Supervised Classification

Annamaria Porreca

University of «G.d'Annunzio», Chieti-Pescara, Italy

K-NN in R

knn() does predictions using a single command. The function requires four inputs:

1. A matrix containing the predictors associated with the training data, labeled train.X below.

2. A matrix containing the predictors associated with the data for which we wish to make predictions, labeled test.X below.

3. A vector containing the class labels for the training observations, labeled train.Direction below.

4. A value for K, the number of nearest neighbours to be used by the classifier.

Better to choose k which is of odd value!

Generally k gets decided on the square root of number of data points. But a large k value has benefits which include reducing the variance due to the noisy data; the side effect being developing a bias due to which the learner tends to ignore the smaller patterns which may have useful insights.

K-NN on the IRIS dataset

```
library("class")
library("datasets")
data("iris")
str(iris)
head(iris)


set.seed(123)
random<- sample(rep(1:150)) # randomly generate numbers from 1 to 150
random


iris<- iris[random,] #randomize "iris"
head(iris)
```

```
normalize <- function(x){
  return ((x-min(x))/(max(x)-min(x)))
}

iris.new<- as.data.frame(lapply(iris[,c(1,2,3,4)],normalize))
head(iris.new)

iris.train<- iris.new[1:100,]
iris.train.label<- iris[1:100,5]
iris.test<- iris.new[101:150,]
iris.test.label<- iris[101:150,5]
summary(iris.new)

?knn
predict<- knn(train=iris.train, test=iris.test, cl=iris.train.label, k=5)

table(iris.test.label, predict)
```

Exercise

1. Do K-NN on IRIS dataset
2. Try different values of “k” and choose the one with maximum accuracy.
3. Plot the misclassification error rate for K from 1 to 20.

Solution

```
# Do K-NN on IRIS dataset

# Try different values of “k” and choose the one with maximum accuracy.

# Plot the misclassification error rate for K in 1:20

length(predict)

err=sum(iris.test.label==predict)

error_rate=1-err/length(predict)

error_rate
```

```
res=rep(NA,20)
for(i in 1:20){mod=knn(train=iris.train, test=iris.test, cl=iris.train.label, k=i)
        err=sum(iris.test.label==mod)
        res[i]=1-err/length(mod)}
print(res)

plot(1:20,res,type="l",col="red", xlab="Neighbours", ylab="Misclassification error rate")

plot(1:20,1-res,type="l",col="blue", xlab="Neighbours", ylab="Accuracy")
```

Exercise

library(clValid)

data(mouse)

Find the optimal k for the mouse dataset (rows are genes and columns as the samples)

Try to change three different sizes of the training and the test set and verify if the optimal k changes.

Look at the solution in the .R file

Exercise

# Use the PimaIndiansDiabetes datasets

# Use knn (tuneLength = 3) and logistic regression

# Use k-fold cross-validation from 5 to 20 splits

# Make a code to understand for which k-fold we get best Accuracy considering LR and Knn for different k