Practical 12

- Shivam Tawari A-58

Aim : Write a program in R for implementing classification using K-Nearest Neighbour.

Theory:

Classification:
Classification is the process of predicting a categorical lable of a data object based on its feature and properties.

K- Nearest Neighbor (KNN):

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. KNN algorithm assumes the similarity between the new case / data and available cases and put the new case into the category that to most similar to the available categories.

KNN stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using KNN.

KNN can be used for Classification as well as regression. But mostly it is used for Classification problems. It is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm, because it does not learn from the training set immediately instead it stores the dataset and at the time of Classification, it performs an action on the dataset.

How does KNN works?

Step 1: Select the number K of the neighbors.

Step 2: Calculate the Euclidean distance of K numbers of neighbors.

Step 3: Take the K nearest neighbors, as per the Euclidean distance.

Step 4: Among these K neighbors, count the number of the data points in each category.

Step 5: Assign the new data points to that category for which the number of the neighbor is maximum.

K Value:

→ There is no particular way to determine the best value for 'k', so we need to try some values to find the best out of them. The most preferred value for k is 5.

→ A very low value for k such as 1 or 2, can be noisy and lead to the effects of outliers in the model.

→ Large values of k are good, but it may find some difficulties.

Advantages of KNN:

1. It is simple to implement
2. It is robust to the noisy training data.
3. It can be more effective if the training data is large.

Code :

```
df <- data (iris)
head (iris)
ran <- sample (1 : nrow (iris), 0.9 * nrow
                                      (iris))
nor <- function (x) { (x - min (x))/ (max (x)
                                  - min (x)) }
iris_norm <- as.data.frame (lapply (iris
                     [, c(1, 2, 3, 4)], nor))
summary (iris_norm)
iris_train <- iris_norm [ran, ]
iris_test <- iris_norm [-ran, ]
iris_target_category <- iris [ran, 5]
iris_test_category <- iris [-ran, 5]
library (class)
pr <- knn (iris_train, iris_test, cl = iris_
              target_category, k = 13 )
tab <- table (pr, iris_test_category )
print (tab)
accuracy <- function (x) { sum (diag (x) /
            (sum ( rowSums (x)))) * 100 }
accuracy (tab)
```

Conclusion: Hence, we successfully implemented a program in R to classify using k-Nearest Neighbor.

Code:

```r
# Shivam Tawari A-58
df <- data(iris)
head(iris)
ran <- sample(1:nrow(iris), 0.9 * nrow(iris))
nor <-function(x) { (x -min(x))/(max(x)-min(x))   }
iris_norm <- as.data.frame(lapply(iris[,c(1,2,3,4)], nor))
summary(iris_norm)
iris_train <- iris_norm[ran,]
iris_test <- iris_norm[-ran,]
iris_target_category <- iris[ran,5]
iris_test_category <- iris[-ran,5]
library(class)
pr <- knn(iris_train,iris_test,cl=iris_target_category,k=13)
tab <- table(pr,iris_test_category)
print(tab)
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
```

Output:

```
> # Shivam Tawari A-58
> df <- data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> ran <- sample(1:nrow(iris), 0.9 * nrow(iris))
> nor <-function(x) { (x -min(x))/(max(x)-min(x))   }
> iris_norm <- as.data.frame(lapply(iris[,c(1,2,3,4)], nor))
> summary(iris_norm)
  Sepal.Length      Sepal.Width       Petal.Length
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.2222   1st Qu.:0.3333   1st Qu.:0.1017
 Median :0.4167   Median :0.4167   Median :0.5678
 Mean   :0.4287   Mean   :0.4406   Mean   :0.4675
 3rd Qu.:0.5833   3rd Qu.:0.5417   3rd Qu.:0.6949
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```
  Petal.Width
 Min.   :0.00000
 1st Qu.:0.08333
 Median :0.50000
 Mean   :0.45806
 3rd Qu.:0.70833
 Max.   :1.00000
> iris_train <- iris_norm[ran,]
> iris_test <- iris_norm[-ran,]
> iris_target_category <- iris[ran,5]
> iris_test_category <- iris[-ran,5]
> library(class)
> pr <- knn(iris_train,iris_test,cl=iris_target_category,k=13)
> tab <- table(pr,iris_test_category)
> print(tab)
            iris_test_category
pr           setosa versicolor virginica
  setosa          5          0         0
  versicolor      0          7         1
  virginica       0          0         2
> accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
> accuracy(tab)
[1] 93.33333
>
```