***KNN Algorithm***

**Aim:** To Create a Machine Learning model to automate identification of cancer cells. This automatic screening helps in improving the efficiency and time-consuming steps in identification and allows oncologists to spend more time diagnosing. Here we are using ML model that detects cancer cells by applying the k-NN algorithm to measurements of biopsied cells from women with abnormal breast masses by using R programming.

**Algorithm selected**: This problem is classification type. Based on the instance values we predict the instance to be benign and malignant. KNN algorithm is used for modelling classification problems. Euclidean distance is generally used to find distance between K nearest neighbors and feature values. Based on the majority vote, classification is done. Usually, k value is taken as square root of number of training data instances.

Let’s dive in into the steps now

1. **DATA COLLECTION**: Data used in this project is Breast cancer diagnostic dataset from UCI machine Learning repository.

This data set has 569 instances and 32 features including ID and diagnosis (we treat as output- benign & malignant). Rest of them were geometries of cancerous cells includes area, smoothness, radius etc.

1. **DATA PREPARATION**: We will start importing the csv data in R, for this

* d=read.csv("wisc\_bc\_data.csv", stringsAsFactors = FALSE)

# To know the structure and features of data str() is used

* str(d)

'data.frame': 569 obs. of 32 variables:

$ id : int 87139402 8910251 905520 868871 9012568 906539 925291 87880 862989 89827 ...

$ diagnosis : chr "B" "B" "B" "B" ...

$ radius\_mean : num 12.3 10.6 11 11.3 15.2 ...

$ texture\_mean : num 12.4 18.9 16.8 13.4 13.2 ...

$ perimeter\_mean : num 78.8 69.3 70.9 73 97.7 ...

$ area\_mean : num 464 346 373 385 712 ...

$ smoothness\_mean : num 0.1028 0.0969 0.1077 0.1164 0.0796 ...

$ compactness\_mean : num 0.0698 0.1147 0.078 0.1136 0.0693 ...

$ concavity\_mean : num 0.0399 0.0639 0.0305 0.0464 0.0339 ...

$ points\_mean : num 0.037 0.0264 0.0248 0.048 0.0266 ...

$ symmetry\_mean : num 0.196 0.192 0.171 0.177 0.172 ...

These are the some of the features in dataset.

We see that ID and diagnosis are nominal categorical variables. For such type, we must either change them to dummy or exclude from the analysis. Here ID is identification of patients, so this doesn’t provide useful insights. Diagnosis is outcome of the test to be predicted.

Hence, excluding ID column from the data we do so by.

* d=d[-1]
* table(d$diagnosis)

B M

357 212

This is indicating number of benign and malignant as 357 and 212. Now renaming them for clear understanding

* d$diagnosis=factor(d$diagnosis,levels=c("B","M"),labels=c("benign","malignant"))
* round(prop.table(table(d$diagnosis)) \* 100, digits = 1)

benign malignant

62.7 37.3

62.7% of data are aligned to benign and 37.3 to malignant.

Looking at the other features now

* summary(d[c("radius\_mean", "area\_mean", "smoothness\_mean")])

radius\_mean area\_mean smoothness\_mean

Min. : 6.981 Min. : 143.5 Min. :0.05263

1st Qu.:11.700 1st Qu.: 420.3 1st Qu.:0.08637

Median :13.370 Median : 551.1 Median :0.09587

Mean :14.127 Mean : 654.9 Mean :0.09636

3rd Qu.:15.780 3rd Qu.: 782.7 3rd Qu.:0.10530

Max. :28.110 Max. :2501.0 Max. :0.16340

From this table, its evident that there is huge difference in ranges among the features. Smoothness mean comprised of low range values as compared to area mean, so area will have higher impact in distance calculation. Rescaling them helps for better results.

*min-max scaling (Normalization):* This method helps to draw all the values between range 0-1. Function for this is

* normalize= function(x){return ((x - min(x)) / (max(x) - min(x)))}

Applying this to all features, we have to use lapply(),this excludes ID and diagnosis features

* d\_n= as.data.frame(lapply(d[2:31], normalize))
* summary(d\_n$area\_mean)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1. 0.1174 0.1729 0.2169 0.2711 1.0000

The higher range of areas are now changed to 0-1

1. **TRAINING & TEST DATA**: Dividing the data into training and test data to predict the model accuracy. For this, we are dividing 82%(469) instances as training data and 18% (100)data as test data. As the data here is randomly ordered we are directly dividing them, but when they are not randomly ordered its better to do resampling techniques.

* d\_train=d\_n[1:469, ]
* d\_test=d\_n[470:569, ]

For training model,We have to include Diagnosis feature ,first column of the data frame to create labels. We will use these in the next steps of training and evaluating our classifier

* d\_train\_labels=d[1:469, 1]
* d\_test\_labels =d[470:569, 1]

**4.TRAINING THE MODEL**: KNN doesn’t really train the model, it stores the data in structured format. We use class package in R which provides some classification functions. knn() function in class package helps in implementation of algorithm.

* d\_test\_predictn=knn(train=d\_train,test=d\_test,cl=d\_train\_labels,k=21)

k is picked as 21 which is nearest square root of number of training data set. Odd number is usually chosen to eliminate the tie in case of same votings.Class (d\_train\_labels )is the diagnosis feature vector of training set.

* d\_test\_predictn=knn(train=d\_train,test=d\_test,cl=d\_train\_labels,k=21)

**5.EVALUATING THE MODEL:** Cross table() function in gmodels package helps in evaluating the KNN. The output will be the confusion matrix indicating the prediction accuracies and anomalies. Executing this will give results as below.

* > CrossTable(x = d\_test\_labels, y = d\_test\_predictn,prop.chisq=FALSE)

Table

Description automatically generated

Accuracy rate of this model is 98%. We have to decide which errors are dangerous based on the problem we have.In this case,

There are 2 False Negatives. These are extremely dangerous in this problem. The predicted value is benign, but it was malignant in reality. These errors are very costly as this would result in late diagnosis or death of patient.

There are 0 False Positive. The predicted value is, malignant where the actual it is benign. This results in financial loss or mental trauma to patient. But when compared to FN it is not as dangerous as FN was.

Let’s explore with more k values,

|  |  |  |
| --- | --- | --- |
| K values | FN | FP |
| 1 | 1 | 3 |
| 50 | 4 | 0 |
| 80 | 6 | 0 |
| 150 | 9 | 0 |

As the K value increase False Negatives are increasing which shows the effect of more benign values. Benign values are more in number in dataset than malignant. After trying with random K values, it’s evident that k=21 provides better results than other k values. So, I am not trying to change K.

**6.IMPROVING THE MODEL**: Let’s explore on improvisations of the model. Our aim is to reduce the dangerous False Negatives

Using other feature transformation techniques like z-score standardization and BoxCox techniques and performing KNN algorithm.

*Z-score standardization*: It rescales the feature values between negative and positive values. We have no predefined min-max values here.

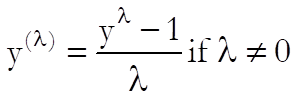
* d\_z=as.data.frame(scale(d[-1]))
* summary(d\_z$area\_mean)
* d\_train\_z= d\_z[1:469, ]
* d\_test\_z = d\_z[470:569, ]
* d\_train\_labels=d[1:469, 1]
* d\_test\_labels=d[470:569, 1]
* d\_test\_predictz=knn(train = d\_train\_z, test = d\_test\_z,cl =d\_train\_labels, k = 21)
* CrossTable(x =d\_test\_labels,y =d\_test\_predictz,prop.chisq = FALSE)

Table

Description automatically generated

False Negatives increased to 5. Accuracy rate is 95%.

*BoxCox Transformation:* This transforms our data such that it closely resembles normal distribution. It is available in caret library of C. It uses lambda value for rescaling ,R selects the lambda randomly and performs the operation .Following is the formula and R code executing the KNN algorithm.



* d1=d[2:31]
* d1\_test\_predictbcx=preProcess(d1,method=c("BoxCox"))
* d1\_boxcox=predict(d1\_test\_predictbcx,d1)
* d\_train\_bcx= d1\_boxcox[1:469, ]
* d\_test\_bcx = d1\_boxcox[470:569, ]
* d\_train\_labels=d[1:469, 1]
* d\_test\_labels=d[470:569, 1]
* d\_test\_predictbcx=knn(train = d\_train\_bcx, test = d\_test\_bcx,cl =d\_train\_labels, k = 21)
* CrossTable(x =d\_test\_labels,y =d\_test\_predictbcx,prop.chisq = FALSE)

Table

Description automatically generated with medium confidence

There are 10 False Negatives and 9 False positives. Overall accuracy rate is 81%.

*Normalization using Resampling*: We are going to do normalization using random resampling technique to run the model and check if the results are better.

* d\_n= as.data.frame(lapply(d[2:31], normalize))
* d\_resample= sort(sample(nrow(d\_n), nrow(d\_n)\*.82))
* train\_resample = d\_n[d\_resample,]
* test\_resample = d\_n[-d\_resample,]
* x= d[d\_resample,]
* y=d[-d\_resample,]
* d\_train\_label=x[1:nrow(train\_resample), 1]
* d\_test\_label =y[1:nrow(test\_resample), 1]
* d\_test\_pred=knn(train=train\_resample,test=test\_resample,cl=d\_train\_label,k=11)
* CrossTable(x =d\_test\_label, y =d\_test\_pred,prop.chisq=FALSE)

Table

Description automatically generated

By repeating the process multiple time with different k values and different training data set percentage we were able to arrive at good results at k=11 and training data as 82% using resampling.

False Negatives and False positives are 1 each. Accuracy rate is 98%.

Using resampling we reduced False Negative by 1.

**Conclusion**:

* Finally, we were able to model 2 KNN algorithm with 98% model accuracy. The models are

1. KNN algorithm with “***Range normalization***” feature scaling method by considering k = 21 (followed general method square root of number of training set observations) and able fit the model with 98% accuracy with 2 false negative costly errors.
2. KNN algorithm with “***Randomization***” sampling technique with 82% of training data set and 18% test data set with k-factor value as 11(randomly tested) and model accuracy of 98% with one false negative and one false positive error.

* Second improved model seems little better than base model because it has 1 false negative costly error and has 1 false positive error which is also harmful because it causes unnecessary financial burden to the patient with mental trauma.