Exerise14_AyachitMadhukar

R Exercise 14

```
setwd("~/MadR/Workspaces/dsc520")
```

Loading data from file

```
data=read.csv("data/binary-classifier-data.csv")
```

Observations at first glance

```
#Normalization
summary(data)
```

```
##      label             x                 y          
##  Min.   :0.000   Min.   : -5.20   Min.   : -4.019  
##  1st Qu.:0.000   1st Qu.: 19.77   1st Qu.: 21.207  
##  Median :0.000   Median : 41.76   Median : 44.632  
##  Mean   :0.488   Mean   : 45.07   Mean   : 45.011  
##  3rd Qu.:1.000   3rd Qu.: 66.39   3rd Qu.: 68.698  
##  Max.   :1.000   Max.   :104.58   Max.   :106.896
```

```
str(data)
```

```
## 'data.frame':    1498 obs. of  3 variables:
##  $ label: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ x    : num  70.9 75 73.8 66.4 69.1 ...
##  $ y    : num  83.2 87.9 92.2 81.1 84.5 ...
```

```
head(data)
```

```
##   label        x        y
## 1     0 70.88469 83.17702
## 2     0 74.97176 87.92922
## 3     0 73.78333 92.20325
## 4     0 66.40747 81.10617
## 5     0 69.07399 84.53739
## 6     0 72.23616 86.38403
```

Randomization

```
set.seed(9850)
gp<-runif(nrow(data))
data<-data[order(gp),]
head(data)
```

```
##     label         x        y
## 216     0 12.006713 58.20435
## 405     0 88.024357 13.26384
## 316     0  7.993121 54.15258
## 804     1 16.669075 73.98231
## 103     0 40.565872 74.84798
## 20      0 69.521713 89.94501
```

Normalization

```
normalize<-function(x){
  return (
            (x - min(x))/max(x)-min(x)
         )
}
data.n<-as.data.frame(lapply(data[,c(2:3)], normalize))

str(data.n)
```

```
## ’data.frame’:    1498 obs. of  2 variables:
##  $ x: num  5.37 6.09 5.33 5.41 5.64 ...
##  $ y: num  4.6 4.18 4.56 4.75 4.76 ...
```

```
summary(data.n)
```

```
##        x               y
##  Min.   :5.200   Min.   :4.019
##  1st Qu.:5.439   1st Qu.:4.255
##  Median :5.650   Median :4.475
##  Mean   :5.681   Mean   :4.478
##  3rd Qu.:5.885   3rd Qu.:4.700
##  Max.   :6.250   Max.   :5.057
```

Preparing training/testing models

```
r<-round(0.8*nrow(data.n))
l<-nrow(data.n)

data.train<-data.n[1:r,]
data.test<-data.n[r:l,]
data.train.target<-data[1:r,1]
data.test.target<-data[r:l,1]
```

Logistic regression

```
library(caTools)

split<- sample.split(data,SplitRatio=.8)

train<-subset(data,split=="TRUE")
test<-subset(data,split=="FALSE")

GLM.1 <- glm(train$label ~ ., family=binomial(), data=train)
summary(GLM.1)
```

```
##
## Call:
## glm(formula = train$label ~ ., family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3655  -1.1433  -0.9494   1.1749   1.4434
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.407395   0.142478   2.859  0.00424 **
## x           -0.001240   0.002250  -0.551  0.58164
## y           -0.009435   0.002319  -4.069 4.72e-05 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1382.4  on 997  degrees of freedom
## Residual deviance: 1363.2  on 995  degrees of freedom
## AIC: 1369.2
##
## Number of Fisher Scoring iterations: 4
```

```
res<-predict(GLM.1,test,type="response")
res<-predict(GLM.1,train,type="response")

conffmatrix<- table(actual_value=train$label,Predicted_value=res>0.5)

conffmatrix
```

```
##             Predicted_value
## actual_value FALSE TRUE
##            0   296  220
##            1   242  240
```

```
acc<-(conffmatrix[1,1] + conffmatrix[2,2])/(conffmatrix[1,1] + conffmatrix[2,2] + conffmatrix[2,1]+ con
```

KNN Classifire

Determining k value

```
 k_value<-round(sqrt(nrow(data)))
```

Predicting knn and accuracy

```
library("class")

knn.39<- knn(train = data.train, test=data.test,cl=data.train.target, k=k_value)
ACC.39<-100 * sum( data.test.target == knn.39) / NROW(data.test.target)

table(data.test.target,knn.39)
```

```
##                  knn.39
## data.test.target   0   1
##                0 156   2
##                1   1 142
```

```
ACC.39
```

```
## [1] 99.00332
```

```
knn.39<- knn(train = data.train, test=data.test,cl=data.train.target, k=k_value)
```

Conclusion

a. What is the accuracy of the logistic regression classifier?

> *logistic regression shows close to 99.0033223 accuracy*

b. How does the accuracy of the logistic regression classifier compare to the nearest neighbors algorithm?

> *logistic regression is way off from KNN whch is nearly 53.7074148% vs KNN being 99.0033223% accurate*

c. Why is the accuracy of the logistic regression classifier different from that of the nearest neighbors?