Assignment time series

2024-09-20

library("lubridate")

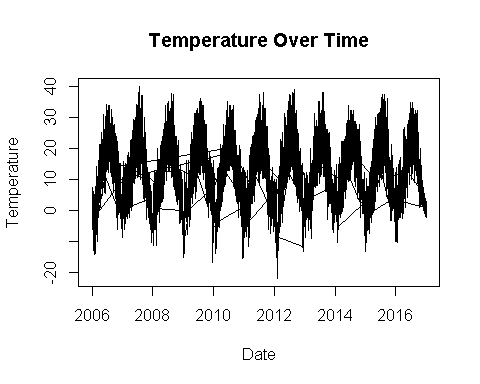
##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

# Import the weather data  
weather\_data <- read.csv("weather.csv")  
head(weather\_data)

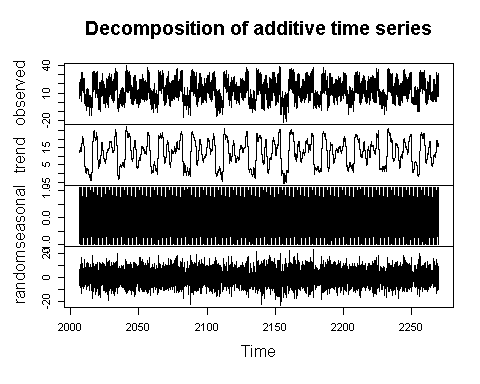
## Formatted.Date Summary Precip.Type Temperature..C.  
## 1 2006-04-01 00:00:00.000 +0200 Partly Cloudy rain 9.472222  
## 2 2006-04-01 01:00:00.000 +0200 Partly Cloudy rain 9.355556  
## 3 2006-04-01 02:00:00.000 +0200 Mostly Cloudy rain 9.377778  
## 4 2006-04-01 03:00:00.000 +0200 Partly Cloudy rain 8.288889  
## 5 2006-04-01 04:00:00.000 +0200 Mostly Cloudy rain 8.755556  
## 6 2006-04-01 05:00:00.000 +0200 Partly Cloudy rain 9.222222  
## Apparent.Temperature..C. Humidity Wind.Speed..km.h. Wind.Bearing..degrees.  
## 1 7.388889 0.89 14.1197 251  
## 2 7.227778 0.86 14.2646 259  
## 3 9.377778 0.89 3.9284 204  
## 4 5.944444 0.83 14.1036 269  
## 5 6.977778 0.83 11.0446 259  
## 6 7.111111 0.85 13.9587 258  
## Visibility..km. Loud.Cover Pressure..millibars.  
## 1 15.8263 0 1015.13  
## 2 15.8263 0 1015.63  
## 3 14.9569 0 1015.94  
## 4 15.8263 0 1016.41  
## 5 15.8263 0 1016.51  
## 6 14.9569 0 1016.66  
## Daily.Summary  
## 1 Partly cloudy throughout the day.  
## 2 Partly cloudy throughout the day.  
## 3 Partly cloudy throughout the day.  
## 4 Partly cloudy throughout the day.  
## 5 Partly cloudy throughout the day.  
## 6 Partly cloudy throughout the day.

# Convert to Time Series:  
# Convert 'date' column to Date format  
weather\_data$Formatted.Date <- as.Date(weather\_data$Formatted.Date, format="%Y-%m-%d")  
  
# Assuming daily weather data  
ts\_weather <- ts(weather\_data$Temperature..C., start=c(year(min(weather\_data$Formatted.Date)), month(min(weather\_data$Formatted.Date))), frequency=365)  
   
#Plot the Time Series:  
plot(weather\_data$Formatted.Date, weather\_data$Temperature..C., type="l", xlab="Date", ylab="Temperature", main="Temperature Over Time")



#Decompose the Time Series:  
decomposed\_weather <- decompose(ts\_weather)  
plot(decomposed\_weather)  
   
#Check for Stationarity:  
library(tseries)

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo



adf\_test <- adf.test(ts\_weather)

## Warning in adf.test(ts\_weather): p-value smaller than printed p-value

print(adf\_test)

##   
## Augmented Dickey-Fuller Test  
##   
## data: ts\_weather  
## Dickey-Fuller = -10.099, Lag order = 45, p-value = 0.01  
## alternative hypothesis: stationary

## H0 = Data is not stationary  
## H0 is rejected as p value is less than 0.05 so data is stationary  
  
#Fit AR, MA, and ARIMA Models:  
  
# AR model  
  
ar\_model <- arima(ts\_weather, order=c(1,0,0))  
summary(ar\_model)

## Length Class Mode   
## coef 2 -none- numeric   
## sigma2 1 -none- numeric   
## var.coef 4 -none- numeric   
## mask 2 -none- logical   
## loglik 1 -none- numeric   
## aic 1 -none- numeric   
## arma 7 -none- numeric   
## residuals 96453 ts numeric   
## call 3 -none- call   
## series 1 -none- character  
## code 1 -none- numeric   
## n.cond 1 -none- numeric   
## nobs 1 -none- numeric   
## model 10 -none- list

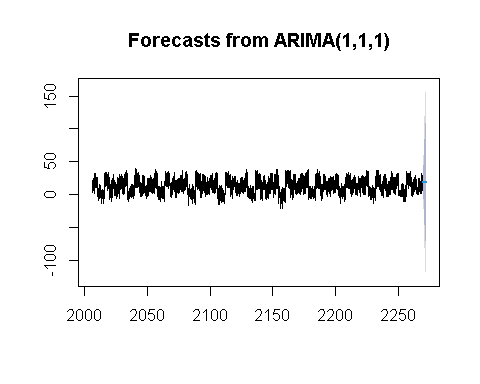
# MA model  
ma\_model <- arima(ts\_weather, order=c(0,0,1))  
summary(ma\_model)

## Length Class Mode   
## coef 2 -none- numeric   
## sigma2 1 -none- numeric   
## var.coef 4 -none- numeric   
## mask 2 -none- logical   
## loglik 1 -none- numeric   
## aic 1 -none- numeric   
## arma 7 -none- numeric   
## residuals 96453 ts numeric   
## call 3 -none- call   
## series 1 -none- character  
## code 1 -none- numeric   
## n.cond 1 -none- numeric   
## nobs 1 -none- numeric   
## model 10 -none- list

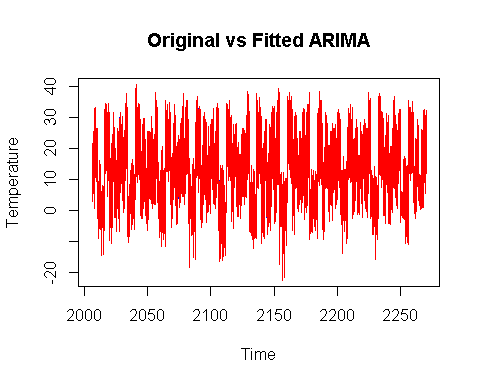
# ARIMA model  
arima\_model <- arima(ts\_weather, order=c(1,1,1))  
summary(arima\_model)

## Length Class Mode   
## coef 2 -none- numeric   
## sigma2 1 -none- numeric   
## var.coef 4 -none- numeric   
## mask 2 -none- logical   
## loglik 1 -none- numeric   
## aic 1 -none- numeric   
## arma 7 -none- numeric   
## residuals 96453 ts numeric   
## call 3 -none- call   
## series 1 -none- character  
## code 1 -none- numeric   
## n.cond 1 -none- numeric   
## nobs 1 -none- numeric   
## model 10 -none- list

#Plot and Interpret Results:  
library(forecast)  
forecast\_arima <- forecast(arima\_model)  
plot(forecast\_arima)



# Compare original and fitted values  
plot.ts(ts\_weather, col='blue', main="Original vs Fitted ARIMA", ylab="Temperature")  
lines(fitted(arima\_model), col='red')



**Implementation of the analysis** :

 **Data Import and Preprocessing:**

* The weather data is imported from a CSV file and includes variables like temperature, humidity, wind speed, and visibility.
* The Formatted.Date column is converted to a date format for time series analysis.

 **Time Series Conversion:**

* The temperature data is converted into a time series object (ts\_weather) for analysis, assuming daily frequency.

 **Stationarity Check:**

* The Augmented Dickey-Fuller (ADF) test is performed to check the stationarity of the data. The null hypothesis (H0) that the data is not stationary is rejected based on the p-value (< 0.05), indicating the data is stationary.

 **Modeling**:

* Various models (AR, MA, and ARIMA) are fitted to the time series data:
  + **AR Model**: The autoregressive model is fitted using the arima function with specified order.
  + **MA Model**: A moving average model is fitted similarly.
  + **ARIMA Model**: The ARIMA (Auto-Regressive Integrated Moving Average) model is fitted, which combines both AR and MA components.

 **Forecasting**:

* The ARIMA model is used to forecast future temperature values, and the results are plotted.

 **Plotting Results:**

* A plot is generated comparing the original temperature values with the fitted ARIMA model values, providing insights into the model's performance.