Homework Assignment 8

Panagiotis (Panos) Toulis – Chicago Booth
BUS41100 Applied Regression Analysis

1 Exploring the Time Series Zoo

`R` comes with a lot of time series datasets already in its `base` distribution. See here for a description `https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html`. The purpose of this exercise is to explore some of these time series, and recognize and describe their different types.

(a) Consider the `airquality$wind` dataset.

- What data does this time series represent? What is the frequency of measurement?
- What is the most prevailing characteristic: autocorrelation, seasonality, trend or recurring "special events"?
- Produce the `acf` plot of the time series and describe how autocorrelation decays. What type of time series do you see: Mean-reverting, seasonal or random walk?
- Model the time series according to the prevailing characteristic; i.e., if it is autocorrelation, run $Y_t \sim Y_{t-1}$; if it is trend run OLS on $Y_t \sim t$; if it is seasonality, run $Y_t \sim \cos(2\pi t/k) + \sin(2\pi t/k)$ for an appropriate k, etc.
- Produce the `acf` plot of the residuals from the above regression. Have you taken into account all temporal dependence? (i.e., could we argue that the residuals are *not* a time series?) If not, propose one simple way to account for the remaining temporal dependence, but do not implement it.

(b) Repeat these steps for `austres`.

(c) Repeat these steps for `BJSales` (*Note:* You may have to use `Yt = as.numeric(BJsales)` to make this work).

2 Food Inspections in Chicago

The file `Food_Inspections_proc.csv` contains detailed information about inspections on food establishments by the City of Chicago.[1] The information includes the name of the establishment, the date of inspection, and the results from the inspection. The results from an inspection can be either "Pass", "Pass w/Conditions", or "Fail".

We will work with a cleaned-up version of the data obtained using the following code:

```
require(tidyverse)
food = read.csv("Food_Inspections_proc.csv", header=T)
food = food %>% mutate(FAIL=(Results=="Fail"),
                       SERIOUS_FAIL=(IsSerious),   # flag indicating FAIL was serious
                       PASS=Results %in% c("Pass", "Pass w/ Conditions"),
                       NO_ENTRY=(Results=="No Entry"),
```

[1] Source: `https://data.cityofchicago.org/Health-Human-Services/Food-Inspections/4ijn-s7e5`

```
                    NOT_READY=(Results=="Not Ready")) %>%
            select(-Results, -IsSerious)
```

(a) We want to rank establishments (variable `DBA.Name`) according to their *pass rate* (i.e., success rate in inspections) restricted to those larger establishments that have undergone at least 40 inspections. We use the following code:

```
quality = food %>% group_by(DBA.Name) %>%
                    summarise(NUM=n(), PASS_RATE=...) %>%
                    filter(NUM > 40) %>% arrange(...)
```

Fill in the dots. If you are Chicago-based, report the score of your favorite establishment in the list, and also your level of disappointment.

(b) Next, we focus on the time evolution of inspection success rate citywide. We write the following code:

```
agg = food %>% group_by(Year, Month) %>%
                summarise(TESTS=n(), PASS_RATE=sum(PASS)/(sum(PASS)+sum(FAIL))) %>%
                  mutate(DATE=paste(Year,"-", Month, sep=""))
Y = agg$PASS_RATE   # Outcome of interest
plot(Y, type="l", xaxt='n')
axis(1, labels=agg$DATE, at=1:nrow(agg))
```

How many observations are in time series Y? How many years does it span? What is the frequency of observation?

(c) Present the `acf` plot of Y. Do we observe significant autocorrelations? Which regime are we in? (explosive, stationary, non-stationary?).

(d) Fit an AR(1) model to Y and report the results. If this is a stationary time series, report the estimated mean level.

(e) Present the `acf` plot of residuals from the fit in (d). Have we taken care of most autocorrelations? Which lag appears to have a significant residual autocorrelation?

(f) Based on your answer in (e), add periodic predictors to model any seasonality. Using `anova` or `AIC` argue whether the new model is better than simple AR(1).

3 Walmart Challenge

The file `walmart_train.csv` contains time-series for actual weekly sales at Walmart from 2010 to 2012 (partially). Specifically, it contains:

- `Weekly_Sales` = sales at week t (our Y_t).
- `PrevWeekly_Sales` = weekly sales at $t-1$ (our Y_{t-1}).
- `Temperature` = temperature at week t.
- `Fuel_Price` = fuel price at week t.
- `CPI` = consumer price index estimate in the region at time t.
- `IsHoliday` = binary, whether week t contains a holiday or not.

Load the data using:

```
wal_train = read.csv("walmart_train.csv")
```

Then, answer the following questions:

(a) An easy way to visualize the data with proper axis labeling is the following.

```
wal_train = read.csv("walmart_train.csv")
plot(wal_train$Weekly_Sales, xaxt="n", type="b", pch=20)
axis(1, labels=wal_train$Date, at=1:nrow(wal_train))
```

Comment on the time-series plot you see. Is there a linear trend? Is there seasonality? What other effects do you see?

(b) Fit the AR(1) model $\log(Y_t) = \beta_0 + \beta_1 \log(Y_{t-1}) + \varepsilon_t$, where Y_t are weekly sales. Do you see significant autocorrelation? Which regime are we in?

(c) Show the autocorrelation plot of the residuals from (b). What are the lags with statistically significant nonzero autocorrelation?

(d) Investigate the residuals with respect to month of prediction:

```
mon = month(as.Date(wal_train$Date)[-1])
boxplot(fit$residuals ~ mon)
```

From this boxplot, do you see any systematic mistakes made by our model?

(e) **Out-of-sample Prediction Challenge.** Add the code from file `hw8-walmart.R` directly into your script. This contains two functions, namely `prepare_X()` and `predict_with_model()`, which automate model building and model testing. You are asked to build a model that will predict the 23 values (about 6 months ahead) corresponding to the dates in `walmart_test.csv`, which come immediately after the dates in `walmart_train`. Your performance will be measured as the mean absolute relative error from `ytrue` (the true sales):

```
diff = 100*(ypred-ytrue)/ytrue
mean(abs(diff))
```

You should only change the `prepare_X()` function in the script by adding your features in the placeholder `''Your additional operations here...''` and utilize `predict_with_model` to generate the predictions. Using your model, create a vector `ypred` with your 23 predictions and save it as follows:

```
wal_train = prepare_X(wal_train)  # using your own prepare_X()
fit = lm(WeeklySales ~ ., wal_train) # your model here
ypred = predict_with_model(fit) # predictions
save(ypred, file="41100_TeamName.rda")
```

Submit the file "`41100_TeamName.rda`" along with your writeup.

4 Unit Root Tests

A "unit root test" is a statistical test to determine whether a time series is stationary or not. These tests are very popular in financial analyses, and you will probably encounter it again at Booth if you take a finance course. The most popular test of this kind is arguably the "Augmented Dickey–Fuller

test" (ADF). [2] The null hypothesis is:

$$H_0 : \text{time series is non-stationary.}$$
$$H_1 : \text{time series is stationary.}$$

In its simplest form, the idea is to test $H_0 : \beta_1 = 1$ in the AR(1) model $Y_t = \beta_0 + \beta_1 Y_{t-1} + \varepsilon_t$, as we saw in class. However, our test in class was only approximate because Y_{t-1} is a regressor that is not a "fixed X". The ADF test tries to correct for that.

In this exercise, we will use ADF to test several examples of time series for stationarity. The following function implements this test.

```
library(forecast) # install.packages("forecast") if not done yet
library(tseries)  # install.packages("tseries") if not done yet
#
stationarity_test = function(y) {
  print("H0: series is non-stationary  (test at 1% level)")
  # ADF test.
  out = adf.test(y)
  print(paste("ADF test p-value=",round(out$p.value,3), "-- Reject=", out$p.value <= 0.01))
}
```

For help answering the following questions, look into the Wikipedia lemma and also type "`adf.test`" in the `R` console and press enter—this will show you the source code of the function.

(a) What is the test statistic of the ADF test? (you may look at the "`STAT`" line in the source code of `adf.test`). Which test statistic from our class does it relate to?

(b) How does `adf.test` calculate significance? (Look at the "`PVAL`" line in the source code of `adf.test` for help).

(c) Execute the test for 6 different synthetic datasets as follows:

```
set.seed(41100)
# example from ?adf.test
y1 = rnorm(1000)  # iid
stationarity_test(y1)

y2 = cumsum(rnorm(3000))  # Random walk
stationarity_test(y2)

y3 = arima.sim(n=1000, list(ar=c(0.5)), sd=1) # simulates AR(1) with beta1=0.5
stationarity_test(y3)

y4 = arima.sim(n=1000, list(ar=c(0.9)), sd=1) # simulates AR(1) with beta1=0.9
stationarity_test(y4)

y5 = arima.sim(n=1000, list(ar=c(0.99)), sd=1) # simulates AR(1) with beta1=0.99
stationarity_test(y5)

beer = read.csv("beer.csv")
y6 = beer$production
stationarity_test(y6)
```

For every dataset, from `y1` to `y6`: (i) determine whether H_0 (non-stationarity) is true or not, and (ii) report whether ADF decides correctly. Try to explain briefly any failures.

[2] `https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test`