Prediction & Recommendation

Mohammed Amir

April 7, 2017

Dinesafe = read.csv("D:/CAPSTONE/CAPSTONE/DATASET/Final\_DineSafe.csv", na.strings='NULL')

## select a subset of dataset  
Dinesafe1 <- unique(Dinesafe[c(2,5:7)])  
  
## Select unique rows  
Dinesafe2 <- unique(Dinesafe1)  
  
nrow(Dinesafe2)

## [1] 2723

## Index the cuisine Type label  
CUISINE\_IDX <- function(CUISINE)  
{  
 if(CUISINE == "African")  
 {  
 print ("1")  
 }  
 else  
 {  
 if(CUISINE == "Bakeries")  
 {  
 print ("2")   
 }  
 else  
 {  
 if(CUISINE == "Bar")  
 {  
 print ("3")   
 }  
 else  
 {  
 if(CUISINE == "Cafe")  
 {  
 print ("4")   
 }  
 else  
 {  
 if(CUISINE == "Caribbean")  
 {  
 print ("5")   
 }  
 else  
 {  
 if(CUISINE == "Deli")  
 {  
 print ("6")   
 }  
 else  
 {  
 if(CUISINE == "Dessert")  
 {  
 print ("7")   
 }  
 else  
 {  
 if(CUISINE == "European")  
 {  
 print ("8")   
 }  
 else  
 {  
 if(CUISINE == "Far Eastern")  
 {  
 print ("9")   
 }  
 else  
 {  
 if(CUISINE == "Mediterranean")  
 {  
 print ("10")   
 }  
 else  
 {  
 if(CUISINE == "Middle Eastern")  
 {  
 print ("11")   
 }  
 else  
 {  
 if(CUISINE == "North American")  
 {  
 print ("12")   
 }  
 else  
 {  
 if(CUISINE == "Juicery")  
 {  
 print ("13")   
 }  
 else  
 {  
 if(CUISINE == "Pastries")  
 {  
 print ("14")   
 }  
 else  
 {  
 if(CUISINE == "South Asian")  
 {  
 print ("15")   
 }  
 else  
 {  
 if(CUISINE == "South East Asian")  
 {  
 print ("16")   
 }  
 else  
 {  
 if(CUISINE == "Latin American")  
 {  
 print ("17")   
 }  
 else  
 {  
 print ("0")  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
   
 }  
 }  
}  
  
## Apply the Index function to cuisine type column  
Dinesafe2$CUISINE\_IDX <- mapply(CUISINE\_IDX,Dinesafe2$CUISINE\_TYPE)

## Create a binary matrix based on Cuisine\_Idx

Dinesafe2$African <- ifelse(Dinesafe2$CUISINE\_TYPE == "African",1,0)  
Dinesafe2$Bakeries <- ifelse(Dinesafe2$CUISINE\_TYPE == "Bakeries",1,0)  
Dinesafe2$Bar <- ifelse(Dinesafe2$CUISINE\_TYPE == "Bar",1,0)  
Dinesafe2$Cafe <- ifelse(Dinesafe2$CUISINE\_TYPE == "Cafe",1,0)  
Dinesafe2$Caribbean <- ifelse(Dinesafe2$CUISINE\_TYPE == "Caribbean",1,0)  
Dinesafe2$Deli <- ifelse(Dinesafe2$CUISINE\_TYPE == "Deli",1,0)  
Dinesafe2$Dessert <- ifelse(Dinesafe2$CUISINE\_TYPE == "Dessert",1,0)  
Dinesafe2$European <- ifelse(Dinesafe2$CUISINE\_TYPE == "European",1,0)  
Dinesafe2$FarEastern <- ifelse(Dinesafe2$CUISINE\_TYPE == "Far Eastern",1,0)  
Dinesafe2$Mediterranean <- ifelse(Dinesafe2$CUISINE\_TYPE == "Mediterranean",1,0)  
Dinesafe2$MidEastern <- ifelse(Dinesafe2$CUISINE\_TYPE == "Middle Eastern",1,0)  
Dinesafe2$NAmerican <- ifelse(Dinesafe2$CUISINE\_TYPE == "North American",1,0)  
Dinesafe2$Juicery <- ifelse(Dinesafe2$CUISINE\_TYPE == "Juicery",1,0)  
Dinesafe2$Pastries <- ifelse(Dinesafe2$CUISINE\_TYPE == "Pastries",1,0)  
Dinesafe2$SouthAsian <- ifelse(Dinesafe2$CUISINE\_TYPE == "South Asian",1,0)  
Dinesafe2$SEastAsian <- ifelse(Dinesafe2$CUISINE\_TYPE == "South East Asian",1,0)  
Dinesafe2$LAmerican <- ifelse(Dinesafe2$CUISINE\_TYPE == "Latin American",1,0)

str(Dinesafe2)

## 'data.frame': 2723 obs. of 22 variables:  
## $ ESTABLISHMENT\_ID: int 1222579 1222807 1223056 9000004 9000026 9000029 9000031 9000046 9000109 9000116 ...  
## $ REVIEW : num 5 3.5 3 4 2.5 2.5 2.5 2.5 3 2 ...  
## $ VALUE : num 1 1 2 1 2 2 2 2 2 2 ...  
## $ CUISINE\_TYPE : Factor w/ 17 levels "African","Bakeries",..: 16 9 8 8 8 8 8 8 3 4 ...  
## $ CUISINE\_IDX : chr "15" "9" "8" "8" ...  
## $ African : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bakeries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bar : num 0 0 0 0 0 0 0 0 1 0 ...  
## $ Cafe : num 0 0 0 0 0 0 0 0 0 1 ...  
## $ Caribbean : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Deli : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Dessert : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ European : num 0 0 1 1 1 1 1 1 0 0 ...  
## $ FarEastern : num 0 1 0 0 0 0 0 0 0 0 ...  
## $ Mediterranean : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ MidEastern : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ NAmerican : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Juicery : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Pastries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ SouthAsian : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ SEastAsian : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ LAmerican : num 0 0 0 0 0 0 0 0 0 0 ...

head(Dinesafe2)

## ESTABLISHMENT\_ID REVIEW VALUE CUISINE\_TYPE CUISINE\_IDX African Bakeries  
## 1 1222579 5.0 1 South Asian 15 0 0  
## 2 1222807 3.5 1 Far Eastern 9 0 0  
## 9 1223056 3.0 2 European 8 0 0  
## 13 9000004 4.0 1 European 8 0 0  
## 18 9000026 2.5 2 European 8 0 0  
## 23 9000029 2.5 2 European 8 0 0  
## Bar Cafe Caribbean Deli Dessert European FarEastern Mediterranean  
## 1 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 1 0  
## 9 0 0 0 0 0 1 0 0  
## 13 0 0 0 0 0 1 0 0  
## 18 0 0 0 0 0 1 0 0  
## 23 0 0 0 0 0 1 0 0  
## MidEastern NAmerican Juicery Pastries SouthAsian SEastAsian LAmerican  
## 1 0 0 0 0 1 0 0  
## 2 0 0 0 0 0 0 0  
## 9 0 0 0 0 0 0 0  
## 13 0 0 0 0 0 0 0  
## 18 0 0 0 0 0 0 0  
## 23 0 0 0 0 0 0 0

#Dinesafe3 <- subset( Dinesafe2, select = -c( 1 ))  
#Dinesafe3  
#str(Dinesafe3)  
  
Dinesafe2$CUISINE\_IDX <- as.numeric(Dinesafe2$CUISINE\_IDX)  
str(Dinesafe2)

## 'data.frame': 2723 obs. of 22 variables:  
## $ ESTABLISHMENT\_ID: int 1222579 1222807 1223056 9000004 9000026 9000029 9000031 9000046 9000109 9000116 ...  
## $ REVIEW : num 5 3.5 3 4 2.5 2.5 2.5 2.5 3 2 ...  
## $ VALUE : num 1 1 2 1 2 2 2 2 2 2 ...  
## $ CUISINE\_TYPE : Factor w/ 17 levels "African","Bakeries",..: 16 9 8 8 8 8 8 8 3 4 ...  
## $ CUISINE\_IDX : num 15 9 8 8 8 8 8 8 3 4 ...  
## $ African : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bakeries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bar : num 0 0 0 0 0 0 0 0 1 0 ...  
## $ Cafe : num 0 0 0 0 0 0 0 0 0 1 ...  
## $ Caribbean : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Deli : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Dessert : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ European : num 0 0 1 1 1 1 1 1 0 0 ...  
## $ FarEastern : num 0 1 0 0 0 0 0 0 0 0 ...  
## $ Mediterranean : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ MidEastern : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ NAmerican : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Juicery : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Pastries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ SouthAsian : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ SEastAsian : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ LAmerican : num 0 0 0 0 0 0 0 0 0 0 ...

#Normalize the dataset feature  
normalize <- function(x)  
{  
num <- x - min(x)  
denom <- max(x) - min(x)  
return (num/denom)  
}

#Apply normalizeto dataset feature  
Norm\_RATING <- as.data.frame(lapply(Dinesafe2[,c(2,3,5)], normalize))  
str(Norm\_RATING)

## 'data.frame': 2723 obs. of 3 variables:  
## $ REVIEW : num 1 0.625 0.5 0.75 0.375 0.375 0.375 0.375 0.5 0.25 ...  
## $ VALUE : num 0 0 0.333 0 0.333 ...  
## $ CUISINE\_IDX: num 0.882 0.529 0.471 0.471 0.471 ...

#str(Norm\_Dinesafe1)  
Norm\_Dinesafe <- subset( Dinesafe2, select = -c( 2,3,5 ))  
#str(Norm\_Dinesafe)  
  
  
Norm\_Dinesafe5 <- cbind.data.frame(Norm\_Dinesafe, Norm\_RATING)

Norm\_Dinesafe6 <- subset( Norm\_Dinesafe5, select = -c( 1,2 ))  
  
str(Norm\_Dinesafe6)

## 'data.frame': 2723 obs. of 20 variables:  
## $ African : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bakeries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Bar : num 0 0 0 0 0 0 0 0 1 0 ...  
## $ Cafe : num 0 0 0 0 0 0 0 0 0 1 ...  
## $ Caribbean : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Deli : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Dessert : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ European : num 0 0 1 1 1 1 1 1 0 0 ...  
## $ FarEastern : num 0 1 0 0 0 0 0 0 0 0 ...  
## $ Mediterranean: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ MidEastern : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ NAmerican : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Juicery : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Pastries : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ SouthAsian : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ SEastAsian : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ LAmerican : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ REVIEW : num 1 0.625 0.5 0.75 0.375 0.375 0.375 0.375 0.5 0.25 ...  
## $ VALUE : num 0 0 0.333 0 0.333 ...  
## $ CUISINE\_IDX : num 0.882 0.529 0.471 0.471 0.471 ...

head(Norm\_Dinesafe6)

## African Bakeries Bar Cafe Caribbean Deli Dessert European FarEastern  
## 1 0 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0 1  
## 9 0 0 0 0 0 0 0 1 0  
## 13 0 0 0 0 0 0 0 1 0  
## 18 0 0 0 0 0 0 0 1 0  
## 23 0 0 0 0 0 0 0 1 0  
## Mediterranean MidEastern NAmerican Juicery Pastries SouthAsian  
## 1 0 0 0 0 0 1  
## 2 0 0 0 0 0 0  
## 9 0 0 0 0 0 0  
## 13 0 0 0 0 0 0  
## 18 0 0 0 0 0 0  
## 23 0 0 0 0 0 0  
## SEastAsian LAmerican REVIEW VALUE CUISINE\_IDX  
## 1 0 0 1.000 0.0000000 0.8823529  
## 2 0 0 0.625 0.0000000 0.5294118  
## 9 0 0 0.500 0.3333333 0.4705882  
## 13 0 0 0.750 0.0000000 0.4705882  
## 18 0 0 0.375 0.3333333 0.4705882  
## 23 0 0 0.375 0.3333333 0.4705882

nrow(Norm\_Dinesafe5)

## [1] 2723

nrow(Norm\_Dinesafe6)

## [1] 2723

#set.seed(9850)  
#gp <- runif(nrow(Norm\_Dinesafe6))  
#Dinesafe4 <- Norm\_Dinesafe6[order(gp),]  
#head(Dinesafe4)

## create a feature  
Dine\_train <- Norm\_Dinesafe6[1:2000,]  
Dine\_test <- Norm\_Dinesafe6[2001:2723,]  
nrow(Dine\_train)

## [1] 2000

nrow(Dine\_test)

## [1] 723

Dine\_trainLabel <- Dinesafe2[1:2000,4]  
Dine\_testLabel <- Dinesafe2[2001:2723,4]  
  
NROW(Dine\_trainLabel)

## [1] 2000

NROW(Dine\_testLabel)

## [1] 723

# Determine best K value in KNN Crosss Validation  
set.seed(3333)  
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)  
  
  
knn\_fit <- train(CUISINE\_IDX~., data = Dine\_train, method = "knn", trControl=trctrl, preProcess = c("center", "scale"),tuneLength = 10)

knn\_fit  
plot(knn\_fit)

k-Nearest Neighbors

2000 samples

19 predictor

Pre-processing: centered (19), scaled (19)

Resampling: Cross-Validated (10 fold, repeated 3 times)

Summary of sample sizes: 1800, 1800, 1799, 1800, 1800, 1800, ...

Resampling results across tuning parameters:

k RMSE Rsquared

5 5.834634e-16 1.0000000

7 2.668497e-03 0.9996632

9 1.729579e-02 0.9857054

11 3.022194e-02 0.9692707

13 4.094209e-02 0.9503906

15 4.964589e-02 0.9374691

17 6.042854e-02 0.9077873

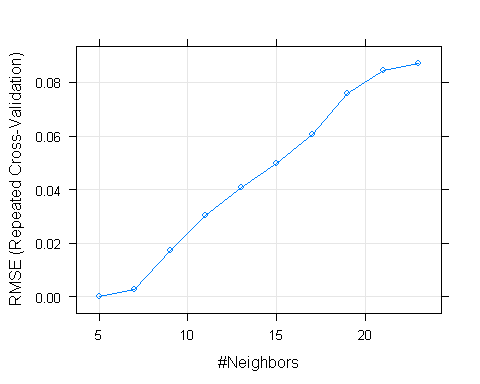
19 7.599317e-02 0.8641550

21 8.439008e-02 0.8395811

23 8.720328e-02 0.8307749

RMSE was used to select the optimal model using the smallest value.

The final value used for the model was k = 5.



model <- knn(train = Dine\_train, test = Dine\_test, cl = Dine\_trainLabel, k = 5)  
model

## 17 Levels: African Bakeries Bar Cafe Caribbean Deli Dessert ... South East Asian

table (Dine\_testLabel, model)

## model  
## Dine\_testLabel African Bakeries Bar Cafe Caribbean Deli Dessert  
## African 5 0 0 0 0 0 0  
## Bakeries 0 2 0 0 0 0 0  
## Bar 0 0 9 0 0 0 0  
## Cafe 0 0 0 203 0 0 0  
## Caribbean 0 0 0 0 6 0 0  
## Deli 0 0 0 0 0 125 0  
## Dessert 0 0 0 0 0 0 12  
## European 0 0 0 0 0 0 0  
## Far Eastern 0 0 0 0 0 0 0  
## Juicery & Smoothies 0 0 0 0 0 0 0  
## Latin American 0 0 0 0 0 0 0  
## Mediterranean 0 0 0 0 0 0 0  
## Middle Eastern 0 0 0 0 0 0 0  
## North American 0 0 0 0 0 0 0  
## Pastries 0 0 0 0 0 0 0  
## South Asian 0 0 0 0 0 0 0  
## South East Asian 0 0 0 0 0 0 0  
## model  
## Dine\_testLabel European Far Eastern Juicery & Smoothies  
## African 0 0 0  
## Bakeries 0 0 0  
## Bar 0 0 0  
## Cafe 0 0 0  
## Caribbean 0 0 0  
## Deli 0 0 0  
## Dessert 0 0 0  
## European 96 0 0  
## Far Eastern 0 60 0  
## Juicery & Smoothies 0 0 21  
## Latin American 0 0 0  
## Mediterranean 0 0 0  
## Middle Eastern 0 0 0  
## North American 0 0 0  
## Pastries 0 0 0  
## South Asian 0 0 0  
## South East Asian 0 0 0  
## model  
## Dine\_testLabel Latin American Mediterranean Middle Eastern  
## African 0 0 0  
## Bakeries 0 0 0  
## Bar 0 0 0  
## Cafe 0 0 0  
## Caribbean 0 0 0  
## Deli 0 0 0  
## Dessert 0 0 0  
## European 0 0 0  
## Far Eastern 0 0 0  
## Juicery & Smoothies 0 0 0  
## Latin American 18 0 0  
## Mediterranean 0 31 0  
## Middle Eastern 0 0 4  
## North American 0 0 0  
## Pastries 0 0 0  
## South Asian 0 0 0  
## South East Asian 0 0 0  
## model  
## Dine\_testLabel North American Pastries South Asian South East Asian  
## African 0 0 0 0  
## Bakeries 0 0 0 0  
## Bar 0 0 0 0  
## Cafe 0 0 0 0  
## Caribbean 0 0 0 0  
## Deli 0 0 0 0  
## Dessert 0 0 0 0  
## European 0 0 0 0  
## Far Eastern 0 0 0 0  
## Juicery & Smoothies 0 0 0 0  
## Latin American 0 0 0 0  
## Mediterranean 0 0 0 0  
## Middle Eastern 0 0 0 0  
## North American 91 0 0 0  
## Pastries 0 13 0 0  
## South Asian 0 0 7 0  
## South East Asian 0 0 0 20

## Accurry where predicted value is not equal to given label  
sum(model != Dine\_testLabel)

## [1] 0

confusion <- confusionMatrix(model, Dine\_testLabel )  
plot <- ggplot(as.data.frame(as.table(confusion)))

# Put `iris.testLabels` in a data frame  
DineTestLabels <- data.frame(Dine\_testLabel)  
  
# Merge `iris\_pred` and `iris.testLabels`   
merge <- data.frame(model, Dine\_testLabel)  
  
# Specify column names for `merge`  
names(merge) <- c("Predicted Cuisine", "Observed Cuisine")  
  
# Inspect `merge`   
head(merge,10)

## Predicted Cuisine Observed Cuisine  
## 1 North American North American  
## 2 North American North American  
## 3 Dessert Dessert  
## 4 Cafe Cafe  
## 5 Dessert Dessert  
## 6 European European  
## 7 Deli Deli  
## 8 Cafe Cafe  
## 9 European European  
## 10 Latin American Latin American

## RECOMMENDATION   
  
library(class)  
library(caret)  
#library(lattice)  
#library(ggplot2)  
  
#load dataset  
recommender <- Dine\_test

## Create a matrix using euclidean distance   
distances <- as.matrix(dist(recommender , method="euclidean"))

## Computes the nearest neighbors for i  
k.nearest.neighbors <- function(i, recommender, k = 5)  
{  
 ordered.neighbors <- order(recommender[i, ])  
 return(ordered.neighbors[2:(k + 1)])  
}

## calculate probability of the closest restaurant   
seen.probability <- function(cuisine, restaurant, recommender, distances, k = 25)  
{  
 neighbors <- k.nearest.neighbors(which(row.names(recommender) == restaurant), distances, k)  
 return(mean(recommender[neighbors, cuisine]))  
}

## Predict a recommendation based on cuisine input, recommender matrix, distance between restaurants and K value   
most.probable.recommend <- function(cuisine, recommender, distances, k = 25)  
{  
 probabilities <- rep(0, nrow(recommender))  
 for (i in 1:nrow(recommender)) { # For each restaurant  
 if (recommender[i,cuisine] == 1) {  
 next   
 }  
 probabilities[i] <- seen.probability(cuisine, row.names(recommender)[i], recommender, distances, k)  
 }  
 return(order(probabilities, decreasing=T))  
}

cuisine <- "African"  
listing <- most.probable.recommend(cuisine, recommender, distances)  
rownames(recommender)[listing[1:3]]

## [1] "12970" "12996" "13057"

Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.