# Descubrimiento del Conocimiento usando herramientas de Big Data Módulo 2

Marco Andrés Vázquez Hernández

Práctica KNN.

Septiembre de 2018

Instituto Politécnico Nacional

## Descripción

Con las muesras de entrenamiento, clasificar las nuevas instancias. Suerte

## Carga de archivos

```
setwd("C:/Users/marco/IPN_BigData/Modulo2/Práctica_KNN")

train<-read.csv("Ejercicio Knn.csv", skip=1, nrows= 12)
eval<-read.csv("Ejercicio Knn.csv", skip=16, nrows=4, header=F)
colnames(eval)<-colnames(train)
```

## Transformación de datos

Se convirtió la variable "Invertir" a dicotómica y se juntaron los datos para tomar todos los valores en la normalización.

```
train$Invertir<-ifelse(as.character(train$Invertir)=="Si",1,0)
eval$Invertir<-ifelse(as.character(eval$Invertir)=="Si",1,0)
dats<-rbind(train,eval)
```

Se creó la función para normalizar los datos:

```
Normalizar<- function (x){
  if (all(is.na(x))){
    return(rep(NA, length(x)))
  } else if (sum(x)==0){
    return(rep(0,length(x)))
  } else if (min(x)==max(x) & max(x)!=0){
    return(rep(1,length(x)))
  } else
  return((x-min(x))/(max(x)-min(x)))
}
```

Se aplicó a la base con todos los valores y después se separaron de nuevo el conjunto de entrenamiento y el de evaluación:

```
aux<-sapply(dats[,1:(ncol(dats)-1)], Normalizar)
train2<-as.data.frame(cbind(aux[1:nrow(train),], train[,ncol(train)]))
colnames(train2)[ncol(train2)]<-"Invertir"
eval2<-as.data.frame(cbind(aux[(nrow(train)+1):nrow(aux),], eval[,ncol(eval)]))
colnames(eval2)[ncol(eval2)]<-"Invertir"
```

Una muestra de los datos normalizados queda:

```
head(train2)
```

```
##       Precio Metros.Cuadrados Baños Cuartos Estacionamiento Mantenimiento
## 1 0.02439024           0.0625     0       0             0.0     0.1052632
## 2 0.07317073           0.3125     0       1             0.0     0.0000000
## 3 0.26829268           0.2500     0       0             0.5     0.4736842
## 4 0.39024390           0.3125     0       0             0.5     0.3684211
## 5 0.09756098           0.2125     0       0             0.0     0.3684211
```

```
## 6 0.14634146           0.6875     1       0             0.5     0.5263158
##   Invertir
## 1        1
## 2        0
## 3        1
## 4        1
## 5        0
## 6        0
```

```
head(eval2)
```

```
##       Precio Metros.Cuadrados Baños Cuartos Estacionamiento Mantenimiento
## 1 0.04878049           0.1875     0       0             0.5     0.2105263
## 2 0.21951220           0.2500     0       0             0.5     0.4736842
## 3 0.34146341           0.3375     0       1             0.5     0.6842105
## 4 0.73170732           0.8125     1       1             0.5     0.8947368
##   Invertir
## 1       NA
## 2       NA
## 3       NA
## 4       NA
```

# Algoritmo

Se crearon matrices para medir las distancias de los puntos de evaluación a cada uno de los puntos en el conjunto de entrenamiento:

```
aux<-data.frame()
for(i in 1:nrow(eval2)){
  for(j in 1:nrow(train2)){
    aux[j,i]<-dist(rbind(train2[j,-ncol(train2)], eval2[i,-ncol(eval2)]), method="euclidean")
  }
}
euclidian<-aux
euclidian
```

```
##           V1         V2        V3        V4
## 1  0.5265930 0.67746801 1.3271550 1.9839522
## 2  1.1447887 1.22462190 0.8892413 1.6535490
## 3  0.3483446 0.04878049 1.0282663 1.6457353
## 4  0.3964253 0.21008545 1.0501083 1.6259168
## 5  0.5271957 0.52664854 1.1936651 1.8136014
## 6  1.1658650 1.09523070 1.4783448 1.2222979
## 7  2.1067139 1.92319253 1.4907380 0.6068093
## 8  1.8110267 1.67441738 1.2358558 0.2638279
## 9  1.3394420 1.17460247 0.4087944 1.0869831
## 10 0.2806603 0.09622996 1.0511393 1.7060374
## 11 0.5760704 0.76489368 1.3959397 2.0606081
## 12 0.1341810 0.34536999 1.1659718 1.8527778
```

```
aux<-data.frame()
for(i in 1:nrow(eval2)){
  for(j in 1:nrow(train2)){
    aux[j,i]<-dist(rbind(train2[j,-ncol(train2)], eval2[i,-ncol(eval2)]), method="maximum")
  }
```

```
}
max<-aux
max
```

```
##           V1         V2        V3     V4
## 1  0.5000000 0.50000000 1.0000000 1.0000
## 2  1.0000000 1.00000000 0.6842105 1.0000
## 3  0.2631579 0.04878049 1.0000000 1.0000
## 4  0.3414634 0.17073171 1.0000000 1.0000
## 5  0.5000000 0.50000000 1.0000000 1.0000
## 6  1.0000000 1.00000000 1.0000000 1.0000
## 7  1.0000000 1.00000000 1.0000000 0.5000
## 8  1.0000000 1.00000000 1.0000000 0.1875
## 9  1.0000000 1.00000000 0.3500000 1.0000
## 10 0.2631579 0.07317073 1.0000000 1.0000
## 11 0.5000000 0.50000000 1.0000000 1.0000
## 12 0.1250000 0.26315789 1.0000000 1.0000
```

Se creó la función para tomar el promedio del pronóstico de los k-vecinos más cercanos de cada punto de evaluación:

```
kesimo<-function(v,k){
  mean(train2$Invertir[which(v %in% sort(v)[1:k])])
}
```

Se aplicó a cada punto de evaluación para k=1,3 y 5 para la distancia euclidiana quedando:

```
matriz_inversion_e<-data.frame()
for(v in 1:nrow(eval2)){
  for(k in 1:3){
    matriz_inversion_e[v,k]<-kesimo(euclidian[,v],k*2-1)
  }
}
colnames(matriz_inversion_e)<-c("k=1","k=3","k=5")
matriz_inversion_e
```

```
##   k=1       k=3 k=5
## 1   1 1.0000000 1.0
## 2   1 1.0000000 0.8
## 3   1 0.6666667 0.8
## 4   1 0.6666667 0.6
```

De donde se puede observar que para k=1 sugiere invertir en todos los casos, mientras que para k=3 y 5; se puede definir un parámetro de "confianza", si se tomara .5 (redondeo) se invertiría en todos los casos, si se tomara un parámetro de confianza más alto, tal vez quedarían fuera las inversiones en los casos 3 y 4 (e incluso 2).

Para la distancia máxima se tiene:

```
matriz_inversion_max<-data.frame()
for(v in 1:nrow(eval2)){
  for(k in 1:3){
    matriz_inversion_max[v,k]<-kesimo(max[,v],k*2-1)
  }
}
colnames(matriz_inversion_max)<-c("k=1","k=3","k=5")
matriz_inversion_max
```

```
##   k=1       k=3       k=5
## 1   1 1.0000000 0.7142857
## 2   1 1.0000000 0.7142857
## 3   1 0.5833333 0.5833333
## 4   1 0.5833333 0.5833333
```

En donde para k=1 se sugiere invertir en todos los casos y para k=3 y 5 podría variar de acuerdo a un parámetro de "confianza".