Chapter 10 -Solutions

10.2.5 Exercises

Exercise 10.1

The program for this exercise is Exercise 10-1-code.R, the dataset is `AEP_hourly.csv` from the Case Study of Chapter 3, and the reader is advised to run this program and see the code comments written within it. In the narrative that follows, we will just summarize the results. But before that, we will comment about the data and the programs used to inspire the one used for this exercise.

The dataset from which the time series for the Case Study in Chapter 3 comes measures the hourly energy consumption in megawatts (MW) for different eastern regions in the United States as collected from the PJM website (pjm.com). The Case Study in Chapter 3 focuses on the time series corresponding to the American Electric Power Co., Inc. (time series dataset `AEP_hourly.csv`), which can be found in the folder for Chapter 3's Case Study. This time series comes with a column containing a character date variable (`Datetime`) and the energy consumption variable (or feature), (`AEP_MW`). This contrasts with the format of the USVSales data read into program *MLreg.R*, where the date is just a date, not a date and time, and the frequency is monthly, not hourly. So in order to use Program MLEreg.R, we will make some compromises, given that the `h2o` package expects a certain format for its acceptable datasets. The reader will notice that *Exercise 10-1-code.R* also borrows some steps from Program *chapter3case.R*. We recommend, before starting, that the reader read again the Case Study in Chapter 3, and Example 10.1. We will be creating more features than in Example 10.1, given the granularity of the `AEP_MW` and conclusions we reached in Chapter 3, where we investigated the seasonalities. We will not repeat in this Exercise the exploratory analysis done in the Case Study of Chapter 3. There, we noticed that the month of the year, the day of the week, and the hour of the day appeared relevant in explaining the variability of the energy consumption, at least descriptively (Figure 3.14). The time series has a slight nonlinear trend, but the number of years observed, 2004-2018, are not enough years to determine the trend. The slightly upward and then downward trend that we observe in Figure 3.12 could just be the business cycles (see `https://www.nber.org/research/business-cycle-dating`

In what follows we discuss the results. The reader can enrich this summary with more details, as needed.

Comparing Models

The dataset studied, `AEP_hourly.csv` comes with two variables, as we said: `AEP_MW` and `Datetime`. The energy consumption estimates are reported from October 1st 2004, 1:00 AM to August 3, 2018, at the 00:00 hour. Descriptive analysis can be found in the Case Study of Chapter 3. The time series is split, as usual, into a training set and a test set. The features used as inputs in the predictive models for the training set considered are: as factors, `month, weekday, hour` and as numerical variables, `lag24, lag168, lag 8760`, `trend, trend_sqr`. The target variable is energy consumption, which we end up labeling as `y`, for convenience.

We fit three models to the training set, and forecast the test set period with each of them to evaluate the models' performance. A benchmark model against which we compare a Random Forest (RF) model and a Gradient Boosting (GB) model is the Multiple Regression (MR) model. These models are fit to the training set which contains all observations (cases, examples) except those of the last week, which is forecasted. Thus we use a test dataset of 168 hours of energy consumption in the last week. We could use as test set, and forecast, a whole year if we want, but

then the time plot of the energy consumption and the model's forecast in each of the 8760 hours that we are assuming a year to have is hard to visualize. The reader can try several horizons for the test set dataset. We point out in the R program where to change the program in order to predict a different test dataset.

The reader should notice the following that we did not do: (a) We did not pursue further fixing the Daylight Savings (DS) issue that we observed in Chapter 3's Case Study. The data does indeed have a few days with 25 hours (when DS ends) and other days with 23 hours (when DS starts). But since we extract features and the date time is not used directly in the models, that will not pose a problem. (b) We did not standardize the numeric variables manually to have mean of zero and standard deviation one. The `h2o` package does that for us. For most ML models, it is recommmended to scale the numerical variables.

After running Program *Exercise 10-1-code.R* , the MAPE for the train and test period (rounded to the fifth decimal) are compared for the three models next.

Model	MAPE train period	MAPE test period	Residuals
MR	0.04773	0.03092	autocorrelated
RF	0.01706	0.03401	autocorrelated
GB	0.00418	0.04026	autocorrelated

As we can see, GB has the largest MAPE for the test period and GB has the lowest for the training period. We also notice in the table that both RF and GB present larger MAPE in the test period than in the training period, an indication of overfitting during the train period, a tendency of ML regression models.

The variables considered as most important by the RF and GB are, in the given order:

RF: lag24, lag168, lag8760, hour, month, weekday, and the two trend variables
GB: lag24, weekday, lag168, hour, month, lag8760, and the two trend variables.

We observe that in all the models, the residuals of the model fitted to the training period are autocorrelated. This is an indication that the models have not captured all features that might affect the target variable. Perhaps the slight trend is better represented with other trend model, instead of a quadratic polynomial. Other variables relevant to hourly energy consumption might be relevant, but they are omitted.

One last observation is that we noticed some outlying residuals when plotting the residuals of the model fitted to the training data.

The reader can add additional details and sections as needed.

□

10.4.1 Exercises

Exercise 10.2

The data set for this Exercise is *BirminghamCh10.csv* and the program is *Exercise10-2-code.R*

The Birmingham data set used for Chapter 4's Case Study contains occupancy rates updated every 30 minutes between 8:00 and 16:30 from October 4th 2016 to December 19th, 2016 (11 weeks of consecutive days). It contains 4 variables (features). The `SystemCodeNumber` for the sensor, `Capacity, Occupancy` and `LastUpdated`. The interest is in the occupancy rate, calculated by dividing the occupancy by the capacity. Program *Birmingham.R* analyzed the quality of the data and prepared the data for cluster analysis. The reader should review that Case Study and go over the Program and the data again before proceeding with this new problem. Here, we will be using the `finaldata` created in Program *Birmingham.R*. We saved that file with name *BirminghamCh10.csv*. Before starting the analysis requested for this exercise, we will briefly summarize how Program *Birmingham.R* arrived to the final data set.

As indicated in Case Study of Chapter 4, initially, the data set contained data for 30 parking structures. But after inspection of the quality of the data, explained in Case Study of Chapter 4, we got reduced to 29 for the descriptive analysis (NIA North was removed) in Table 4.4 of Chapter 4 because it had negative values of occupancy and very few

observations. Then we selected, for the Case Study of Chapter 4, six parking structures, none of which had negative rates or occupancy rates larger than one. In appearance there were no missing values in these 6 parking structures (no NA or empty spaces for a given date). However, upon inspection, it is clear that the sensor was broken between October 19, 4:25 PM and October 22, 7:59 AM. We discovered observing Figure 4.4. The reader can add the following R code after reading the `park` dataset in Case Study of Chapter 4 with Chapter 4's Case Study Program *Birmingham.R* to observe the missing dates and values.

```
mytest=park

library(lubridate)  # install first if not in your system
mytest$date=ymd_hms(LastUpdated)
head(mytest)
str(mytest)

dataA=with(mytest, mytest[(month(date)=10 & day(date)>=19)  & (month(date)=10 & day(date)<=22),])
View(dataA)
```

We will work henceforth with the six parking structures selected, which can be found in the data set *BirminghamCh10.csv*.

We obtained this dataset in Chapter 4's Case Study Program *Birmingham.R* by typing the following command after the `finaldata` is created:

```
write.csv(finaldata, file="BirminghamCh10.csv",row.names=F)
```

The fact that the data set is interrupted at 16:30 each day, and the fact that some interval in the data set is not recorded would cause some problems in classical time series analysis models and Base R's case study . The time index's regularity is something desirable, as it makes analysis simpler. But for ML regression models those hurdles pose no problem, since we extract features from the data set, so we do not need that the time stamp is uninterrupted. In fact, in the ML Regression models and the MR models that we consider, the time stamp plays no role other than helping extract the time features such as month, etc...

We fitted the MR, RF and GBM separately first. Then we applied `AutoML` to *BirminghamCh10.csv* (which takes about 20 minutes), to see which model is selected as the winner. The leaderboard indicated that the NN model was not the winner. Hence it just gave the output of the winner. In order for us to be able to fit a NN and forecast the test set we use another function in `h2o`, the `h2o.deeplearning`.

Model	MAPE train period	MAPE test period	Residuals
MR	0.11824	0.12977	autocorrelated
RF	0.02573	0.09557	autocorrelated
GB	0.00071	0.09635	autocorrelated
NN	0.10030	0.11154	autocorrelated

As we can see, GB has the lowest MAPE of all four models for the test period and GB has the lowest for the training period as well. But the MAPE of the training period is much smaller than the one in the test period, indicating overfitting. We also notice in the table that both RF, GB and NN present larger MAPE in the test period than in the training period, an indication of overfitting during the train period, a tendency of ML regression models.

The variables considered as most important by the MR, RF and GB are, in the given order:

MR (scaled): CCCPS135a, hour16, hour15, weekday4, hour, CCCPS8, BHMBCCMKT01, lag119, month.

RF: CCCPS135a, Shopping, weekday, CCCPS98, lag17, hour, CCCPS8, BHMBCCMKT01, lag119, month.

GB: CCCPS135a (most important), the next ones very little importance: Shopping, weekday, BHMBCCMKT01, hour, CCCPS98, CCCPS8, lag17, lag119, month.

NN: CCCPS135a, Shopping, weekday.7, hour.16, month.11, BHMBCCMKT01, weekday.1, hour.12, weekday.5, hour.11

We observe that in all the models, the residuals of the model fitted to the training period are autocorrelated. This is an indication that the models have not captured all features that might affect the target variable. Other variables affecting parking rates that we have omitted as inputs might be behind the behavior observed in the parking rate at Broad Street park. People expert in parking occupancy would have to be consulted and more data would have to be obtained to improve the model. Sometimes practitioners play with the arguments of the functions to improve the model, but autocorrelated residuals is a problem in ML for time series despite that.

One last observation is that we noticed some outlying residuals when plotting the residuals of the model fitted to the training data.

The reader can do further analysis and add more sections, as needed.

□

Exercise 10.3

Please, notice correction to version in the textbook.

Using the same input as used in the RF and GB example discussed in Example 10.1, Krispin [99] used the `AutoML` of the `h2o` package and this procedure chose DL (Deep Learning) models with different tuning settings. `AutoML` can scale up while having multiple time series to forecast with minimal intervention from the user. "Use `AutoML` with the AEP data used in Exercise 10.1 to determine which is the top model in the leaderboard."

Basically, to do this exercise, do the same activities that we did in Exercise 10.1, but add the part where we used the `AutoML` or if necessary `h2o.deeplearning`.

□

Extra Exercises

Extra Exercise 1

This exercise is on unsupervised ML. Summary features will be obtained and a cluster analysis will be done. But first, a little bit of practice.

To get started, use the code *Extra-Exercise-1.R*

You will have to complete the code to do the k-means using examples we did earlier in the book.

(a) Acquaint yourself with the package `tsfeatures`. To do that, visit

`https://cran.r-project.org/web/packages/tsfeatures/vignettes/tsfeatures.html`

to see an introduction to the package, and the type of summary features that it produces for a list of time series provided by the authors and observe what you obtain.

The features obtained by this package are summary features, unlike for example "month, day, hour, lag12, etc.." that we would use to do ML regression. That is, for each time series and each feature there will be just one number, e.g., the mean, or the standard deviation, or an autocorrelation. So for four time series the generated summary features time series will have only 4 lines.

You will notice that the time series used in the documentation are very different. Not all features are useful for all of them (see Section 10.5).

Select only `AirPassengers` (that is, remove from the list given in the documentation all the others). Obtain summary features just for the AirPassengers time series. Select only the features that you think are useful for the time series and indicate why you chose them (do that for each feature). Enter the features and explanation. Justify your choices carefully, that is, indicate why you think those features are important for `AirPassengers`.

(b) The next step is to apply what is learned in the documentation and practice in part (a) to analyze data that is in a flat file (rectangular data set) containing seasonally adjusted monthly unemployment rate for all the OECD (Organization for Economic Cooperation and Development) countries between January 2005 and August 2023.

Read the data file *OECDunemployment.csv* from the Chapter 10 data folder to proceed. The citation for this data set is : *OECD (2023), Unemployment rate (indicator). doi: 10.1787/52570002-en (Accessed on 31 October 2023)* .

When initially downloaded at the indicated date, the data file had the following name: `DP_LIVE_31102023170613427`

To acquaint ourselves with the source of the data, go to `https://data.oecd.org/unemp/unemployment-rate.htm`. We downloaded the monthly data from there. You will need to do the following:

- Split the data by levels of the factor `LOCATION`. That is, prepare the data file so that it can be read as a file with used to extract summary features from all the time series using the `tsfeatures` package.
- With the file containing summary features for each country perform k-means clustering. Determine which countries go to which cluster and plot the clusters using the summary features that best partition the countries. Is there something in common to the countries in each cluster? What distinguishes the clusters? Answering these last questions may require a little bit of research about the economies of the countries with, for example, Wikipedia, if not familiar with the countries.
- Discuss what might we be missing by using seasonally adjusted data for comparing the OECD countries' unemployment rate? Explain.

□

Extra Exercise 2

Do some research to investigate the R function `neuralnet()` of the `neuralnet` package and its uses and examples in unsupervised ML. Try to apply it to the dataset that we explored with the *MLreg.R* program.