [11,]   28   82   79   76
## [12,]   26   85   82   79
## [13,]   32   86   85   82
## [14,]   40   83   86   85
## [15,]   55   84   83   86
## [16,]   63   82   84   83
## [17,]   72   80   82   84
## [18,]   72   78   80   82
## [19,]   67   84   78   80
## [20,]   60   86   84   78
## [21,]   44   85   86   84
## [22,]   40   87   85   86
## [23,]   32   94   87   85
## [24,]   27   92   94   87
## [25,]   28   95   92   94
## [26,]   33   96   95   92
## [27,]   41   94   96   95
## [28,]   52   96   94   96
## [29,]   64   91   96   94
## [30,]   71   90   91   96
```

Exercise 9

Use the obtained matrix to fit three extened ARIMA models that use the following variables as additional regressors: 1. temperature, income, 2. temperature, income at lags 0,1, 3. temperature, income at lags 0,1,2.

Examine the summary for each model, and find the model with the lowest value of the Akaike information criterion (AIC). Note that the AIC cannot be used for comparison fo ARIMA models with different orders of integration (expressed by the middle terms in the model specifications) because of a difference in the number of observations. For example, an AIC value from a non-differenced model, ARIMA (p, 0, q), cannot be compared to the corresponding value fo a differenced model, ARIMA (p, 1, q).

```
fit_vars_0 <- auto.arima(df$cons, xreg = vars_matrix[, 1:2])
fit_vars_1 <- auto.arima(df$cons, xreg = vars_matrix[, 1:3])
fit_vars_2 <- auto.arima(df$cons, xreg = vars_matrix[, 1:4])
print(fit_vars_0)
```

```
## Series: df$cons
## Regression with ARIMA(1,0,0) errors
##
## Coefficients:
##          ar1  vars_matrix[, 1:2]1  vars_matrix[, 1:2]2
##       0.5672               0.0034               0.0023
```

```
## s.e.  0.1944               0.0006               0.0004
##
## sigma^2 estimated as 0.001125:  log likelihood=60.67
## AIC=-113.34   AICc=-111.74   BIC=-107.73
```

```
print(fit_vars_1)
```

```
## Series: df$cons
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       intercept  vars_matrix[, 1:3]1  vars_matrix[, 1:3]2
##         -0.1990               0.0035              -0.0003
## s.e.     0.0896               0.0004               0.0021
##       vars_matrix[, 1:3]3
##                    0.0048
## s.e.               0.0020
##
## sigma^2 estimated as 0.0009752:  log likelihood=60.96
## AIC=-111.92   AICc=-109.42   BIC=-104.92
```

```
print(fit_vars_2)
```

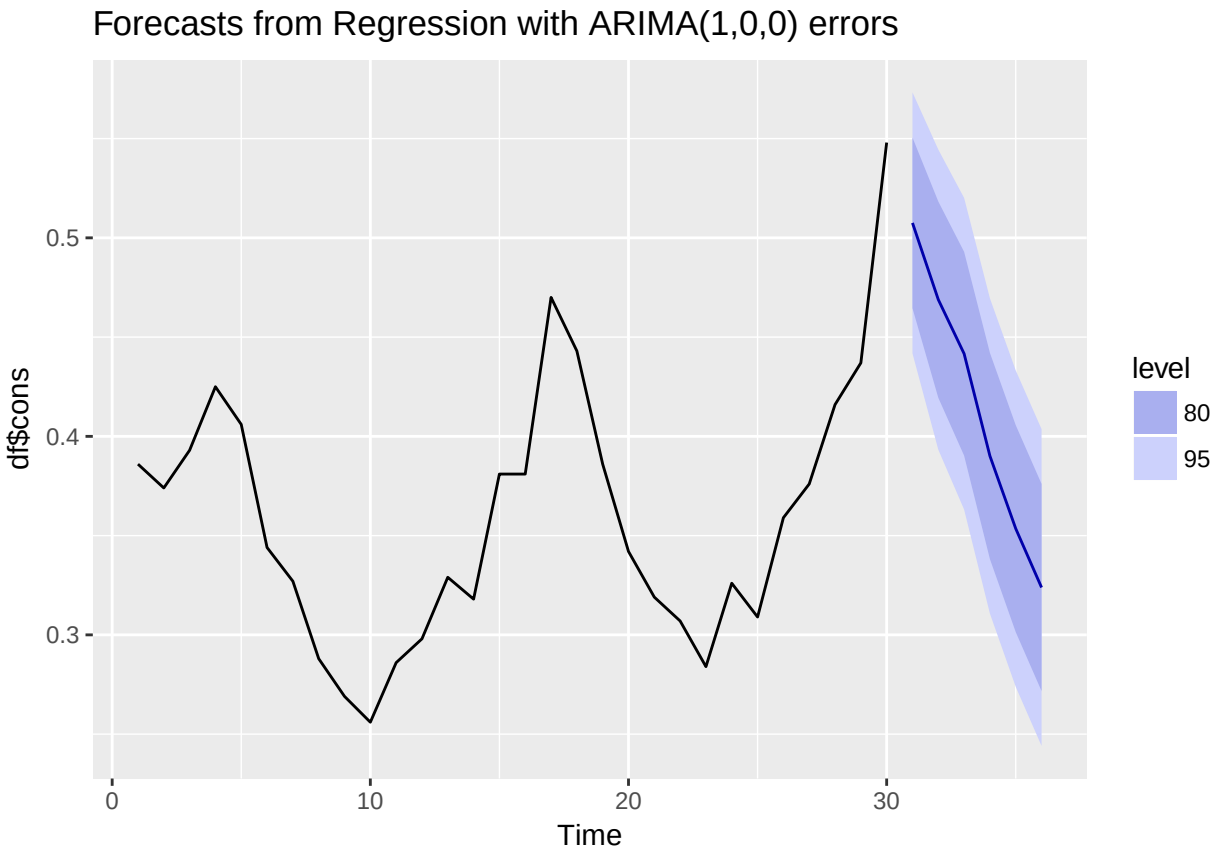
```
## Series: df$cons
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       intercept  vars_matrix[, 1:4]1  vars_matrix[, 1:4]2
##          -0.224               0.0033              -0.0006
## s.e.      0.084               0.0004               0.0019
##       vars_matrix[, 1:4]3  vars_matrix[, 1:4]4
##                    0.0012               0.0044
## s.e.               0.0024               0.0019
##
## sigma^2 estimated as 0.0008329:  log likelihood=61.12
## AIC=-110.25   AICc=-106.6   BIC=-101.84
```

The AIC can be used because the models have the same order of integration (d=0). The model with lowest value of AIC is the first model.

Exercise 10

Use the model found in the previous exercise to make a forecast for the next 6 periods, and plot the forecast. (The forecast requires a matrix of the expected temperature and income for the next 6 periods; create the matrix using the **fcast_temp** variable, and the following values for expected income: 91, 91, 93, 96, 96, 96). Find the mean absolute scaled error of the model, and compare it with the ones from the first two models in this exercise set.

```
expected_temp_income <- matrix(c(fcast_temp, 91, 91, 93, 96, 96, 96), ncol=2, nrow=6)
fcast_cons_temp_income <- forecast(fit_vars_0, xreg=expected_temp_income, h=6)
autoplot.forecast(fcast_cons_temp_income)
```


Forecasts from Regression with ARIMA(1,0,0) errors
df$cons
0.5
0.4
0.3
0
10
20
30
Time
level
80
95