IMS Inleveropdracht 3

by Gijs Turkenburg (s3573532) and Sam Noordam (s3680657)

2023-11-06

We want to check whether an increase of the minimum wage has an effect on the number of: a) business closures, b) employees. We therefore look at the dataset FastFood.Rdata of fast-food stores in New Jersey and Pennsylvania. It contains the number of FTE (full-time equivalent) employees at 391 stores in March 1992 and December 1992. If a restaurant reports 0 FTE it means that it is closed. The first column indicates where the store is located (0=Pennsylvania, 1=New Jersey). New Jersey raised the minimum wage on April 1, 1992, from 4.25 to 5.05 per hour. In Pennsylvania the minimum wage remained at 4.25 over that time period. Choose suitable (parametric or non-parametric) statistical models for the data and apply hypothesis tests to answer the initial questions. Do not forget to formulate null- and alternative hypotheses and to justify your test choices.

Introduction

...

Organizing the data

The available data is first organised so it can be used to calculate with. If further data is to be added it can be placed anywhere in the data file, the data is extracted based on the NewJersey column of elements.

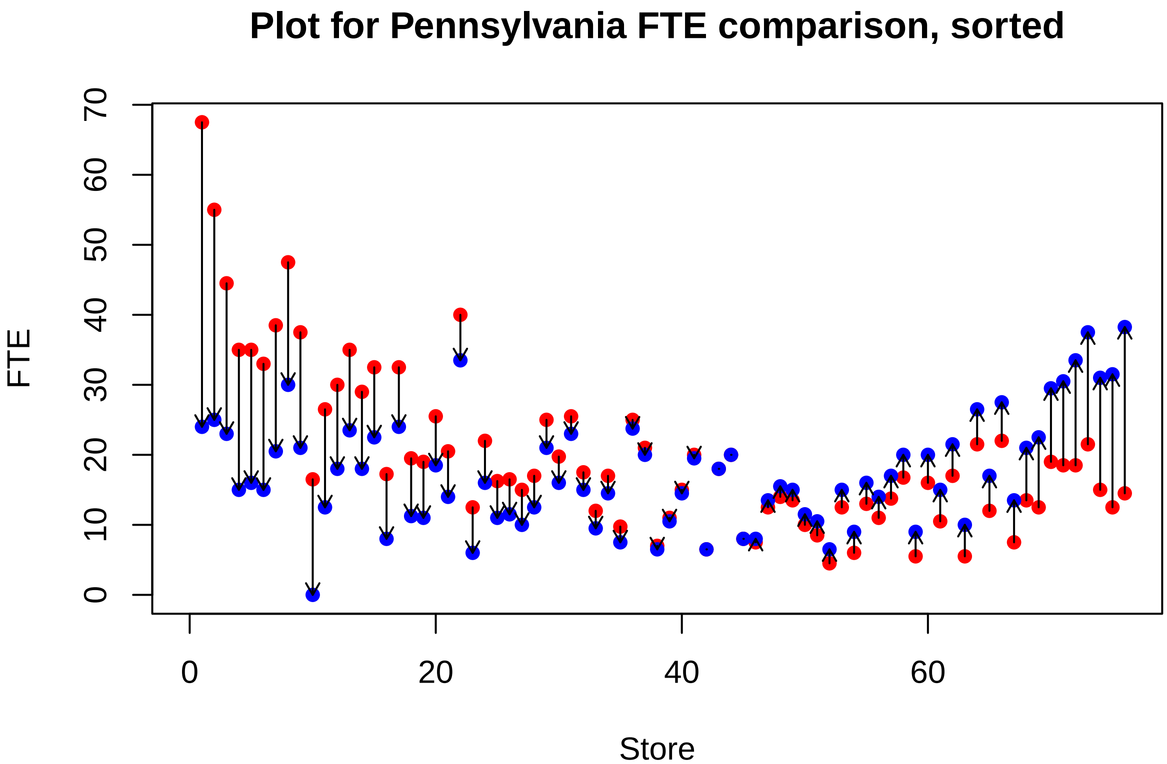
```r
#organising the data
setwd("/Users/gijs/Desktop/Universiteit Leiden/Programmas/SamCollab/R-collab/Inleveropdracht 3")
load("FastFood.Rdata") #loading the data

pennCounter <- 0 #amount of rows that contains information about Pennsylvania
njCounter <- 0 #amount of rows that contains information about New Jersey
dataFTE <- list(pennBefore = numeric (0), pennAfter = numeric (0), pennDiff = numeric(0),
                njBefore = numeric (0), njAfter = numeric (0), njDiff = numeric(0))
for(i in 1:length(FastFood$NewJersey)){#iterating over the list
  if (FastFood$NewJersey[i] == 0){#information about a Pennsylvania store
    pennCounter = pennCounter + 1
    dataFTE$pennBefore <- c(dataFTE$pennBefore, FastFood$FTEbefore[i])
    dataFTE$pennAfter <- c(dataFTE$pennAfter, FastFood$FTEafter[i])
    dataFTE$pennDiff <- c(dataFTE$pennDiff, FastFood$FTEafter[i] - FastFood$FTEbefore[i])
  }
  else{#information about a New Jersey store
    njCounter = njCounter + 1
    dataFTE$njBefore <- c(dataFTE$njBefore, FastFood$FTEbefore[i])
    dataFTE$njAfter <- c(dataFTE$njAfter, FastFood$FTEafter[i])
    dataFTE$njDiff <- c(dataFTE$njDiff, FastFood$FTEafter[i] - FastFood$FTEbefore[i])
  }
}
#an element in our dataset can be accessed by dataFTE$datatype[rownumber]
```

Visualising the data

Before the coice of a paramatric or a non-parametric model can be made, the data is visualised to obtain clues about it's properties. We start with a plot that compares the values of pennBefore and pennAfter.

```
#plot for the Pennsylvania data
max_penn <- max(dataFTE$pennBefore, dataFTE$pennAfter)
sort_index_penn <- order(dataFTE$pennDiff)
pennBeforeSorted <- dataFTE$pennBefore[sort_index_penn]
pennAfterSorted <- dataFTE$pennAfter[sort_index_penn]
pennDiffSorted <- dataFTE$pennDiff[sort_index_penn] #we also sort the differences to say consistent
plot(x = 1,
     type = "n",
     xlim = c(0, pennCounter),
     ylim = c(0, max_penn),
     pch = 16,
     xlab = "Store",
     ylab = "FTE",
     main = "Plot for Pennsylvania FTE comparison, sorted")
points(pennBeforeSorted, pch = 16, lwd = 1, col = "red")
points(pennAfterSorted, pch = 16, lwd = 1, col = "blue")
arrows(x0 = (1:pennCounter), y0 = pennBeforeSorted, x1 = (1:pennCounter), y1 = pennAfterSorted, col = "
```

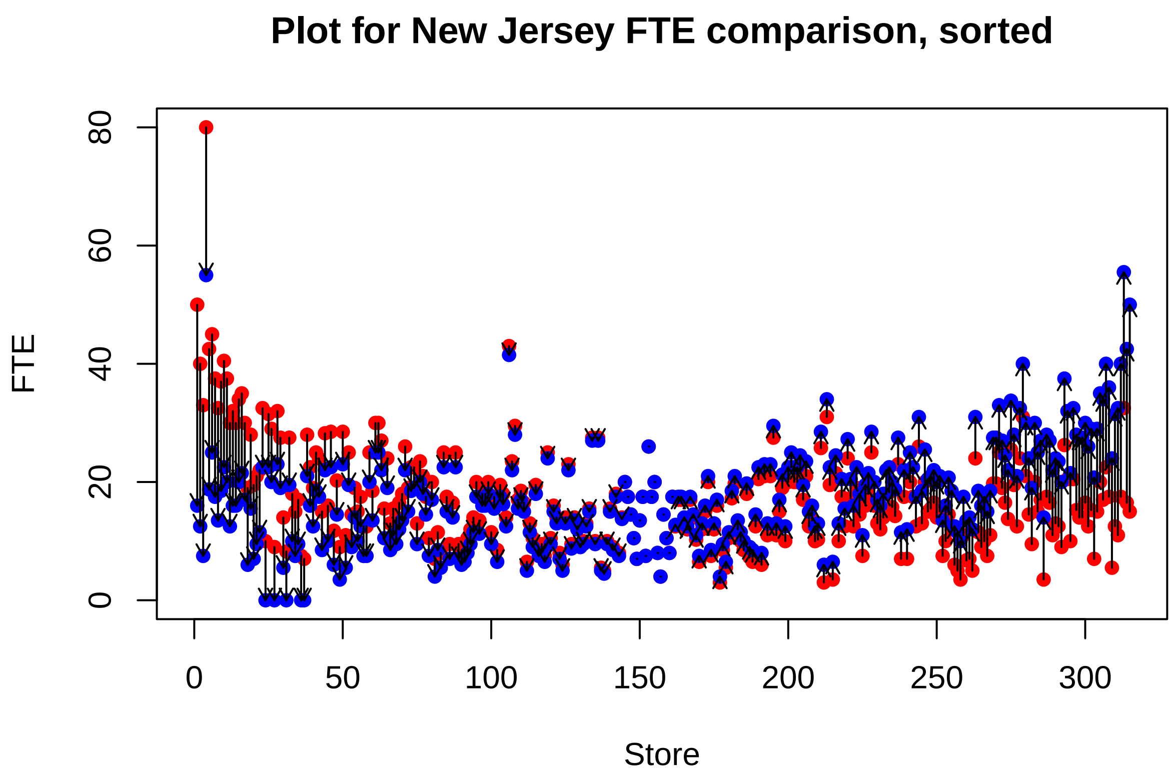
```
## Warning in arrows(x0 = (1:pennCounter), y0 = pennBeforeSorted, x1 =
## (1:pennCounter), : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:pennCounter), y0 = pennBeforeSorted, x1 =
## (1:pennCounter), : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:pennCounter), y0 = pennBeforeSorted, x1 =
## (1:pennCounter), : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:pennCounter), y0 = pennBeforeSorted, x1 =
## (1:pennCounter), : zero-length arrow is of indeterminate angle and so skipped
```




```
#plot for the New Jersey data
max_nj <- max(dataFTE$njBefore, dataFTE$njAfter)
sort_index_nj <- order(dataFTE$njDiff)
njBeforeSorted <- dataFTE$njBefore[sort_index_nj]
njAfterSorted <- dataFTE$njAfter[sort_index_nj]
njDiffSorted <- dataFTE$njDiff[sort_index_nj] #we also sort the differences to say consistent
plot(x = 1,
     type = "n",
     xlim = c(0, njCounter),
     ylim = c(0, max_nj),
     pch = 16,
     xlab = "Store",
     ylab = "FTE",
     main = "Plot for New Jersey FTE comparison, sorted")
points(njBeforeSorted, pch = 16, lwd = 1, col = "red")
points(njAfterSorted, pch = 16, lwd = 1, col = "blue")
arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter), y1 = njAfterSorted, col = "black", l
```

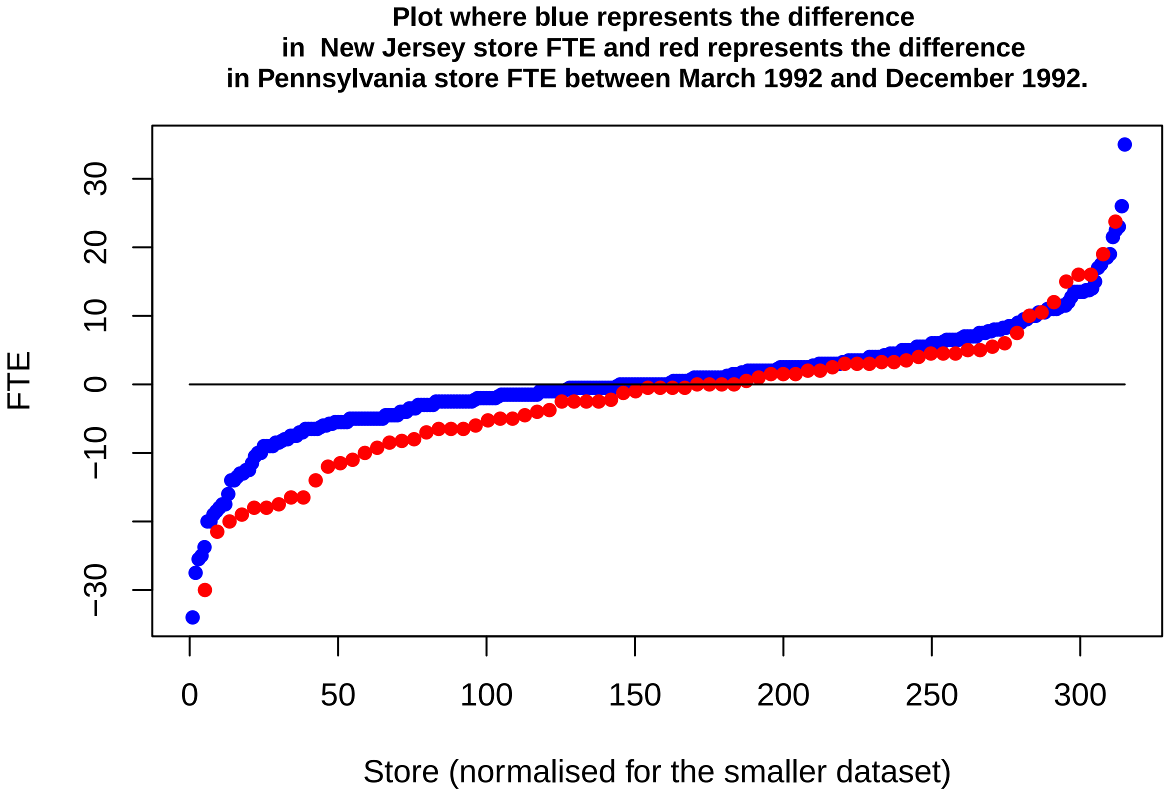
```
## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped
```

```
## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped

## Warning in arrows(x0 = (1:njCounter), y0 = njBeforeSorted, x1 = (1:njCounter),
## : zero-length arrow is of indeterminate angle and so skipped
```



Since no conclusions can be drawn from these points graphs, we zoom in on the pennDiff and njDiff data. Let's plot them in one plot.

```
if(njCounter > pennCounter){
  blue_name <- "New Jersey"
  red_name <- "Pennsylvania"
} else{
  blue_name <- "Pennsylvania"
  red_name <- "New Jersey"
}
plot(x = 1,
     type = "n",
     xlim = c(0, njCounter),
     ylim = c(min(njDiffSorted), max(njDiffSorted)),
     pch = 16,
     xlab = "Store (normalised for the smaller dataset)",
     ylab = "FTE")
title(
  main = paste("Plot where blue represents the difference \nin ", blue_name , "store FTE and red represe
  cex.main = 0.8  # You can adjust the value as needed
)

if(njCounter > pennCounter){
  points(njDiffSorted, pch = 16, lwd = 1, col = "blue")
  points(seq(from = 1, to = njCounter, by = njCounter/pennCounter), pennDiffSorted, pch = 16, lwd = 2, c
} else{
  points(pennDiffSorted, pch = 16, lwd = 1, col = "blue")
  points(seq(from = 1, to = pennCounter, by = pennCounter/njCounter), njDiffSorted, pch = 16, lwd = 2, c
}
segments(x0 = 0, y0 = 0, x1 = max(njCounter, pennCounter), y1 = 0)
```



Final conclusion

...

Sources

For this assignment the handbook of distributions has been used to determine the family of distributions that was used, and for the density function of the Poisson distribution.