**IST 687 PREP EXERCISE 03**

**Student Name: Bhavish Kumar**

**Prep Exercise No: 03**

**Date Due: 11th September 2019**

1. **Getting the Data**
2. Use R code to read directly from a URL on the web. *Hint: use read.csv and url() to read the file from the web*.

urlread <- <https://www2.census.gov/programs-surveys/popest/tables/2010-2011/state/totals/nst-est2011-01.csv>

1. Store the dataset into a new dataframe, called dfStates. Hint: Use stringsAsFactors=FALSE.

dfStates <- read.csv("Nst-est2011.csv", stringsAsFactors = FALSE)

1. The URL is:

<https://www2.census.gov/programs-surveys/popest/tables/2010-2011/state/totals/nst-est2011-01.csv>

1. **Clean Up the Dataframe**
2. Use View( ), head( ), and tail( ) to examine the data frame. Briefly describe what each of the commands show you about a dataframe in general and as it relates to this dataset.

The View Command shows us the entire dataframe and all the rows and columns. The head command returns the headers along with the first 6 rows of the dataframe. The tail command returns the header rows along with the last 6 rows of the dataframe.

View(dfStates)

head(dfStates)

tail(dfStates)

1. Remove unneeded rows by using the minus sign in the row selector of the [ , ] subsetting method. *Hint: Start by removing extra rows that appear* ***at the end*** *of the data set.*

dfStates1 <- dfStates[-60:-66,]

dfStates2 <- dfStates1[-1:-8,]

1. Use the dim() command to make sure there are exactly 51 rows (one per state + the district of Columbia).

dim(dfStates2)

View(dfStates2)

1. Remove unneeded columns by using the minus sign in the column selector of the [ , ] subsetting method.

dfstates3 <- dfStates2[,-6:-10]

1. Use the dim() command to make sure there are precisely five columns.

dim(dfstates3)

View(dfstates3)

1. **Let’s add some meaningful metadata!**
2. Rename the columns with the following names: stateName, Census, Estimated, Pop2010, Pop2011. *Hint: use colnames( )*

newcolnames <- c("stateName", "Census", "Estimated", "Pop2010", "Pop2011")

colnames(dfstates3) <- newcolnames

1. **More cleansing!** 
   1. Use gsub(",", "", vectorName) to remove the commas from each column of numeric data. Place the converted results back into the data frame.

stateNameCleaned <- gsub(",","",dfstates3$stateName)

CensusCleaned <- gsub(",","",dfstates3$Census)

EstimatedCleaned <- gsub(",","",dfstates3$Estimated)

Pop2010Cleaned <- gsub(",","",dfstates3$Pop2010)

Pop2011Cleaned <- gsub(",","",dfstates3$Pop2011)

dfstates4 <- data.frame(stateNameCleaned,CensusCleaned,EstimatedCleaned,Pop2010Cleaned, Pop2011Cleaned)

* 1. Use as.numeric( ) to coerce each numeric data column into numbers. Place the converted results back into the data frame

CensusCleaned <- as.numeric(dfstates4$CensusCleaned)

EstimatedCleaned <- as.numeric(dfstates4$EstimatedCleaned)

Pop2010Cleaned <- as.numeric(dfstates4$Pop2010Cleaned)

Pop2011Cleaned <- as.numeric(dfstates4$Pop2011Cleaned)

dfstates5 <- data.frame(stateNameCleaned,CensusCleaned,EstimatedCleaned,Pop2010Cleaned, Pop2011Cleaned)

* 1. Calculate the mean of the 4 numeric variables and fill in the table below:

## Census

mean(dfstates4$CensusCleaned)

##Estimated

mean(dfstates4$EstimatedCleaned)

## Pop2010

mean(dfstates4$Pop2010Cleaned)

## Pop2011

mean(dfstates4$Pop2011Cleaned)

|  |  |
| --- | --- |
| **Census** | 6053834 |
| **Estimated** | 6053834 |
| **Pop2010** | 6065298 |
| **Pop2011** | 6109645 |

1. **List any additional resources you used here.**
2. **Be sure to save your R file as this will become the starting code for your homework.**

