This README file explains the R-code for Q2.R

# Packages Installed

* + - install.packages("data.table")

Extension of ‘data.frame’: Fast aggregation of large data (e.g. 100GB in RAM), fast ordered joins, fast add/modify/delete of columns by group using no copies at all, list columns, a fast friendly file reader and parallel file writer. Offers a natural and flexible syntax, for faster development.

Required for function: fread()

* + - install.packages("moments")

To include these Functions which calculate: moments, Pearson's kurtosis, Geary's kurtosis and skewness.

* + - install.packages("e1071")

Functions for latent class analysis, short time Fourier transform, fuzzy clustering, support vector machines, shortest path computation, bagged clustering, naive Bayes classifier.

Required for function: kurtosis()

* + - install.packages("plyr")

A set of tools that solves a common set of problems: you need to break a big problem down into manageable pieces, operate on each piece and then put all the pieces back together. For example, you might want to fit a model to each spatial location or time point in your study, summarise data by panels or collapse high-dimensional arrays to simpler summary statistics. The development of 'plyr' has been generously supported by 'Becton Dickinson'.

Required for function: ddply()

# List of Functions used

|  |  |
| --- | --- |
| 1. Setwd | Specify the path to the desired directory |
| 1. Fread | To import data from regular delimited files directly into R, without any detours or nonsense |
| 1. Rbind | Combines vector, matrix or data frame by rows |
| 1. Cbind | Combines vector, matrix or data frame by columns |
| 1. Rm | Remove- used to remove objects |
| 1. head | Obtain the first several rows of a matrix or data frame |
| 1. median | To calculate the median of the value passed |
| 1. cat | Outputs the objects, concatenating the representations |
| 1. levels | Provides access to the levels attribute of a variable |
| 1. summary | Used to produce result summaries of the results of various model fitting functions |
| 1. max | Return the maximum of all the values |
| 1. min | Return the minimum of all the values |
| 1. sum | Return the sum of all the values |
| 1. sprintf | Returns a character vector containing a formatted combination of text and variable values |
| 1. aggregate | Splits the data into subsets, computes summary statistics for each, ad returns the result in a convenient form |
| 1. strsplit | Outputs a list, where each list item corresponds to an element that has been split |
| 1. which | Return the position of the elements in a logical vector which are TRUE |
| 1. subset | Return subsets of vectors, matrices or data frame which meet conditions |
| 1. sd | Computes the standard deviation of the values passed |
| 1. ddply | It split data frame, apply function ten combine results into a data frame |
| 1. sys.time | To get the current date and time |

# Data Structures Used

|  |  |
| --- | --- |
| * List | T/he contents of a list are not restricted to a single mode and can encompass any mixture of data types |
| * matrix | They are not a separate type of object but simply an atomic vector with dimensions added on to it. Matrices have rows and columns. |
| * data frame | A data frame is a very important data type in R. It's pretty much the *de facto* data structure for most tabular data and what we use for statistics |
| * factors | Represents the categorical data |

# Input files

* loadstats3c.csv & loadstats3d.csv

These files contain complete loan data for all loans issued in 2014 and 2015, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information.

The file containing loan data through the "present" contains complete loan data for all loans issued through the previous completed calendar quarter.

* LCdatadictionary.xlsx

Data dictionary includes definitions for all the data attributes included in loan data file.

# Assumptions

* Number of days in every month is considered as 30.
* Assumption in part(f) : rate of return percent = (interest received/original value ) \* 100
* Assumption in part(h): default loan status is assumed as ALL - (fully paid + current + in grace)

# Approach used to solve the statistical problem

Approach Used For Respective Parts of Question:

###### What is the median loan amount?

To find the median of loan amount we first convert the values of column loan\_amnt and apply the median function to the column loan\_amnt and print the value using cat function.

###### Each loan is categorized into a single purpose. What fraction of all loans are for the most common purpose?

###### Firstly, the ‘purpose’ column is converted into factors and then count the occurrences of each level in the factor using summary function. Finally, the level of maximum occurrences is divided by the sum of occurrences of each level to find the fraction. The result is printed using ‘sprintf’.

###### Calculate the average interest rate across loans for each purpose. What is the ratio of minimum average rate to the maximum average rate? (The ratio should be less than 1.)

###### We make a matrix of avg\_int\_rates grouped by purpose and the calculate the ratio of minimum average rate to maximum average using aggregate function.

###### What is the difference in the fraction of the loans with a 36-month term between 2014 and 2015?

###### The column named ‘term’ is stored in a variables ‘data2014 & data2015’ and converted into factor. To find the fraction summary of data2014 is divided by sum of summary of data2014. Similarly, fraction for data2015 is calculated. At last, the difference in the fraction of the loans with a 36-month term between 2014 and 2015 is printed using ‘sprintf’ function.

###### I will consider all loans that are not in the 'Fully Paid', 'Current', 'In Grace Period' statuses to be in default. Calculate the ratio of the time spent paying the loan, defined as the difference between the last payment date and the issue date, divided by the term of loan. What is the standard deviation of this ratio for all the loans in default?

###### A subset is created using following columns: loan\_status, term, issue\_d, last\_pymnt\_d. All the ‘NA’ values in last\_pymnt\_d is handled at last in SD function. Calculate the months spent paying the loan by the borrower by subtracting issue\_date from last\_payment\_date and divided it by the number of days in a month i.e. 30.

###### Ratio of time spent = time spent by user / total term and standard deviation of this ratio is calculated

###### What is the Mean, Median, Mean Absolute Deviation, Variance, IQR, Skewness and Kurtosis for the total rate of return, as figured from the total payments and the loan amount, and the interest rate? Consider only loans that have reached the end of their term. [Summary function NOT to be used here]

###### Here in this part we assumed that rate of return percent = (interest received/original value ) \* 100. Then find the Mean, Median, Mean Absolute Deviation, Variance ,IQR,kurtosis, skewness using their direct R functions. Finally plot skewness and kurtosis:

###### plot(density(rate\_of\_return), xlim=c(min(rate\_of\_return), max(rate\_of\_return)))

###### Let's find a loan purpose that shows up abnormally often in one state. Call A the probability of a loan going to a specific purpose nationwide. Call B the probability of a loan going to a specific purpose for each state. Out of all (state, purpose) pairs with at least 10 loans, what is the highest ratio of B / A (i.e. the most surprising)?

###### We find a matrix with state-purpose-countofloan data and create state purpose pairs using ddply() function. Then find the probability of all car loans in Delhi by creating a matrix for totals of each state. Transpose the matrix so, column names can be accessed and count of state can be obtained. Probability of all state purpose pairs is find i.e. probability B.

###### Favourite outcomes: the total car loans in INDIA , total outcomes: total loans in INDIA

###### Transpose the matrix for accessing and manipulating the data to find nation probability i.e. probability A. Finally, the probability ratio (B\A) is find.

###### We merge A and B with original state purpose pairs for easy user view using cbind(). At last we find the abnormality column based on ration value and store them in a variable abnormal\_ratio.

###### Group all loans by their sub-grade and calculate their average interest rate, average default rate, and percentage of loan status categories.

###### Average Interest rate in a subgrade is calculated mean of int\_rate i.e. , then Default rate in a particular sub-grade i.e. no of default loan status/ total loan status. But default loan status is assumed as ALL - (fully paid + current + in grace). The 2-level grouping of loan status categories is done and Displays all subgrade-loan\_status pairs. The Percentage tells the amount of the contribution each status makes to that subgrade.

# Project Challenges

* How to read csv files??

The biggest problem we encountered is to read these big files. There are header and footer in the .csv files which cause all sorts of issues in reading. when we read the files, and store the data in a variable then we try to print the variable and it returns all null values.

Finally, the issues are solved by skipping the unused rows using function fread() and loading the required columns only.

* How to reduce the time taken to read the files?

The code optimisation is an important aspect for us. So, the first thing we all should consider is the amount of execution time of a code.

When we used read.csv function, it takes 23 sec and 45 seconds time to read the file.

Which was then optimised by fread to 5 sec and 6 sec.

Which was then further optimised by reading only the required columns.

* How to handle the blank strings, missing data values?

Null strings may create problems in reading files and further calculations.

The blank strings are converted to NA which saves us from additional overhead later.

fread(“input file”, na.strings=c("","NA","n/a"))

Specifically,

In the as.Date() function, null strings were not processed as expected.

So we converted values to NA, which are implicitly handled in the as.Date function, and return NA.

# Running the code

Set the working directory

Place the csv files in the working directory

Install the packages, if missing by uncommenting the lines of code given to do so.