Practice-2

Harsh

17/05/2020

Question 1 : The built-in dataset USArrests contains statistics about violent crime rates in the US States. Determine which states are outliers in terms of murders. Outliers, for the sake of this question, are defined as values that are more than 1.5 standard deviations from the mean.

```r
head(USArrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

```r
library(data.table)

usarrest_data <- USArrests

states <- data.table("States" = state.name)
usarrest_data <- cbind(states,usarrest_data)

mean_murder <- mean(usarrest_data$Murder)
sd_murder <- sd(usarrest_data$Murder)

z_score <- abs((mean_murder-usarrest_data$Murder)/sd_murder)
z <- z_score > 1.5

outliers_states <- usarrest_data[z,1]
outliers_states
```

```
##            States
## 1:        Florida
## 2:        Georgia
## 3:      Louisiana
## 4:    Mississippi
## 5:   North Dakota
## 6: South Carolina
```

```r
# In this problem, first we import the data to usarrest_data variable. Since we don't have the state na
```

Question 2 : For the same dataset as in (1), is there a correlation between urban population and murder, i.e., as one goes up, does the other statistic as well? Comment on the strength of the correlation. Calculate the Pearson coefficient of correlation in R.

```r
cor.test(usarrest_data$UrbanPop, usarrest_data$Murder)
```

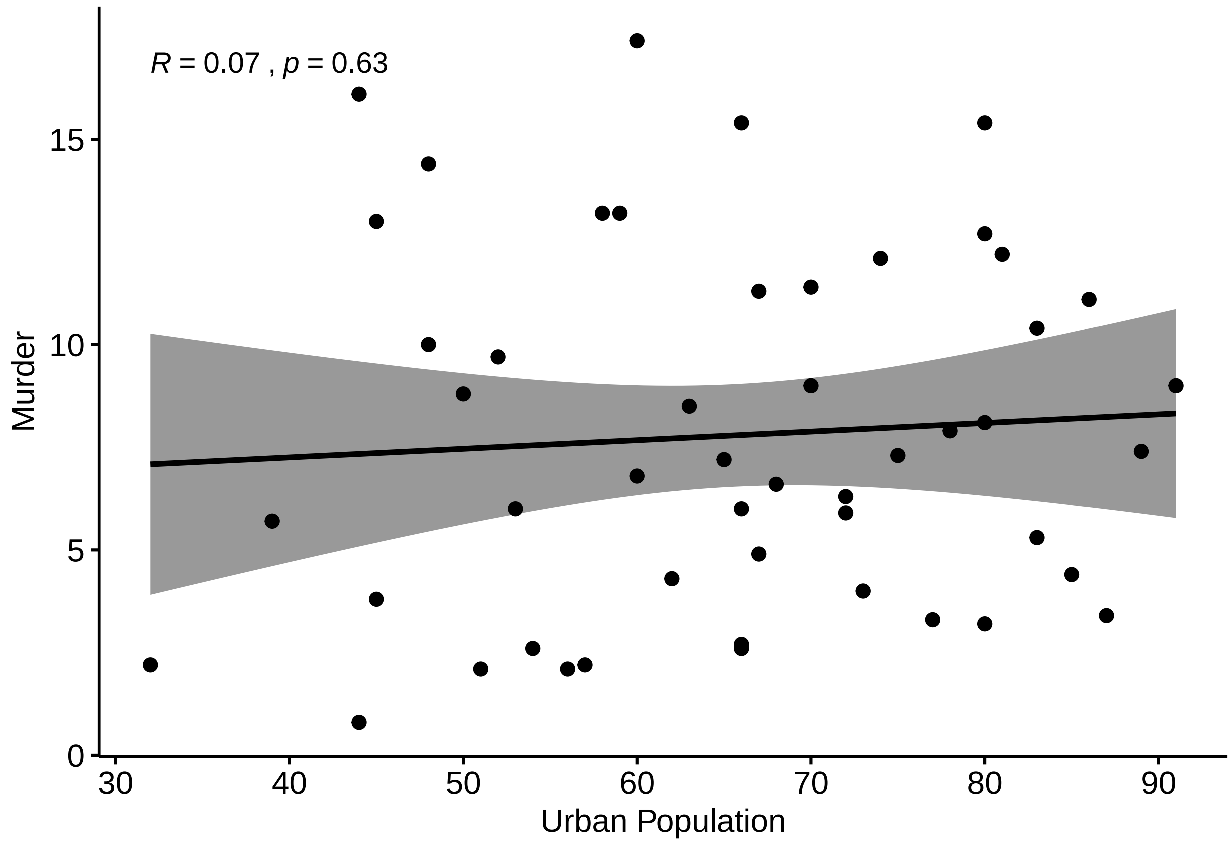
```
##
##  Pearson's product-moment correlation
##
## data:  usarrest_data$UrbanPop and usarrest_data$Murder
## t = 0.48318, df = 48, p-value = 0.6312
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.2128979  0.3413107
## sample estimates:
##        cor
## 0.06957262
```

```r
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 3.6.3
```

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

```r
ggscatter(usarrest_data, x = "UrbanPop" , y = "Murder" , add = "reg.line", conf.int = TRUE,
          cor.coef = TRUE, cor.method = "pearson", xlab = "Urban Population", ylab = "Murder")
```




```
# As we can see that the correlation between Urban population and Murder is very low (0.06). Because of
```

Question 3 : Based on the data on the growth of mobile phone use in Brazil (you'll need to copy the data and create a CSV that you can load into R or use the gsheet2tbl() function from the gsheet package), forecast phone use for the next time period using a 2-year weighted moving average (with weights of 5 for the most recent year, and 2 for other), exponential smoothing (alpha of 0.4), and linear regression trendline.

```
# install.packages("gsheet")
library(gsheet)
```

```
## Warning: package 'gsheet' was built under R version 3.6.3
```

```
mobile_data <- data.frame(gsheet2tbl("https://docs.google.com/spreadsheets/d/1tOnM9XceK4Ak8tzWQ2vDelWlJ

mobile_data <- mobile_data[-12,]


###############

# 2-year weighted average

n <- nrow(mobile_data)

last2 <- mobile_data[c(n,n-1), 2]
```

```r
weight <- c(5,2)

sw <- weight*last2

weighted_average <- sum(sw)/sum(weight)


###############

# Exponential Smoothing with alpha = 0.4

mobile_data_1 <- mobile_data

alpha <- 0.4

mobile_data_1$Ft <- 0
mobile_data_1$E <- 0
mobile_data_1$sqrdError <- 0

mobile_data_1$Ft[1] <- mobile_data_1[1,2]

# F(t) = F(t-1) + a * E(t-1)

for (i in 2:nrow(mobile_data_1)) {
  mobile_data_1$Ft[i] <- mobile_data_1$Ft[i-1] + alpha * mobile_data_1$E[i-1]
  mobile_data_1$E[i] <- mobile_data_1$Subscribers[i] - mobile_data_1$Ft[i]
  mobile_data_1$sqrdError[i] <- mobile_data_1$E[i] ^ 2
}

forecast_exponential_smoothing <- mobile_data_1$Ft[n] + alpha * mobile_data_1$E[n]


###############

# Linear Regression

mobile_data_2 <- mobile_data

model <- lm(mobile_data_2$Subscribers ~ mobile_data_2$Year)

summary(model)
```

```
##
## Call:
## lm(formula = mobile_data_2$Subscribers ~ mobile_data_2$Year)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -12307858  -9795553  -4238521   7402838  20622182
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -15710760    8041972  -1.954   0.0825 .
```

```
## mobile_data_2$Year  18276748    1185724  15.414  8.9e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12440000 on 9 degrees of freedom
## Multiple R-squared:  0.9635, Adjusted R-squared:  0.9594
## F-statistic: 237.6 on 1 and 9 DF,  p-value: 8.903e-08
```

```
print(model)
```

```
##
## Call:
## lm(formula = mobile_data_2$Subscribers ~ mobile_data_2$Year)
##
## Coefficients:
##        (Intercept)  mobile_data_2$Year
##          -15710760            18276748
```

```
forecast_linear_regression <- -15710760 + 18276748 * 12

###############


sprintf("2-year Moving Average : %s",weighted_average)
```

```
## [1] "2-year Moving Average : 194662700.142857"
```

```
sprintf("Forecast with Exponential Smoothing : %s",forecast_exponential_smoothing)
```

```
## [1] "Forecast with Exponential Smoothing : 165168213.62273"
```

```
sprintf("Forecast with linear regression : %s",forecast_linear_regression)
```

```
## [1] "Forecast with linear regression : 203610216"
```

```
# In this problem we tested 3 types of forecasting methods on the same mobile data. I have duplicated t
```

Question 4 : Calculate the squared error for each model, i.e., use the model to calculate a forecast for each given time period and then the squared error. Finally, calculate the average (mean) squared error for each model. Which model has the smallest mean squared error (MSE)?

```
mobile_data_3 <- mobile_data


###############

# MSE Calculation for Linear Regression method

mobile_data_3$F <- 0
mobile_data_3$absError <- 0
```

```r
mobile_data_3$sqrdError <- 0

for (i in 1:nrow(mobile_data_3)) {
  mobile_data_3$F[i] <- -15710760 + 18276748 * mobile_data_3$Year[i]
  mobile_data_3$absError[i] <- abs(mobile_data_3$Subscribers[i] - mobile_data_3$F[i])
  mobile_data_3$sqrdError[i] <- mobile_data_3$absError[i] ^ 2
}

###############

# MSE Calculation for Weighted Average method

mobile_data_4 <- mobile_data

mobile_data_4$Forecast <- 0
mobile_data_4$Error <- 0
mobile_data_4$sqrdError <- 0

mobile_data_4$Forecast[1] <- mobile_data_4$Subscribers[1]
mobile_data_4$Forecast[2] <- mobile_data_4$Subscribers[2]



for (i in 3:nrow(mobile_data_4)) {
  last2year <- mobile_data_4$Subscribers[c(i-1,i-2)]
  weight <- c(5,2)
  sw1 <- weight*last2year
  mobile_data_4$Forecast[i] <- sum(sw1)/sum(weight)
  mobile_data_4$Error[i] <- abs(mobile_data_4$Subscribers[i]-mobile_data_4$Forecast[i])
  mobile_data_4$sqrdError[i] <- mobile_data_4$Error[i] ^ 2
}

###############

# Calculation of MSE for all 3 models
MSE_lm <- mean(mobile_data_3$sqrdError)

MSE_es <- mean(mobile_data_1$sqrdError)

MSE_wa <- mean(mobile_data_4$sqrdError)

sprintf("Mean Squared Error for Linear Regression : %s",MSE_lm)
```

```
## [1] "Mean Squared Error for Linear Regression : 126534746000250"
```

```r
sprintf("Mean Squared Error for Exponential Smoothing : %s",MSE_es)
```

```
## [1] "Mean Squared Error for Exponential Smoothing : 1473838293531657"
```

```r
sprintf("Mean Squared Error for 2-year Weighted Average : %s",MSE_wa)
```

```
## [1] "Mean Squared Error for 2-year Weighted Average : 544143882735677"
```

```r
# Table to observe the minimum MSE
model <- c("2-year Weighted Average", "Exponential Smoothing", "Linear Regression")
MSE <- c(MSE_wa,MSE_es,MSE_lm)
min_MSE <- data.frame(model,MSE)

min_MSE[order(MSE),]
```

```
##                     model          MSE
## 3       Linear Regression 1.265347e+14
## 1 2-year Weighted Average 5.441439e+14
## 2   Exponential Smoothing 1.473838e+15
```

```r
# In this problem we calculated MSE for all 3 forecasts. Since we calculated the errors for exponential
```

Question 5 : Calculate a weighted average forecast by averaging out the three forecasts calculated in (3) with the following weights: 4 for trend line, 2 for exponential smoothing, 1 for weighted moving average. Remember to divide by the sum of the weights in a weighted average.

```r
#Calculation of weighted average forecast of the above 3 forecasts
values <- c(forecast_linear_regression,forecast_exponential_smoothing,weighted_average)

weights <- c(4,2,1)

wv <- values*weights

weighted_average_forecast <- sum(wv)/sum(weights)
weighted_average_forecast
```

```
## [1] 191348570
```

```r
# Here we created a new variable called values to store the output of the 3 forecasts which we calculat
```