DS670 Capstone : Big Data & Business Analytics
Lab 6 : Handwritten Lines of Code.

Project - Code

```
# To plot small cap vs large cap stocks performance.
%r
1. small_cap_stock <- read_csv("/home/madhumita/Desktop/sem_3/
                                DS670/Project/iwm.csv",
                               header = TRUE)
2. large_cap_stock <- read_csv("/home/madhumita/Desktop/sem_3/
                                DS670/Project/spy.csv",
                               header = TRUE)
   View(small_cap_stock)
   View(large_cap_stock)
3. large_cap_stock_2000 <- subset(large_cap_stock,
                                  large_cap_stock >= "2000-01-01")
   View(large_cap_stock_2000)

4. plot(small_cap_stock$Date, small_cap_stock$'Adj Close',
        ylim = range(c(small_cap_stock$'Adj Close',
                       large_cap_stock_2000$'Adj Close')), type="l",
        xlab = "Date", ylab = "Closing Price", col = "red",
        main = 'Small (red) vs Large (Blue) Cap Performance",
        sub = "Jan 2000 - Feb 2017")
5. lines(large_cap_stock_2000$Date, large_cap_stock_2000$'Adj Close',
         col = "blue").
```

```
# To read the quarterly data file & calculate
  returns
%r
arqdata ← read.csv (file = "/home/madhumita/Desktop/sem.3
                    DS670/Project/ARQ.Data.csv",
                    header = TRUE)
save (arqdata, file = "arq.data")
load ("arq.rdata")
nrow (arqdata)
# Consider the column price from file - Calculate Returns
arqdata_na ← subset (arqdata, arqdata$price != "NA")
return ← vector();
for (i in 2 : length (arqdata_na [i-1, 1])) {
    if (identical (arqdata_na [i,1], arqdata_na [i-1,1]))
    {
        return [i] = arqdata_na [i,72] / arqdata_na [i-1, 72];
    }
    else
    {
        return [i] = 0;
    }
}
return [1] = 0;

# Adding "return" to 'arqdata' dataset
%r
arqdata_returns ← cbind (arqdata_na, return)
```

```
# Considering Indicators listed.
%or
# Calculating ratios by choice 1. sgnamargin 2. ebitmargin.
19 sgnamargin = arqdata_returns$ sgna / arqdata_returns$ revenue.
20 ebitmargin = arqdata_returns$ ebit / arqdata_returns$ revenue.

# adding ratios by choice to dataset - arqdata_return
21. arqdata_returns_ratios <- cbind (arqdata_returns, sgnamargin,
                                     ebitmargin).

# Consider the 20 Indicator chosen
# Assign them to a new dataset.

22. arq_data_factors <- arqdata_returns_ratios [ c( 1, 3, 77, 16, 42,
                        84, 24, 43, 95, 96, 45, 91, 61, 29, 30, 65, 7, 9,
                        50, 17, 93, 13, 94)]
23 View (arq_data_factors)

# Normalize all the Indicators for all the dates -
# 20 quarters (5 years of data).
%or
24 caldate = unique (arq_data_factors$ calendardate)
25 length (caldate)
26 prj2_arq_date <- vector();
27 prj2_arq_hm <- vector();
28 prj2_date_replace <- vector();
```

```
29. factors1 <- NULL
30. factors2 <- NULL
31. factors3 <- NULL

    # Loop for dates - each date we get a dataset
    # "prj2_arg - 'date'"
32. for (i in 1:length(caldate)) {
33.     print (paste0("calendar date: ", caldate[i])
34.     factors1 <- subset(arg_data_factors, calendardate ==
                                                caldate[i])

        # Calculate log(returns) #
35.     factors1 <- subset (factors1, factors1$return != 0)
36.     return_log <- log (factors1$return)
37.     factors1 <- cbind (factors1, return_log)

        # Remove all NA's before normaliging the dataset.
38.     factors1 <- na.omit (factors1).

        # Normaliging indicators.
39.     revenue_nor <- (factors1[,3] - mean(factors1[,3]))/sd(factors1[,3])
40.     cor_nor <- (factors1[,4] - mean(factors1[,4]))/sd(factors1[,4])
41.     gp_nor <- (factors1[,5] - mean(factors1[,5]))/sd(factors1[,5])
42.     sgna_nor <- (factors1[,6] - mean(factors1[,6]))/sd(factors1[,6])
43.     ebit_nor <- (factors1[,7] - mean(factors1[,7]))/sd(factors1[,7])
44.     gm_nor <- (factors1[,8] - mean(factors1[,8]))/sd(factors1[,8])
45.     sgna_rg_nor <- (factors1[,9] - mean(factors1[,9]))/sd(factors1[,9])
```

```
46. ebit_mg_nor <- (factors1[,10] - mean(factors1[,10]))/sd(factors1[,10])
47. interp_nor <- (factors1[,11] - mean(factors1[,11]))/sd(factors1[,11])
48. taxexp_nor <- (factors1[,12] - mean(factors1[,12]))/sd(factors1[,12])
49. netinc_nor <- (factors1[,13] - mean(factors1[,13]))/sd(factors1[,13])
50. ebt_nor <- (factors1[,14] - mean(factors1[,14]))/sd(factors1[,14])
51. eps_nor <- (factors1[,15] - mean(factors1[,15]))/sd(factors1[,15])
52. netmar_nor <- (factors1[,16] - mean(factors1[,16]))/sd(factors1[,16])
53. assets_nor <- (factors1[,17] - mean(factors1[,17]))/sd(factors1[,17])
54. assetsc_nor <- (factors1[,18] - mean(factors1[,18]))/sd(factors1[,18])
55. liabc_nor <- (factors1[,19] - mean(factors1[,19]))/sd(factors1[,19])
56. cur_ratio_nor <- (factors1[,20] - mean(factors1[,20]))/sd(factors1[,20])
57. wc_nor <- (factors1[,21] - mean(factors1[,21]))/sd(factors1[,21])
58. capex_nor <- (factors1[,22] - mean(factors1[,22]))/sd(factors1[,22])

# Appending normalized columns to factors2.
59. factors1 <- cbind(factors1, revenue_nor, cor_nor, gp_nor, sgna_nor,
                      ebit_nor, gm_nor, sgna_mg_nor, ebit_mg_nor,
                      interp_nor, taxexp_nor, netinc_nor, ebt_nor, eps_nor,
                      netmar_nor, assets_nor, assetsc_nor, liabc_nor,
                      cur_ratio_nor, wc_nor, capex_nor)
60. factors2 <- factors1[c(1,2,25:44,23,24)]
61. factors3 <- factors1[c(25:44,24)]

# Creating datasets based on dates for regression
62. prj2_date_replace[i] <- gsub("-","_", caldate[i])
63. prj2_data_date[i] <- paste("prj2_arq_", prj2_date_replace[i],
                               sep="")
```

```
64.   assign(prj2_arq_date[i], factors2)

      # Creating datasets base on dates for neural networks
65.   prj2_date_replace[i] <- gsub("-", "_", caldate[i])
66.   prj2_arq_nn[i] <- paste("prj2_arqnn_",
                              prj2_date_replace[i], sep="")
67.   assign(prj2_arq_nn[i], factors3)

68.   factors1 <- NULL
69.   factors2 <- NULL
70.   factors3 <- NULL
      }
      # Dataset names
71.   prj2_arq_date
72.   prj2_arq_nn
73.   View(get(prj2_arq_nn[1]))   # first date - dataset

      # Installing "neuralnet" & "MASS" libraries
      % or
74.   library("MASS")
75.   library("neuralnet")
      # neural net does not take strings as input.
76.   names_date <- names(get(prj2_arq_nn[1]))
77.   names_date
```

Madhumita D
2/27/2017
Pg-7.

```
# log(returns) as 'y' we need to get formula.
78 names_date %in% "return_log"
79 !names_date %in% "return_log"
80 paste(names_date[!names_date %in% "return_log"])
81 paste(names_date[!names_date %in% "return_log"], collapse = "+"))
82 paste("return_log ~", names_date[!names_date %in% "return_log"],
                                        collapse = "+")
83 formula1 <- as.formula(paste("return_log ~", names_date[
                !names_date %in% "return_log"], collapse = "+"))
84 formula1.

# Assigning datasets - from date 15 to date 20.
85 uvw1 <- get(prj2_arq_nn[15])
86 uvw2 <- get(prj2_arq_nn[16])
87 uvw3 <- get(prj2_arq_nn[17])
88 uvw4 <- get(prj2_arq_nn[18])
89 uvw5 <- get(prj2_arq_nn[19])
90 uvw6 <- get(prj2_arq_nn[20])

### NEURAL NETWORKS AND PREDICTIONS
# Sept 2014 model & Dec 2014 predict returns
91 nn_2014_09_30 <- neuralnet(formula1, data = uvw1, hidden = c(6,7),
                                linear.output = T, stepmax = 1e6)
92 pred_2014_12_31 <- compute(nn_2014_09_30, uvw2[,1:20]
93 exp_ret_2014_12_31 <- (pred_2014_12_31$net.result) * sd(uvw1$return_log)
                        + mean(uvw1$return_log)
94 uvw2 <- cbind(uvw2, exp_ret_2014_12_31)
```

```
# Dec 2014 model to predict March 2015 returns
nn_2014_12_31 <- neuralnet(formula1, data = uvw2, hidden(7,7),
                           linear.output = T, stepmax = 1e6)
pred_2015_03_31 <- compute(nn_2014_12_31, uvw3[,1:20])
exp_ret_2015_03_31 <- (pred_2015_03_31$net.result)*sd(uvw2$return_log)
                      + mean(uvw2$return_log)
uvw3 <- cbind(uvw3, exp_ret_2015_03_31)

# March 2015 model to predict June 2015 returns
nn_2015_03_31 <- neuralnet(formula1, data = uvw3, hidden = c(8,7),
                           linear.output = T, stepmax = 1e6)
pred_2015_06_30 <- compute(nn_2015_03_31, uvw4[,1:20])
exp_ret_2015_06_30 <- (pred_2015_03_31$net.result)*sd(uvw3$return_log)
                      + mean(uvw3$return_log)
uvw4 <- cbind(uvw4, exp_ret_2015_06_30)

# June 2015 model to predict Sep 2015 returns
nn_2015_06_30 <- neuralnet(formula1, data = uvw4, hidden = c(7,7),
                           linear.output = T, stepmax = 1e6)
pred_2015_09_30 <- compute(nn_2015_06_30, uvw5[,1:20])
exp_ret_2015_09_03 <- (pred_2015_09_30$net.result)*sd(uvw4$return_log)
                      + mean(uvw4$return_log)
uvw5 <- cbind(uvw5, exp_ret_2015_09_30)
```

```
# Sep 2015 model & predict Dec 2015 returns.
nn_2015_09_30 <- neuralnet(formula1, data=uvw5, hidden=c(7,7),
                           linear.output=T, stepmax=1e2)
pred_2015_12_31 <- compute(nn_2015_09_30, uvw6[,1:20])
exp_ret_2015_12_31 <- ((pred_2015_12_31$net.result)*sd(uvw5$return_log))
                      +mean(uvw5$return_log))
uvw6 <- cbind(uvw6, exp_ret_2015_12_31)

# Plot the neural networks for dates 15 to 20.
plot(nn_2014_09_30)
plot(nn_2015_03_31)
plot(nn_2015_06_30)
plot(nn_2015_09_30)
plot(nn_2015_12_31)

# Predicted returns for dates t16-t20 by using NN
# Dataset assigned below.
arq_2014_12_31 <- uvw2[,c(1:22)]
arq_2015_03_31 <- uvw3[,c(1:22)]
arq_2015_06_30 <- uvw4[,c(1:22)]
arq_2015_09_30 <- uvw5[,c(1:22)]
arq_2015_12_31 <- uvw6[,c(1:22)]
```