gaussian_fraud_model

June 17, 2018

1 Multivariate Gaussian Model

1.0.1 By Kumar Rahul

The analysis is on company financial manipulations and devise algorithm to identify a manipulater from a non manipulater based on the financial ratios reported by the companies. There are a total of 1239 observations in the data set. Out of these 1239 observations, there are 1200 non manipulaters and 39 manipulaters

---

In [1]:
```
library(caret)            #for split and model accuracy
setwd("/Users/Rahul/Documents/Rahul Office/IIMB/Work @ IIMB/Training Material/Concepts/
```

```
Loading required package: lattice
Loading required package: ggplot2
```

1.1 Preparing data

Read data from a specified location

In [4]:
```
raw_df <- read.csv("/Users/Rahul/Documents/Rahul Office/IIMB/Work @ IIMB/Training Mate

filter_df <- raw_df[,-c(1)]
head(filter_df)
```

DSRI	GMI	AQI	SGI	DEPI	SGAI	ACCR	LEVI	Status
1.624742	1.128927	7.1850534	0.3662114	1.381519	1.62414487	-0.16680870	1.1610817	Yes
1.000000	1.606492	1.0049879	13.0814332	0.400000	5.19820717	0.06047516	0.9867325	Yes
1.000000	1.015607	1.2413895	1.4750183	1.169353	0.64767093	0.03673163	1.2643050	Yes
1.486239	1.000000	0.4655348	0.6728395	2.000000	0.09288991	0.27343412	0.6809750	Yes
1.000000	1.369038	0.6371120	0.8613464	1.454676	1.74145963	0.12304770	0.9390472	Yes
0.905532	1.360915	0.7839949	1.7933237	1.278244	0.50526004	0.05464238	1.5431371	Yes

In [5]:
```
set.seed(1224)
trainIndex <- createDataPartition(filter_df$Status, p = 0.75, list = FALSE)
train_df <- filter_df[trainIndex,]
test_df <- filter_df[-trainIndex,]
```

```
In [6]: subset_manipulator_df <- subset(train_df,
                                train_df$Status == "Yes")
        new_train_df <- subset(train_df,train_df$Status=="No")
        new_test_df <- rbind(test_df,subset_manipulator_df)
```

Prepare and run numerical summaries

```
In [7]: summary(new_train_df) #summary of the data
        new_train_df <- na.omit(new_train_df) # listwise deletion of missing
        new_test_df <- na.omit(new_test_df) # listwise deletion of missing
```

```
      DSRI              GMI                AQI                SGI
 Min.   :0.0000   Min.   :-20.8118   Min.   :-21.7338   Min.   :0.1541
 1st Qu.:0.8859   1st Qu.:  0.9307   1st Qu.:  0.7664   1st Qu.:0.9703
 Median :1.0278   Median :  1.0000   Median :  1.0111   Median :1.0856
 Mean   :1.0999   Mean   :  0.9631   Mean   :  0.9735   Mean   :1.0927
 3rd Qu.:1.1870   3rd Qu.:  1.0604   3rd Qu.:  1.2293   3rd Qu.:1.1972
 Max.   :7.1177   Max.   :  7.1386   Max.   : 12.8854   Max.   :3.3340
      DEPI              SGAI              ACCR               LEVI
 Min.   :0.06958   Min.   :0.0000   Min.   :-3.14350   Min.   :0.03877
 1st Qu.:0.93672   1st Qu.:0.8969   1st Qu.:-0.07649   1st Qu.:0.92479
 Median :1.00293   Median :1.0000   Median :-0.03004   Median :1.01695
 Mean   :1.03525   Mean   :1.0453   Mean   :-0.03434   Mean   :1.04325
 3rd Qu.:1.08013   3rd Qu.:1.1142   3rd Qu.: 0.02004   3rd Qu.:1.11650
 Max.   :4.79883   Max.   :6.9075   Max.   : 0.95989   Max.   :6.25043
 Status
 No :900
 Yes:  0
```

Train and test dataset with needed variables

```
In [8]: model_train_df <- as.data.frame(new_train_df[,c(#"DSRI",
                                                        #"GMI",
                                                        "AQI",
                                                        #"SGI",
                                                        "DEPI",
                                                        "SGAI",
                                                        "ACCR",
                                                        "LEVI",
                                                        "Status"
        )])

        model_test_df <- as.data.frame(new_test_df[,c(#"DSRI",
```

```
                                                    #"GMI",
                                                    "AQI",
                                                    #"SGI",
                                                    "DEPI",
                                                    "SGAI",
                                                    "ACCR",
                                                    "LEVI",
                                                    "Status"
        )])
```

1.2 Gaussian Model

```
In [9]: mean_model_train_df <- apply(model_train_df[,1:5], 2, mean)
        sd_model_train_df <- apply(model_train_df[,1:5], 2, sd)

In [10]: gaussian_model <- function(x,m,s){
           constant <- 1 / (s * sqrt( 2 * pi))
           value <- exp((-1*(x - m) ^ 2) / (2 * (s ^ 2)))
           constant*value
         }

In [11]: prob <- apply(model_test_df[1:5], 1, function(x,y,z) gaussian_model(x,mean_model_trai
         prob <- t(prob)
         colnames(prob) <- paste("P(", colnames(prob), ")", sep = "")

         gaussian_model_test_df <- cbind(model_test_df,prob)

In [12]: gaussian_model_test_df$JointProb <-
                             #gaussian_model_test_df$`P(DSRI)`*
                             #gaussian_model_test_df$`P(GMI)`*
                             gaussian_model_test_df$`P(AQI)`*
                             #gaussian_model_test_df$`P(SGI)`*
                             gaussian_model_test_df$`P(DEPI)`*
                             gaussian_model_test_df$`P(SGAI)`*
                             gaussian_model_test_df$`P(ACCR)`*
                             gaussian_model_test_df$`P(LEVI)`
         gaussian_model_test_df$Actual.Status <- gaussian_model_test_df$Status

In [13]: #creating empty vectors to store the results.
         msclaf_cost <- c()
         youden_index <- c()
         cutoff <- c()
         P11 <- c() #correct classification of positive as positive
         P00 <- c() #correct classification of negative as negative
         P10 <- c() #misclassification of positive class to negative class
         P01 <- c() #misclassification of negative class to positive class

In [14]: n <- length(gaussian_model_test_df$Actual.Status)
         costs = matrix(c(0,2,1, 0), ncol = 2)
```

```
colnames(costs) = rownames(costs) = c("Yes", "No")
as.table(costs)
```

```
    Yes No
Yes   0  1
No    2  0
```

The misclassification cost table is:

```
In [16]: for (i in seq(0.05, 1, .05)) {
           predicted_status_df = rep("No", n)
           predicted_status_df[gaussian_model_test_df$JointProb < i] = "Yes"
           tbl <- table(gaussian_model_test_df$Actual.Status, predicted_status_df)
           if ( i <= 1) {
             #Classifying Not Joined as Joined
             P10[20*i] <- tbl[2]/(tbl[2] + tbl[4])

             P11[20*i] <- tbl[4]/(tbl[2] + tbl[4])

             #Classifying Joined as Not Joined
             P01[20*i] <- tbl[3]/(tbl[1] + tbl[3])

             P00[20*i] <- tbl[1]/(tbl[1] + tbl[3])

             cutoff[20*i] <- i
             msclaf_cost[20*i] <- P10[20*i]*costs[2] + P01[20*i]*costs[3]
             youden_index[20*i] <- P11[20*i] + P00[20*i] - 1
           }
         }
         cost_table_df <- cbind(cutoff,P10,P01,msclaf_cost, P11, P00, youden_index)
         cost_table_df
```

cutoff	P10	P01	msclaf_cost	P11	P00	youden_index
0.05	0.43589744	0.1233333	0.9951282	0.5641026	0.87666667	0.440769231
0.10	0.35897436	0.1700000	0.8879487	0.6410256	0.83000000	0.471025641
0.15	0.30769231	0.2066667	0.8220513	0.6923077	0.79333333	0.485641026
0.20	0.30769231	0.2633333	0.8787179	0.6923077	0.73666667	0.428974359
0.25	0.25641026	0.2866667	0.7994872	0.7435897	0.71333333	0.456923077
0.30	0.23076923	0.3500000	0.8115385	0.7692308	0.65000000	0.419230769
0.35	0.20512821	0.4066667	0.8169231	0.7948718	0.59333333	0.388205128
0.40	0.15384615	0.4633333	0.7710256	0.8461538	0.53666667	0.382820513
0.45	0.12820513	0.5166667	0.7730769	0.8717949	0.48333333	0.355128205
0.50	0.10256410	0.5700000	0.7751282	0.8974359	0.43000000	0.327435897
0.55	0.07692308	0.6433333	0.7971795	0.9230769	0.35666667	0.279743590
0.60	0.07692308	0.7366667	0.8905128	0.9230769	0.26333333	0.186410256
0.65	0.07692308	0.8433333	0.9971795	0.9230769	0.15666667	0.079743590
0.70	0.05128205	0.9433333	1.0458974	0.9487179	0.05666667	0.005384615
0.75	NA	NA	NA	NA	NA	NA
0.80	NA	NA	NA	NA	NA	NA
0.85	NA	NA	NA	NA	NA	NA
0.90	NA	NA	NA	NA	NA	NA
0.95	NA	NA	NA	NA	NA	NA
1.00	NA	NA	NA	NA	NA	NA

1.3 Model Statistics

In [17]:
```
#variable with all the values as No
n <- length(gaussian_model_test_df$Actual.Status)

predicted_status_df = rep("No", n)

# Changing the Status to Manipulator = Yes if probability is less than threshold
predicted_status_df[gaussian_model_test_df$JointProb < 0.25] = "Yes"

#add the model_precition in the data
gaussian_model_test_df$predicted_status_df <- predicted_status_df

###Create the confusionmatrix###
addmargins(table(gaussian_model_test_df$Actual.Status, gaussian_model_test_df$predicte
mean(gaussian_model_test_df$predicted_status_df == gaussian_model_test_df$Actual.Statu
```

	No	Yes	Sum
No	214	86	300
Yes	10	29	39
Sum	224	115	339

0.716814159292035

End of document