SRM-project_Roll-no_432.R

Priyanshu Mehta

2021-11-25

```
##Graduation Tests in R
##Project on Statistical and Risk Modelling 1
##Name - Priyanshu Mehta, Roll No. 432

##Q1-Read in the data file as a data table & fill in the entries for the
CRUDE column in your table
#importing the Graduation.csv dataset
Grad=read.table("C:/Users/Priyanshu Mehta/OneDrive/Desktop/SRM
proj/Graduation.csv",header=T)

#LOADING THE REQUIRED LIBRARIES
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## Warning: package 'dplyr' was built under R version 4.1.1

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

#Computing the important EDA commands

View(Grad)

head(Grad,3)

##   AGE.ETR.DEATHS.CRUDE.GRADUATED.EXPECTED.ZX
## 1                         25,78500,24,0,0,0,0
## 2                         26,80425,24,0,0,0,0
## 3                         27,81975,24,0,0,0,0

#the output shows all the values in a single cell which is not what was
required out of us so,
#we use the "sep" function to separate the columns by commas instead of
spaces.
```

```
Grad = read.table("C:/Users/Priyanshu Mehta/OneDrive/Desktop/SRM
proj/Graduation.csv",sep=",",header=T)
head(Grad,3)
```

```
##   AGE   ETR DEATHS CRUDE GRADUATED EXPECTED ZX
## 1  25 78500     24     0         0        0  0
## 2  26 80425     24     0         0        0  0
## 3  27 81975     24     0         0        0  0
```

```
tail(Grad,3)
```

```
##    AGE    ETR DEATHS CRUDE GRADUATED EXPECTED ZX
## 49  73 130475   5454     0         0        0  0
## 50  74 129550   6453     0         0        0  0
## 51  75 129400   6453     0         0        0  0
```

```
dim(Grad)
```

```
## [1] 51  7
```

```
glimpse(Grad)
```

```
## Rows: 51
## Columns: 7
## $ AGE       <int> 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
39, ~
## $ ETR       <int> 78500, 80425, 81975, 83725, 84875, 85075, 85275, 86250,
8725~
## $ DEATHS    <int> 24, 24, 24, 24, 72, 48, 120, 24, 72, 72, 24, 48, 24,
168, 12~
## $ CRUDE     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, ~
## $ GRADUATED <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, ~
## $ EXPECTED  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, ~
## $ ZX        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, ~
```

```
colnames(Grad)
```

```
## [1] "AGE"       "ETR"       "DEATHS"    "CRUDE"     "GRADUATED" "EXPECTED"
## [7] "ZX"
```

```
summary(Grad)
```

```
##       AGE             ETR             DEATHS         CRUDE     GRADUATED
##  Min.   :25.0   Min.   : 78500   Min.   :  24   Min.   :0   Min.   :0
##  1st Qu.:37.5   1st Qu.: 96988   1st Qu.:  96   1st Qu.:0   1st Qu.:0
##  Median :50.0   Median :112725   Median : 408   Median :0   Median :0
##  Mean   :50.0   Mean   :112215   Mean   :1300   Mean   :0   Mean   :0
```

```
##  3rd Qu.:62.5   3rd Qu.:129475   3rd Qu.:1850   3rd Qu.:0   3rd Qu.:0
##  Max.   :75.0   Max.   :136625   Max.   :6453   Max.   :0   Max.   :0
##     EXPECTED       ZX
##  Min.   :0   Min.   :0
##  1st Qu.:0   1st Qu.:0
##  Median :0   Median :0
##  Mean   :0   Mean   :0
##  3rd Qu.:0   3rd Qu.:0
##  Max.   :0   Max.   :0
```

```
  ##COMMENT: after deep analysis of EDA outputs, we realize that columns -
{CRUDE, GRADUATED, EXPECTED and ZX}
  #consist of no values and so we must compute the same and fill them in

#filling in the entries for the CRUDE column
#crude rates = no. of deaths/no. of exposed to risk
Grad$CRUDE=round(Grad$DEATHS/Grad$ETR,8)
Grad$CRUDE
```

```
##  [1] 0.00030573 0.00029841 0.00029277 0.00028665 0.00084831 0.00056421
##  [7] 0.00140721 0.00027826 0.00082521 0.00081540 0.00026608 0.00051892
## [13] 0.00025151 0.00170472 0.00120271 0.00242119 0.00217742 0.00141837
## [19] 0.00114805 0.00022414 0.00197938 0.00328992 0.00286042 0.00108918
## [25] 0.00150437 0.00383234 0.00354013 0.00507507 0.00584878 0.00729431
## [31] 0.00412545 0.00536779 0.00683815 0.00658537 0.00960186 0.01202491
## [37] 0.01148507 0.01494390 0.01285953 0.01765289 0.01442635 0.02083027
## [43] 0.02128177 0.02491999 0.03201216 0.03426582 0.03037827 0.03923121
## [49] 0.04180111 0.04981088 0.04986862
```

```
#selecting an arbitrary decimal rounding figure of 8 d.p.

##Q2-Use Gompertz law to fill entries in the GRADUATED column in the data.
#Gompertz law = mu(x)=B*c^x
#We convert the same and use log(mu(x))=log(B)+x*log(c), reason in the report
gompertz_law=lm(log(Grad$CRUDE)~Grad$AGE)
gompertz_law
```

```
##
## Call:
## lm(formula = log(Grad$CRUDE) ~ Grad$AGE)
##
## Coefficients:
## (Intercept)     Grad$AGE
##    -11.0009       0.1063
```

```
#extracting the coefficient values for further use with fn 'coeff'
coef(gompertz_law)
```

```
## (Intercept)    Grad$AGE
## -11.0008670   0.1062977
```

```r
B=exp(as.numeric(coef(gompertz_law)))[1]
C=exp(as.numeric(coef(gompertz_law)))[2]
c(B,C)
```

```
## [1] 1.668723e-05 1.112153e+00
```

```r
#make note of the difference is concatenating 'c' and the variable 'C'

#Filling entries in the GRADUATED column
Grad$GRADUATED<-round(B*C^Grad$AGE,8)
Grad$GRADUATED
```

```
##  [1] 0.00023796 0.00026464 0.00029432 0.00032733 0.00036404 0.00040487
##  [7] 0.00045028 0.00050078 0.00055695 0.00061941 0.00068888 0.00076614
## [13] 0.00085206 0.00094762 0.00105390 0.00117210 0.00130355 0.00144975
## [19] 0.00161234 0.00179317 0.00199428 0.00221794 0.00246669 0.00274334
## [25] 0.00305101 0.00339319 0.00377375 0.00419699 0.00466769 0.00519119
## [31] 0.00577339 0.00642090 0.00714102 0.00794190 0.00883261 0.00982321
## [37] 0.01092492 0.01215018 0.01351285 0.01502836 0.01671383 0.01858834
## [43] 0.02067307 0.02299162 0.02557020 0.02843797 0.03162737 0.03517447
## [49] 0.03911939 0.04350674 0.04838614
```

```r
#selecting an arbitrary decimal rounding figure of 8 d.p.

##Q3-Check for smoothness by applying the third differences to the crude and graduated rates
#we first prepare a function to find the first differences between crude and graduated rates
diff1=function(v)v[-1]-v[-length(v)]

#computing third differences with the aid of above
diff_crude=round(diff1(diff1(diff1(Grad$CRUDE)))*10^8,0)
diff_grad=round(diff1(diff1(diff1(Grad$GRADUATED)))*10^8,0)
diff_crude
```

```
##  [1]     -216    56826  -141354   197286  -309905   364785  -223266
1725
##  [9]   134167  -132241   224087  -367584   367571  -318274    94697
100401
## [17]  -114232   333274  -312385  -129534    39830   352817   -27365  -
453296
## [25]   444733  -258838   143305  -528621   902559  -418318  -195116
499241
## [33]  -386271  -236945   696156  -954187  1242093 -1489763  1765036 -
1558288
## [41]   913914    26723  -829246  -130270  1888170 -1902353  1172291 -
1339190
```

```r
diff_grad
```

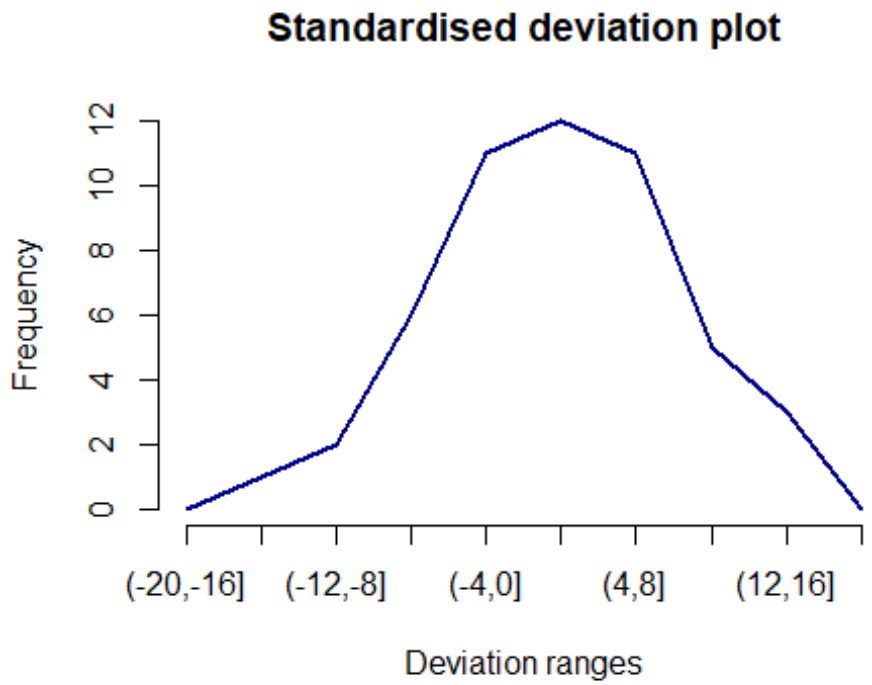
```
##  [1]   33   37   42   46   51   58   62   72   78   87   98  108  120  133
150
## [16]  164  185  204  227  254  281  312  349  387  430  478  534  590  661
730
## [31]  815  907 1006 1122 1244 1386 1543 1712 1908 2118 2360 2621 2916 3244
3607
## [46] 4012 4461 4962
```

```
#reasoning of using diff1 thrice in the report

#Binding together [combining] the difference values of crude and graduation
along with age for better representation
cbind(Grad$AGE[Grad$AGE<=72],diff_crude,diff_grad)
```

```
##          diff_crude diff_grad
##  [1,] 25       -216        33
##  [2,] 26      56826        37
##  [3,] 27    -141354        42
##  [4,] 28     197286        46
##  [5,] 29    -309905        51
##  [6,] 30     364785        58
##  [7,] 31    -223266        62
##  [8,] 32       1725        72
##  [9,] 33     134167        78
## [10,] 34    -132241        87
## [11,] 35     224087        98
## [12,] 36    -367584       108
## [13,] 37     367571       120
## [14,] 38    -318274       133
## [15,] 39      94697       150
## [16,] 40     100401       164
## [17,] 41    -114232       185
## [18,] 42     333274       204
## [19,] 43    -312385       227
## [20,] 44    -129534       254
## [21,] 45      39830       281
## [22,] 46     352817       312
## [23,] 47     -27365       349
## [24,] 48    -453296       387
## [25,] 49     444733       430
## [26,] 50    -258838       478
## [27,] 51     143305       534
## [28,] 52    -528621       590
## [29,] 53     902559       661
## [30,] 54    -418318       730
## [31,] 55    -195116       815
## [32,] 56     499241       907
## [33,] 57    -386271      1006
## [34,] 58    -236945      1122
## [35,] 59     696156      1244
```

```
## [36,] 60    -954187      1386
## [37,] 61    1242093      1543
## [38,] 62   -1489763      1712
## [39,] 63    1765036      1908
## [40,] 64   -1558288      2118
## [41,] 65     913914      2360
## [42,] 66      26723      2621
## [43,] 67    -829246      2916
## [44,] 68    -130270      3244
## [45,] 69    1888170      3607
## [46,] 70   -1902353      4012
## [47,] 71    1172291      4461
## [48,] 72   -1339190      4962

#considered 72 instead of 75 because of 3 level of differences
View(cbind(Grad$AGE[Grad$AGE<=72],diff_crude,diff_grad))
Mod(max(diff_crude))

## [1] 1888170

Mod(min(diff_crude))

## [1] 1902353

#max value = 1902353 for crude rates

Mod(max(diff_grad))

## [1] 4962

Mod(min(diff_grad))

## [1] 33

#max value = 4961 for graduated rates

#COMMENT for both the considerations in the report

##Q4-Calculate the values in EXPECTED and ZX values in the table. Hence,
perform a chi-squared test to
#check goodness of fit between DEATHS and EXPECTED. You should specify the
degrees of freedom used.
#Expected value = graduated rate * exposed to risk
#Zx(Individual Standardised deviation)=(no.of actual deaths-no.of expected
deaths)/no.of expected deaths^(1/2)
Grad$EXPECTED=round(Grad$GRADUATED*Grad$ETR,4)
Grad$ZX<-round((Grad$DEATHS-Grad$EXPECTED)/sqrt(Grad$EXPECTED),4)
head(Grad)

##   AGE   ETR DEATHS      CRUDE  GRADUATED EXPECTED      ZX
## 1  25 78500     24 0.00030573 0.00023796  18.6799  1.2309
## 2  26 80425     24 0.00029841 0.00026464  21.2837  0.5888
```

```
## 3  27 81975     24 0.00029277 0.00029432  24.1269 -0.0258
## 4  28 83725     24 0.00028665 0.00032733  27.4057 -0.6506
## 5  29 84875     72 0.00084831 0.00036404  30.8979  7.3943
## 6  30 85075     48 0.00056421 0.00040487  34.4443  2.3097

#performing the chi-squared test between DEATHS and EXPECTED
chi2=sum((Grad$ZX)^2)
#degrees of freedom used =  49 degrees of freedom  {51 df - 2df [lost due to parameters]}
#Chi-squared value = 2204.468

##Alternate method
chi2=sum((Grad$DEATHS-Grad$EXPECTED)^2/Grad$EXPECTED)


##Q5A-Perform the standardised deviations test on the individual deviations and comment on what follows
test_st_dev=table(cut(Grad$ZX, breaks = seq.int(from = -20,to = 20, by= 4)))
#the above code breaks the standardised deviation values with 4 deviation intervals
plot(test_st_dev, main = "Standardised deviation plot", ylab = "Frequency", xlab ="Deviation ranges", type = "l", col = "dark blue")
```




```
  ##COMMENT: -
#I.The graph is quite dispersed relative to a standard normal dist. graph, with several values higher than magnitude of 10.
#II.The values of the absolute deviations are pretty high as compared to the
```

```
expected value.
#III.The lower bound is approx. -2 and the upper bound is approx. 6. IQR is
8.
    #Also, there are a couple of outliers on the lower side but none on the
upper side.
#IV.The Graph is positively skewed. It looks like it is centered about 2.
#V.The graduated rates do not represent the underlying mortality rates with
accuracy.


#Q5B-Sign test
test_sign=sign(Grad$ZX)
table(test_sign)
```

```
## test_sign
## -1  1
## 20 31
```

```
dbinom(31,51,0.5)
```

```
## [1] 0.03443253
```

```
#COMMENT - Since it's a two tailed test, at 5% significance level, we fail to
reject the NULL hypothesis that the data is a
#true representation of the underlying mortality rates.


#Q5C Cumulative deviations test
total_observed=sum(Grad$DEATHS)
total_expected=sum(Grad$EXPECTED)
z=(total_observed-total_expected)/total_expected
pnorm(-z)
```

```
## [1] 0.4697666
```

```
#COMMENT - At 5% significance level, we fail to reject the NULL hypothesis
that the data is a
#true representation of the underlying mortality rates.


#Q5D Serial Correlations test
library(EnvStats)
```

```
## Warning: package 'EnvStats' was built under R version 4.1.2
```

```
##
## Attaching package: 'EnvStats'
```

```
## The following objects are masked from 'package:stats':
##
##     predict, predict.lm
```

```
## The following object is masked from 'package:base':
##
##     print.default

serialCorrelationTest(Grad$ZX)

##
##  Rank von Neumann Test for Lag-1 Autocorrelation (Beta Approximation)
##
## data:  Grad$ZX
## RVN = 1.8355, p-value = 0.5563
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.1477136

#at 5% significance level, we fail to reject the NULL hypothesis that the
data is a
#true representation of the underlying mortality rates.
```