PeerReview_2_MinxinCheng

0. load packages

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```r
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------------------------- ti

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v stringr 1.4.0
## v tidyr   1.1.2     v forcats 0.5.0
## v readr   1.3.1

## -- Conflicts --------------------------------------------------------------------------------- tidyvers
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggpubr)
library(gsheet)
library(forecast)
```

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

##
## Attaching package: 'forecast'

## The following object is masked from 'package:ggpubr':
##
##     gghistogram
```

0. Read in data sets

```r
data(USArrests)
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
USArrests <- USArrests %>%
  rownames_to_column("State")
```

1. 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
# check if there is any missing data
any(is.na(USArrests))
```

```
## [1] FALSE
```

```r
# check and print out outliers
outlier <- USArrests %>%
  select(State, Murder) %>%
  mutate(meanMurder = mean(Murder),
         sdMurder = sd(Murder),
         checkOutlier = (abs(Murder - meanMurder) / sdMurder)) %>%
  filter(checkOutlier > 1.5)
outlier
```

```
##            State Murder meanMurder sdMurder checkOutlier
## 1        Florida   15.4      7.788  4.35551     1.747671
## 2        Georgia   17.4      7.788  4.35551     2.206860
## 3      Louisiana   15.4      7.788  4.35551     1.747671
## 4    Mississippi   16.1      7.788  4.35551     1.908387
## 5   North Dakota    0.8      7.788  4.35551     1.604405
## 6 South Carolina   14.4      7.788  4.35551     1.518077
```

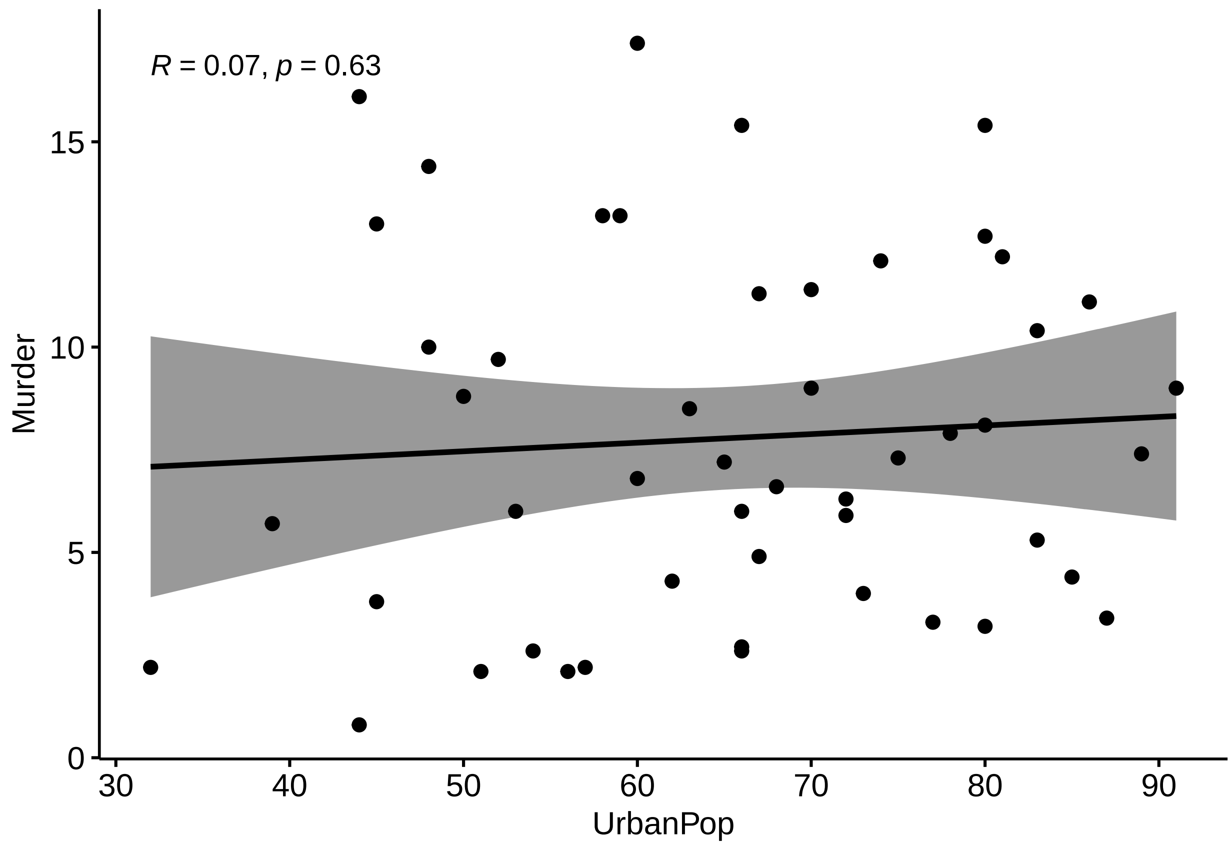
As the table, six states' murder data can be defined as outliers, they are: Florida, Georgia, Louisiana, Mississippi, North Dakota, and South Carolina.

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

```r
# visualize in order to check the relationship between these variables
ggscatter(USArrests,
          x = "UrbanPop",
          y = "Murder",
```

```
         add = "reg.line",
         conf.int = TRUE,
         cor.coef = TRUE,
         cor.method = "pearson")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



From the visualization above, the relationship between urban population and murder is linear. Then test the normality.

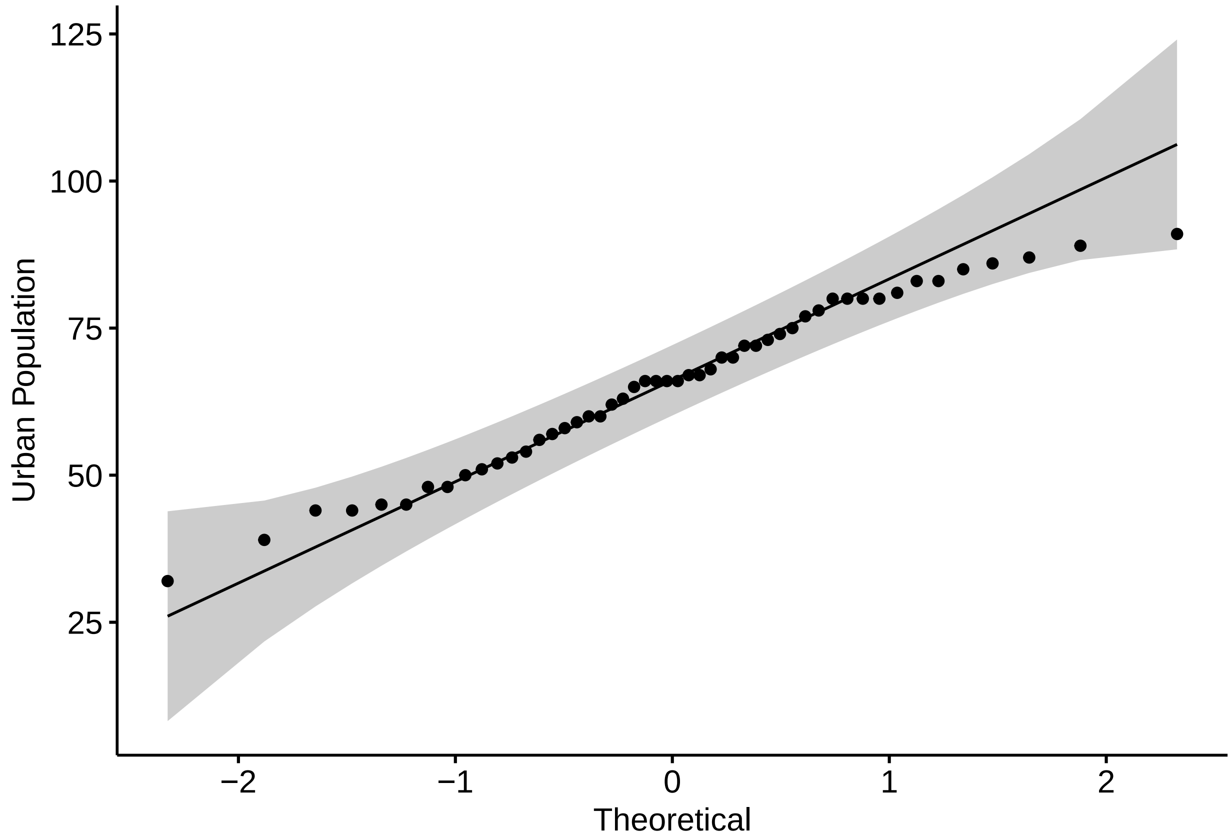
```
# run shapiro test to check the normality
shapiro.test(USArrests$UrbanPop)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  USArrests$UrbanPop
## W = 0.97714, p-value = 0.4385
```

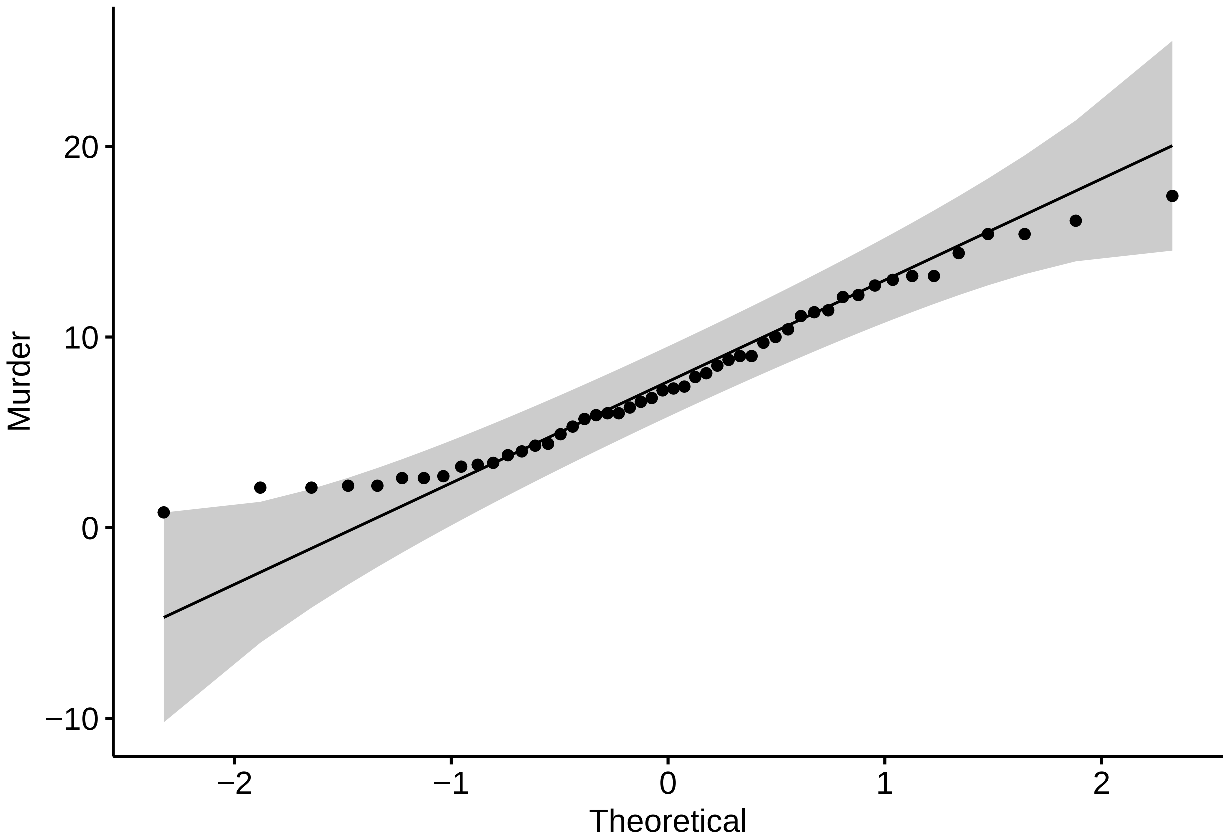
```
shapiro.test(USArrests$Murder)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  USArrests$Murder
## W = 0.95703, p-value = 0.06674
```

```r
# q-q plot to visualize
ggqqplot(USArrests$UrbanPop,
         ylab = "Urban Population")
```




```r
ggqqplot(USArrests$Murder,
         ylab = "Murder")
```



Results from Shapiro test (p = 0.4385 for urban population and p = 0.06674 for murder) indicated there is no strong reason to reject the null hypothesis, therefore, these two variables are normally distributed. Q-Q plots also supported the result.

```
# run Pearson test
cor.test(USArrests$UrbanPop,
         USArrests$Murder,
         method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  USArrests$UrbanPop and USArrests$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
```

From Pearson test's result, p value is 0.6312, indicates these two variables are not strongly correlated. There is almost no association between urban population and murder (estimated correlation 0.06957262).

3. Based on the data (the growth of mobile phone use in Brazil), 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.

3.0. read in data set

```
# read in data set
url <- 'https://docs.google.com/spreadsheets/d/1tOnM9XceK4Ak8tzWQ2vDelWlJexzJiS3LbT6MN6_rW0/edit?usp=sha

# duplicate the data set for each model
phoneDataModel1 <- gsheet2tbl(url, sheetid = NULL)
phoneDataModel2 <- gsheet2tbl(url, sheetid = NULL)
phoneDataModel3 <- gsheet2tbl(url, sheetid = NULL)
phoneDataModel1
```

```
## # A tibble: 12 x 2
##     Year Subscribers
##    <dbl>       <dbl>
##  1     1    23188171
##  2     2    28745769
##  3     3    34880964
##  4     4    46373266
##  5     5    65605000
##  6     6    86210336
##  7     7    99918621
##  8     8   120980103
##  9     9   150641403
## 10    10   173959368
## 11    11   202944033
## 12    12          NA
```

3.1. 2-year weighted moving average model

```
# get the last 2 years data and calculate the weighted value
recentYears <- phoneDataModel1 %>%
  filter(Year == 10 | Year == 11) %>%
  mutate(weighted = case_when(Year == 10 ~ Subscribers * 2,
                              Year == 11 ~ Subscribers * 5))
recentYears
```

```
## # A tibble: 2 x 3
##    Year Subscribers   weighted
##   <dbl>       <dbl>      <dbl>
## 1    10   173959368  347918736
## 2    11   202944033 1014720165
```

```
# calculate the estimated Subscribers for year 12
phoneDataModel1$Subscribers[12] <- sum(recentYears$weighted) / (5 + 2)
phoneDataModel1
```

```
## # A tibble: 12 x 2
##     Year Subscribers
##    <dbl>       <dbl>
##  1     1   23188171
##  2     2   28745769
##  3     3   34880964
```

```
## 4     4   46373266
## 5     5   65605000
## 6     6   86210336
## 7     7   99918621
## 8     8  120980103
## 9     9  150641403
## 10   10  173959368
## 11   11  202944033
## 12   12  194662700.
```

Based on moving average model, the estimation of 12 years subscribers' value is 194662700.

3.2. Exponential smoothing model

```
# with package: estimate year 12 Subscriber's value
#estModel2 <- ses(phoneDataModel2$Subscribers[1:11],
#                 h = 1,
#                 alpha = 0.4,
#                 initial = "simple")
#
#phoneDataModel2$Subscribers[12] <- estModel2$mean
#phoneDataModel2

# without package
phoneDataModel2$Estimation <- 0
phoneDataModel2$Error <- 0
phoneDataModel2$Estimation[1] <- phoneDataModel2$Subscribers[1]

for (n in 2:11){

  phoneDataModel2$Estimation[n] <- phoneDataModel2$Estimation[n - 1] +
    0.4 * phoneDataModel2$Error[n - 1]
  phoneDataModel2$Error[n] <- phoneDataModel2$Subscribers[n] - phoneDataModel2$Estimation[n]

}

# eistimate subscribers value of year 12
phoneDataModel2$Subscribers[12] <- phoneDataModel2$Estimation[11] +
    0.4 * phoneDataModel2$Error[11]
phoneDataModel2
```

```
## # A tibble: 12 x 4
##     Year Subscribers Estimation     Error
##    <dbl>       <dbl>      <dbl>     <dbl>
## 1      1    23188171   23188171         0
## 2      2    28745769   23188171   5557598
## 3      3    34880964   25411210.  9469754.
## 4      4    46373266   29199112. 17174154.
## 5      5    65605000   36068773. 29536227.
## 6      6    86210336   47883264. 38327072.
## 7      7    99918621   63214093. 36704528.
## 8      8   120980103   77895904. 43084199.
## 9      9   150641403   95129584. 55511819.
## 10    10   173959368  117334311. 56625057.
```

```
## 11    11  202944033  139984334. 62959699.
## 12    12  165168214.         0         0
```

Based on exponential smoothing model, the estimation of 12 years subscribers' value is 165168214.

3.3 Linear regression trendline model

```
# run linear regression
regPhoneData <- lm(phoneDataModel3$Subscribers[1:11] ~ phoneDataModel3$Year[1:11])
summary(regPhoneData)
```

```
##
## Call:
## lm(formula = phoneDataModel3$Subscribers[1:11] ~ phoneDataModel3$Year[1:11])
##
## 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 .
## phoneDataModel3$Year[1:11]  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
```

```
# calculated the estimated value
phoneDataModel3$Subscribers[12] <- -15710760 + 18276748 * (12)
phoneDataModel3
```

```
## # A tibble: 12 x 2
##     Year Subscribers
##    <dbl>       <dbl>
##  1     1    23188171
##  2     2    28745769
##  3     3    34880964
##  4     4    46373266
##  5     5    65605000
##  6     6    86210336
##  7     7    99918621
##  8     8   120980103
##  9     9   150641403
## 10    10   173959368
## 11    11   202944033
## 12    12   203610216
```

Based on linear regression trendline model, the estimation of 12 years subscribers' value is 203610216.

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)?

4.1. Weighted moving average model

```
# create 3 empty columns for estimations, errors, and squared errors
phoneDataModel1$Estimation <- 0
phoneDataModel1$Error <- 0
phoneDataModel1$SqError <- 0

# for loop to calculate estimated values
for (i in 3:12) {

phoneDataModel1$Estimation[i] <- sum(2 * (phoneDataModel1[(i - 2), 2]) +
                                       5 * (phoneDataModel1[(i - 1), 2])) / (2 + 5)
phoneDataModel1$Error[i] <- phoneDataModel1$Subscribers[i] - phoneDataModel1$Estimation[i]
phoneDataModel1$SqError[i] <- phoneDataModel1$Error[i] ^ 2

}

# print the table
phoneDataModel1
```

```
## # A tibble: 12 x 5
##     Year Subscribers Estimation     Error SqError
##    <dbl>       <dbl>      <dbl>     <dbl>   <dbl>
##  1     1    23188171          0         0  0.
##  2     2    28745769          0         0  0.
##  3     3    34880964   27157884.  7723080. 5.96e13
##  4     4    46373266   33128051. 13245215. 1.75e14
##  5     5    65605000   43089751. 22515249. 5.07e14
##  6     6    86210336   60110219. 26100117. 6.81e14
##  7     7    99918621   80323097. 19595524. 3.84e14
##  8     8   120980103   96001968. 24978135. 6.24e14
##  9     9   150641403  114962537. 35678866. 1.27e15
## 10    10   173959368  142166746. 31792622. 1.01e15
## 11    11   202944033  167297092. 35646941. 1.27e15
## 12    12   194662700. 194662700.         0  0.
```

```
# calculate the mean squared error
MSE1 <- mean(phoneDataModel1$SqError)
MSE1
```

```
## [1] 4.987986e+14
```

4.2. Exponential smoothing model

```
phoneDataModel2$SqError <- 0

# for loop to calculate estimated values
for (j in 2:11){
```

```r
  phoneDataModel2$Error[j] <- phoneDataModel2$Subscribers[j] - phoneDataModel2$Estimation[j]
  phoneDataModel2$SqError[j] <- phoneDataModel2$Error[j] ^ 2

}

phoneDataModel2
```

```
## # A tibble: 12 x 5
##     Year Subscribers Estimation     Error SqError
##    <dbl>       <dbl>      <dbl>     <dbl>   <dbl>
##  1     1    23188171   23188171         0  0.
##  2     2    28745769   23188171   5557598  3.09e13
##  3     3    34880964   25411210.  9469754. 8.97e13
##  4     4    46373266   29199112. 17174154. 2.95e14
##  5     5    65605000   36068773. 29536227. 8.72e14
##  6     6    86210336   47883264. 38327072. 1.47e15
##  7     7    99918621   63214093. 36704528. 1.35e15
##  8     8   120980103   77895904. 43084199. 1.86e15
##  9     9   150641403   95129584. 55511819. 3.08e15
## 10    10   173959368  117334311. 56625057. 3.21e15
## 11    11   202944033  139984334. 62959699. 3.96e15
## 12    12   165168214.         0         0  0.
```

```r
MSE2 <- mean(phoneDataModel2$SqError)
MSE2
```

```
## [1] 1.351018e+15
```

4.3. Linear regression trendline model MSE

```r
# create 3 empty columns for estimations, errors, and squared errors
phoneDataModel3$Estimation <- 0
phoneDataModel3$Error <- 0
phoneDataModel3$SqError <- 0

# for loop to calculate estimated values
for (k in 1:12){

  phoneDataModel3$Estimation[k] <- -15710760 + 18276748 * phoneDataModel3$Year[k]
  phoneDataModel3$Error[k] <- phoneDataModel3$Subscribers[k] - phoneDataModel3$Estimation[k]
  phoneDataModel3$SqError[k] <- phoneDataModel3$Error[k] ^ 2

}

phoneDataModel3
```

```
## # A tibble: 12 x 5
##     Year Subscribers Estimation     Error SqError
##    <dbl>       <dbl>      <dbl>     <dbl>   <dbl>
##  1     1    23188171    2565988  20622183 4.25e14
##  2     2    28745769   20842736   7903033 6.25e13
```

```
## 3      3    34880964   39119484  -4238520 1.80e13
## 4      4    46373266   57396232 -11022966 1.22e14
## 5      5    65605000   75672980 -10067980 1.01e14
## 6      6    86210336   93949728  -7739392 5.99e13
## 7      7    99918621  112226476 -12307855 1.51e14
## 8      8   120980103  130503224  -9523121 9.07e13
## 9      9   150641403  148779972   1861431 3.46e12
## 10    10   173959368  167056720   6902648 4.76e13
## 11    11   202944033  185333468  17610565 3.10e14
## 12    12   203610216  203610216         0 0.
```

```r
MSE3 <- mean(phoneDataModel3$SqError)
MSE3
```

```
## [1] 1.159902e+14
```

4.4. Compare MSEs

```r
# find out the minimum mean squared value
minMSE <- min(MSE1, MSE2, MSE3)
minMSE
```

```
## [1] 1.159902e+14
```

From calculations above, Weighted Moving Average model has mean squared error 4.987986e+14, Exponential Smoothing model's mean squared error 1.351018e+15, and Linear Regression Trendline model 1.159902e+14. Linear Regression Trendline model has the smallest mean squared error.

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 weights in a weighted average.

```r
weightedAve <- sum(1 * phoneDataModel1[12, 2] +
                     2 * phoneDataModel2[12, 2] +
                     4 * phoneDataModel3[12, 2]) / (1 + 2 + 4)
weightedAve
```

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