Birla Institute of Technology and Science, Pilani
SS ZG565: Machine Learning
1st Semester 2019-20
Labsheet-02 : Naive Bayes Classifier, E M Algorithm

1 Naive Bayes Classifier :

Naive Bayes classifier is a generative classifier with assumption that features are statistically independent. For n features $x_1, x_2, ..., x_n$ and a class C_k the model can be expressed as:

$$P(C_k|x_1, x_2, ..., x_n) \alpha P(C_k) \prod_{i=1}^{n} P(x_i|C_k)$$

The naive Bayes functionality is supported in the e1071 package. Install and load the package by issuing install.packages(“e1071”) and library(e1071). This will make the naiveBayes() function available for us to use. We would use the Analysis_Grade data set which has been shared with you all.

```
#Program Name:NaiveBayes.R

#To clear Global environment variables

rm(list=ls(all=T))


#Install the packages necessary

install.packages('e1071')
install.packages("dplyr")
install.packages("psych")
install.packages('ggplot')


#load the packages needed

library('e1071')
library("dplyr")
library("psych")
library("ggplot2")
library(readxl)
```

```r
#load the data for which Naive Bayes model has to be applied

Analysis_Grade <- read_excel("/home/sachin/Desktop/Analysis_Grade.xlsx")


#Analyse the structure of the  data loaded

str(Analysis_Grade)
head(Analysis_Grade)
summary(Analysis_Grade)


#As, ID(1st column) is not needed to analyse data, we remove that feature from the
dataset

Analysis_2 <- subset(Analysis_Grade, select = Year:Grade)
str(Analysis_2)
View("Analysis_2")


#Remove the feature that doesn't have importance in the  classification. Here, Year and
gender doesn't affect the classification and hence, we remove these two features

Analysis_2$Year <- NULL
Analysis_2$`M/F` <- NULL


#Convert the Grade variable to factor variable, for better classification

Analysis_2$Grade <- as.factor(Analysis_2$Grade)


#Find the correlation among the independent variables. For NaiveBayes classification,
the independent variable should not be highly correlated. Here, Analysis_s[-6] means,
we are not taking the 6th column, i.e Grade factor for finding the correlation among the
variables, as Grade is our final classification.

pairs.panels(Analysis_2[-6])
```

#Generate random training and testing data from the given dataset. Here, 80% of data is taken for training and 20% of data for testing.

```
set.seed(1234)
ind <- sample(2, nrow(Analysis_2), replace =T , prob = c(0.8, 0.2))
train <- Analysis_2[ind == 1,]  #Assigning 80% of data for training
test <- Analysis_2[ind ==2,]  #Assigning 20% of data for testing
```

#Apply the NaiveBayes model for the training data.

```
NB_model <- naiveBayes(Grade ~ ., data = train )
```

#Get the predictive statistics after applying NaiveBayes model for classification. It gives you the predicted percentage of all the classes.

```
NB_model
```

#Predict the class for each training data and calculate the misclassification for the model applied.

```
train_p <- predict(NB_model, train , type ='class')
head(cbind(train_p,train))
train_p1 <- predict(NB_model,train)
train_tab <- table( train_p1,train$Grade)
mis_classification <- 1 - sum(diag(train_tab)) / sum(train_tab)
```

#Now, Predict the class for each testing data and calculate the misclassification for the model applied.

```
test_p <- predict(NB_model, test , type ='class')
head(cbind(test_p,test))
test_p1 <- predict(NB_model,test)
(test_tab <- table( test_p1,test$Grade))
```

```
1 - sum(diag(test_tab)) / sum(test_tab)
```

Questions:

1. Why did we change the Grade variable into factor variable? Will it work if it is not converted to a factor variable ?
2. Find out about Laplace smoothing and its need in Naive Bayes classification. How can you control it using the naiveBayes() function?
3. Apply the Naive Bayes model for any classification dataset and analyse the results(You will get the dataset from kaggle website).
4. Apply some other classification model(knn, Decision Tree, SVM etc) on “Analysis_Grade” dataset and compare the results obtained with the NaiveBayes classification Model.

2. EM Algorithm

In many practical learning settings, only a subset of the relevant instance features might be observable.If some variable is sometimes observed and sometimes not, then we can use the cases for which it has been observed to learn to predict its values when it is not. EM algorithm (Dempster et al. 1977), a widely used approach to learning in the presence of unobserved variables. The EM algorithm can be used even for variables whose value is never directly observed, provided the general form of the probability distribution governing these variables is known. The EM algorithm is also the basis for many unsupervised clustering algorithms (e.g.,Cheeseman et al. 1988), and it is the basis for the widely used Baum-Welch forward-backward algorithm for learning Partially Observable Markov Models (Rabiner 1989).
The library used for the EM algorithm is “**mclust**”.
mclust is a contributed R package for model-based clustering, classification, and density estimation based on finite normal mixture modelling. It provides functions for parameter estimation via the EM algorithm for normal mixture models with a variety of covariance structures, and functions for simulation from these models.

mclust automatically chooses the best number of components and the co variance parameterization by **BIC(Bayesian Information Criterion).**
BIC is the is a criterion for model selection among a finite set of models. When fitting models, it is possible to increase the likelihood by adding parameters, but doing so may

result in overfitting.BIC attempt to resolve this problem by introducing a penalty term for the number of parameters in the model.

```
#ProgramName : EMAlgorithm.R

# “mclust” package is required for EM algorithm bases models. Install and load it

install.packages('mclust')
library(mclust)


# Read the dataset

data1<- read.csv(file.choose(), header=T)
data(diabetes)
class <- diabetes$class
table(class)
head(diabetes)


#exclude the first feature from the dataset

X <- diabetes[,-1]
head(X)


#visualise the scatter plot among the features of the given dataset

clPairs(X, class)



#BIC for parameterized Gaussian mixture models fitted by EM algorithm initialized by
model-based hierarchical clustering.

clust <- mclustBIC(X)
plot(clust)
summary(clust)
```

```
#Gaussian finite mixture model fitted by EM algorithm

mod1 <- Mclust(X, x = clust)
summary(mod1, parameters = TRUE)
plot(mod1, what = "classification")
table(class, mod1$classification)
plot(mod1, what = "uncertainty")
```