exercise_part04

Forecasting: Multivariate Regression

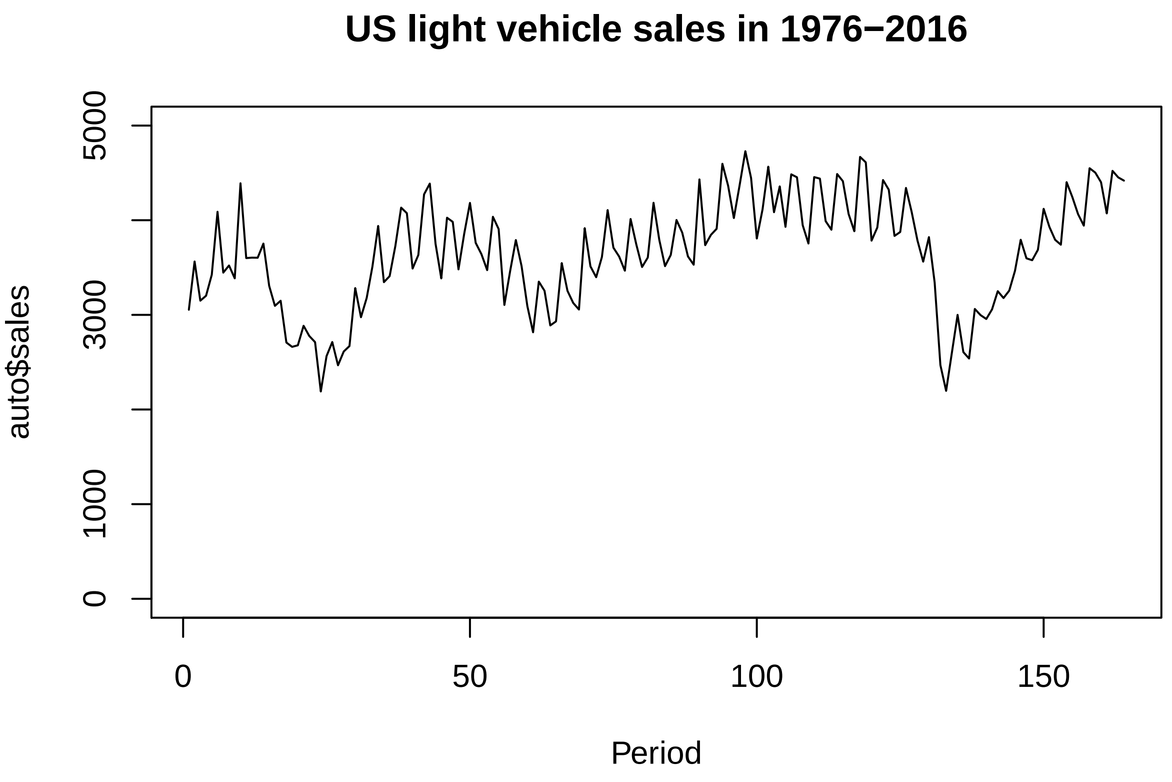
Original Source

Another approach to forecasting is to use external variables, which serve as predictor. Running regressions may appear straightforward but this mehtod of forecasting is subject to some pitfalls: (1) a basic difficulty is selection of predictor variables (which is more of an art than a science), (2) a possible problem is the dependence of a forecast on assumptions about expected values of predictor variables, (3) another problem can arise if autocorrelation is present in regression residuals (it implies, among other things, that not all information, which could be used for forecasting, was retrieved from the forecast variable).

Exercise 1

Load the dataset, and plot the sales variable.

```r
auto <- read.csv("vehicles.csv")
plot(auto$sales, type="n", ylim=c(0,5000), xlab="Period", main="US light vehicle sales in 1976-2016")
lines(auto$sales)
```



Exercise 2

Create the trend variable (by assigning a successive number to each observation), and lagged versions of the variables income, unemp, and rate (lagged by one period). Add them to the dataset. (Note that the base R libraries do not include functions for creating lags for non-time-series data, so the variables can be created manually).

```
auto$trend <- seq(1:nrow(auto))
auto$income_lag <- c(NA, auto$income[1:nrow(auto)-1])
auto$unemp_lag <- c(NA, auto$unemp[1:nrow(auto)-1])
auto$rate_lag <- c(NA, auto$rate[1:nrow(auto)-1])
```

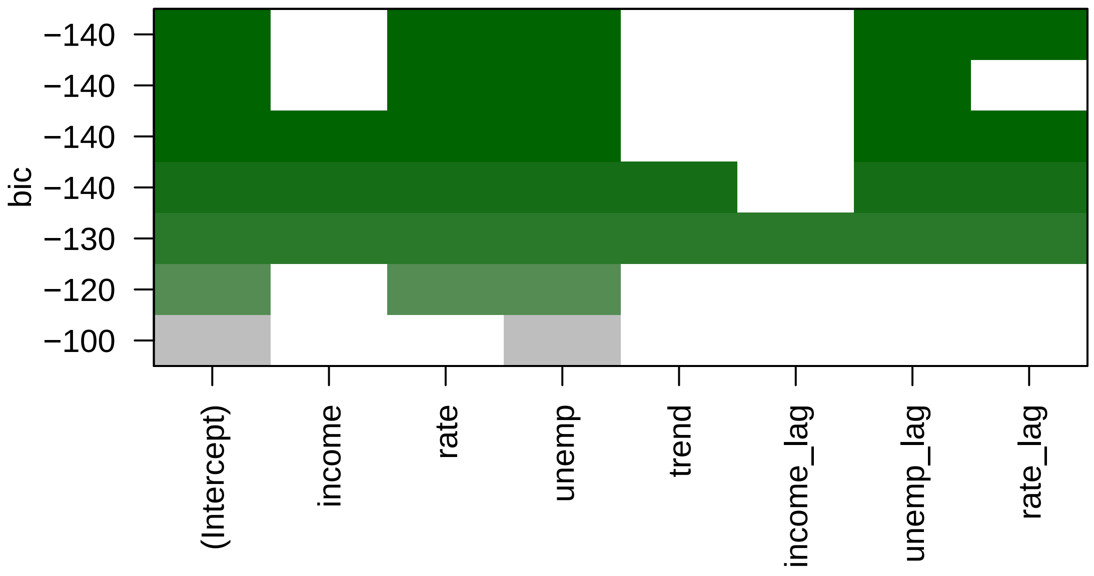
Exercise 3

Run all possible linear regressions with sales as the dependent variable and the others as independent variables using the **regsubsets** function from the **leaps** package (pass a formula with all possible dependent variables, and the dataset as inputs to the function). Plot the output of the function.

```
require(leaps)
```

```
## Loading required package: leaps
```

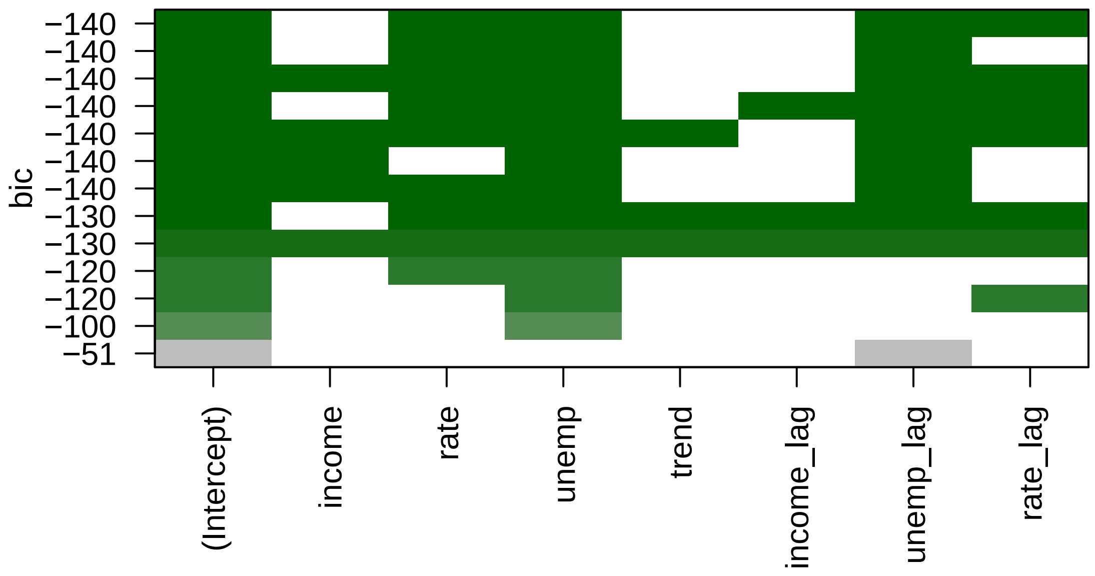
```
regressions_result <- regsubsets(sales~., data=auto)
plot(regressions_result, col=colorRampPalette(c("darkgreen", "grey"))(10))
```



Exercise 4

Note that **regsubsets** returns only one "best" model (in terms of BIC) for each possible number of dependent variables. Run all regressions again, but incease the number of returned models for each size to 2. Plot the output of the funciton.

```
regressions_result <- regsubsets(sales~., data=auto, nbest = 2)
plot(regressions_result, col=colorRampPalette(c("darkgreen", "grey"))(10))
```



Exercise 5

Look at the plots from the previous exercises and find the model with the lowest value of BIC. Run a linear regression for the model, save the result in a variable, and print its summary.

```
# Given that the vertical scale of the plots is reversed the model with the lowest BIC is the uppermost
fit <- lm(sales ~ unemp + rate + unemp_lag + rate_lag, data = auto)
summary(fit)
```

```
##
## Call:
## lm(formula = sales ~ unemp + rate + unemp_lag + rate_lag, data = auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1145.76  -230.19    -3.68   233.05   818.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5963.35     192.77  30.935  < 2e-16 ***
## unemp        -465.56      47.06  -9.892  < 2e-16 ***
## rate         -262.21      72.24  -3.630 0.000383 ***
## unemp_lag     239.50      46.45   5.156 7.46e-07 ***
## rate_lag      197.41      73.44   2.688 0.007961 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 347.1 on 158 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.6401, Adjusted R-squared:  0.631
## F-statistic: 70.26 on 4 and 158 DF,  p-value: < 2.2e-16
```

Exercise 6

Load an additional dataset with asumptions on future values of dependent variables. Use the dataset and the model obtained in the previous exercise to make a forecast for the next 4 quarters with the **forecast** function (from the package with the same name). Note that the names of the lagged variables in the assumptions data have to be identical to the names of the correpsonding variables in the main dataset. Plot the summary of the forecast.

```
require(forecast)
```

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

```
assumptions <- read.csv("vehicles_assumptions.csv")
fcast <- forecast(fit, newdata = assumptions, h=4)
summary(fcast)
```

```
##
## Forecast method: Linear regression model
##
## Model Information:
##
## Call:
## lm(formula = sales ~ unemp + rate + unemp_lag + rate_lag, data = auto)
##
## Coefficients:
## (Intercept)        unemp         rate    unemp_lag     rate_lag
##      5963.3       -465.6       -262.2        239.5        197.4
##
##
## Error measures:
##                        ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 8.34353e-15 341.6871 273.3704 -0.9414979 7.769387 0.5926906
##
## Forecasts:
##   Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 1       3703.978 3240.687 4167.270 2992.952 4415.005
## 2       4517.922 4061.405 4974.439 3817.293 5218.551
## 3       4122.130 3668.325 4575.935 3425.663 4818.597
## 4       4364.266 3910.470 4818.063 3667.812 5060.721
```

Exercise 7

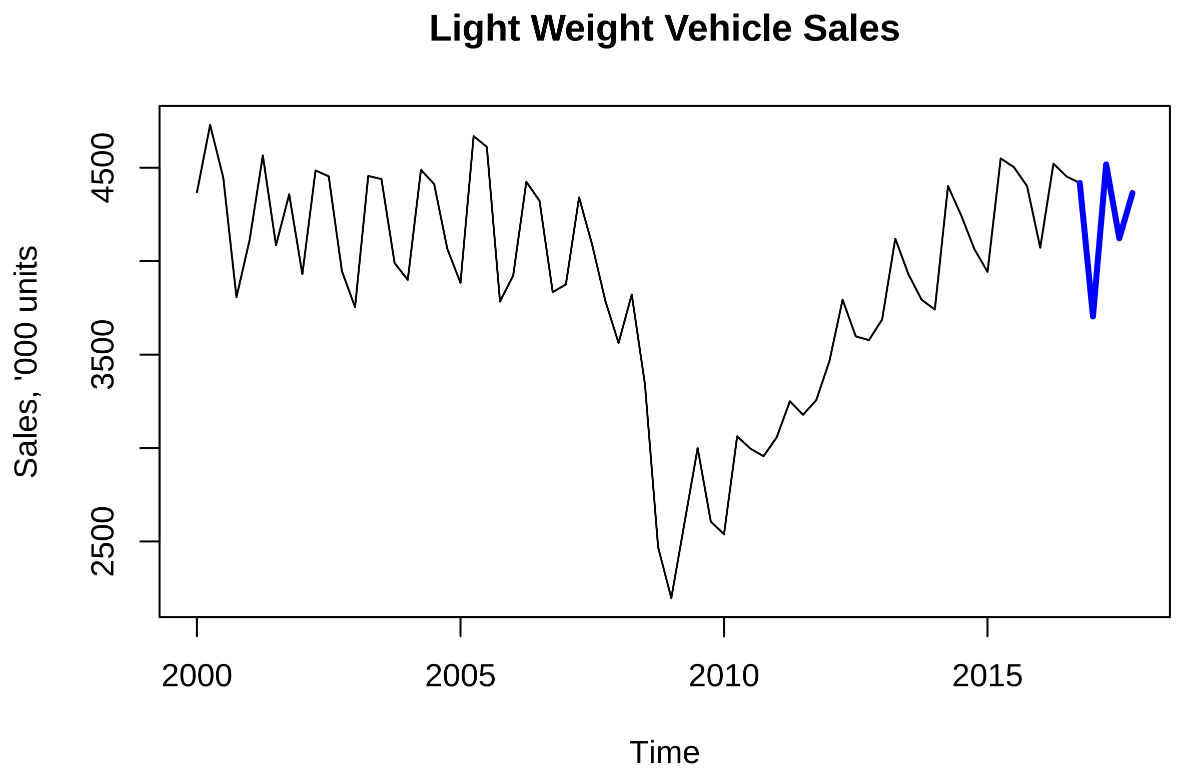
The **plot** function does not automatically draw plots for forecasts obtained from regression models with multiple predictors, but such plots can be created manually. As the first step, create a vector from the sales variable, and append the forecast (mean) values to this vector. Then use the **ts** function to transform the vector to a quarterly time series that starts in the first quarter of 1976.

```
fcast_sales <- append(auto$sales, fcast$mean)
fcast_sales <- ts(fcast_sales, start = c(1976,1), frequency = 4)
```

Exercise 8

Plot the forecast in the following steps: (1) create an empty plot for the period from the first quarter of 2000 to the forth quarter of 2017, (2) plot a black line for the slaes time series for the period 2000 - 2016, (3) plot a thick blue line for the sales time series for the forth quarter fo 2016 and all quarters of 2017. Note that a line can be plotted using the **lines** function, and a subset of a time series can be obtained with the **window** function.

```r
plot(window(fcast_sales, start=c(2000,1), end=c(2017,4)), type='n', ylab="Sales, '000 units", main="Ligl
lines(window(fcast_sales, start=c(2000,1), end=c(2016,4)))
lines(window(fcast_sales, start=c(2016,4), end=c(2017,4)), col="blue", lwd=3)
```



Exercise 9

Perform the Breusch-Godfrey test (the **bgtest** function from the **lmtest** package) to test the linear model obtained in the exercise 5 for autocorrelation of residuals. Set the maximum order of serial correlation to be tested to 4. Is the autocorrelation present? (Note that the null hypothesis of the test is the absence of autocorrelation of the specified orders).

```r
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
```

```r
bgtest(fit, 4)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 4
```

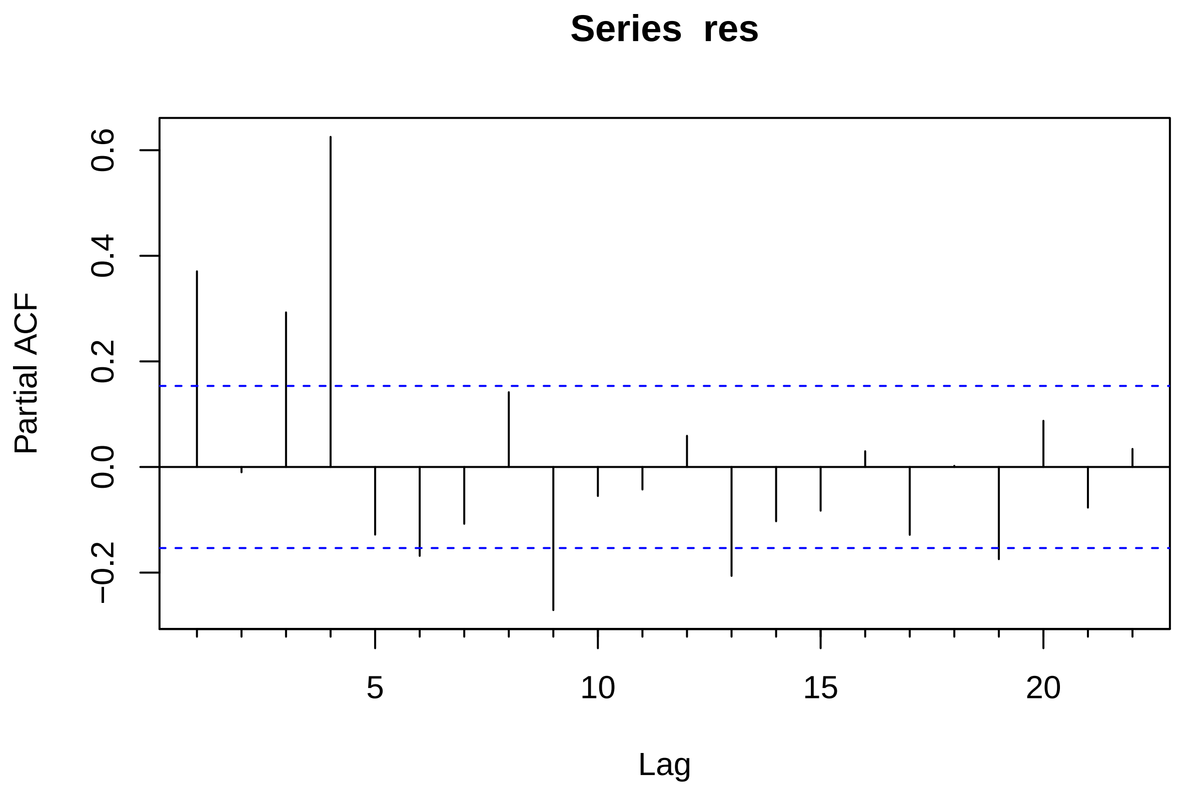
```
##
## data:  fit
## LM test = 95.587, df = 4, p-value < 2.2e-16
```

The p-value is less than 0.05, which implies that the null hypothesis of the abscence of autocorrelation fo the orders 1-4 can be rejected.

Exercise 10

Use the **Pacf** function from the forecast package to explore autocorrelation of residuals of the linear model obtained in the exercise 5. Find at which lags partial correlation betwen lagged values is statistically significant at 5% level. Residuals can be obtained from the model using the residuals function.

```r
res <- residuals(fit)
Pacf(res)
```



The plot shows that correlation is present at lags 1,3,4,6,9,13,19.