KNN: Riding Lawn Mowers

Example from Data Mining for Business Analytics Chapter 7

Algorithm identifies k observations that are "similar" or nearest to the new record being predicted and then uses the average response value (regression) or the most common class (classification) of those k observations as the predicted output.

- Different Distance Measures can be use to determine the nearest neighbors. With Euclidean being the most common and the defualt for most r packages.
- Supervised Machine Learning: Requires Training & Testing dataset
- Used to solve both classification and regression problems

Measuring Distance Between Records

Distance measures determine similarity between observations.

1. **Euclidean Distance:** Measures distance based on a stright line, using the shortest path between two observations.

- Most common distance measure
- More sensitive to outliers
- Computing is computationally slower than Manhattan as it requires square roots
- Variable measurement scales should be standardized before running KNN (typically on 0 to 1 scale)

2. **Manhattan Distance:** Measures distance based on point to point travel time (city-block)

- Used in One Hot Encoding 0/1 indicator variables
- Computing is computationally faster as it only requires additions and subtractions

```
library(caret) #Trains the KNN Model
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(FNN) #Run KNN Model
```

Get Data

For this example we will be using the RidingMowers.csv that accompanies the book, Data Mining for Business Analytics with R. Which has 24 household observations, with income, lot size and lawn mower ownership status.

```
#Find and Set the Working Directory
getwd()
```

```
## [1] "/Users/amandapiter/Documents/Projects/Github?"
```

```
setwd("/Users/amandapiter/Documents/Projects/Github?")

mower.df <- read.csv("RidingMowers.csv")
head(mower.df)
```

```
##   Income Lot_Size Ownership
## 1   60.0     18.4     Owner
## 2   85.5     16.8     Owner
## 3   64.8     21.6     Owner
## 4   61.5     20.8     Owner
## 5   87.0     23.6     Owner
## 6  110.1     19.2     Owner
```

Create Trainings and Testing Datasets

Before we can create our knn model we need to split our dataset into a training and testing (also know as validation) datasets. Here a 70/30 approach is used.

```
set.seed(123)
train.index <- sample(row.names(mower.df), 0.7*dim(mower.df)[1])
valid.index <- setdiff(row.names(mower.df), train.index)
train.df <- mower.df[train.index,]
valid.df <- mower.df[valid.index,]

head(train.df)
```

```
##    Income Lot_Size Ownership
## 15   64.8     17.2  Nonowner
## 19   59.4     16.0  Nonowner
## 14   52.8     20.8  Nonowner
## 3    64.8     21.6     Owner
## 10   93.0     20.8     Owner
## 18   49.2     17.6  Nonowner
```

New Observation

We have a new homeowner and would like to determine if they are likely to purchase a riding lawnmower based on their income and the lot size of their new home.

```
#Create a New Dataframe for the New Household
new.df <- data.frame(Income = 60, Lot_Size = 20)
head(new.df)
```

```
##   Income Lot_Size
## 1     60       20
```

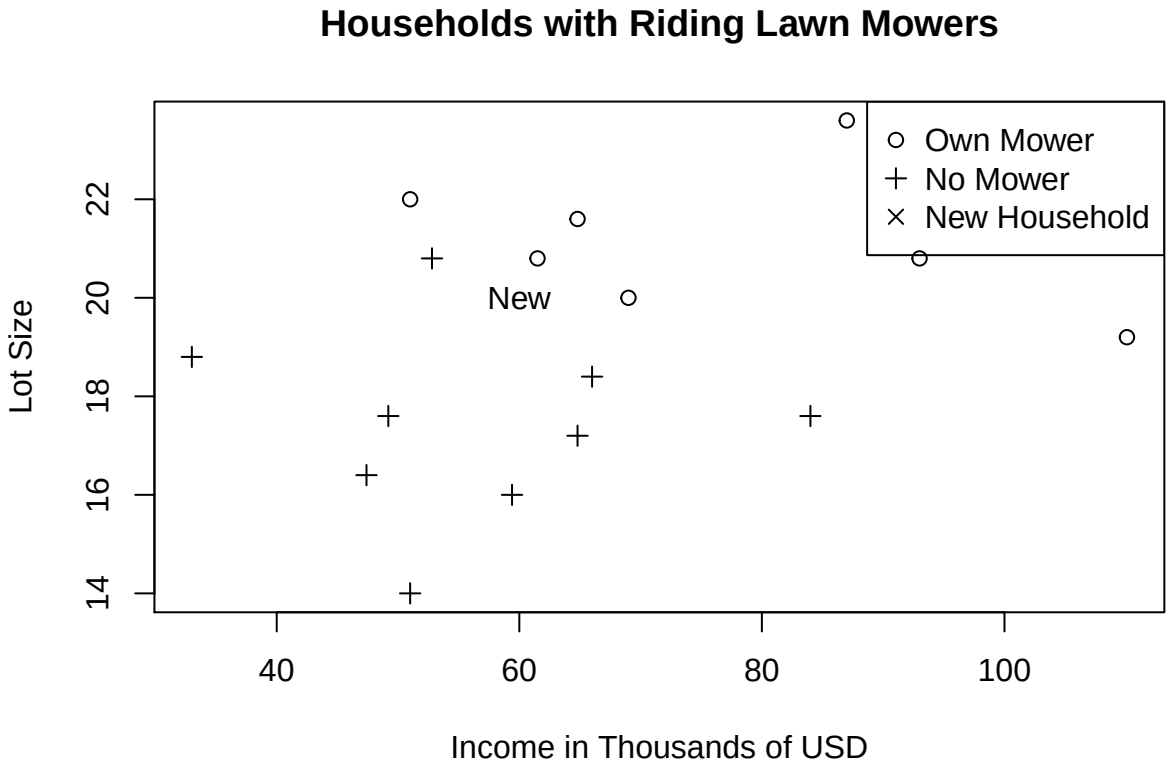
Scatterplot of Training Observations

Below is a scatterplot of which households own a riding lawnmower, indicated with circles, compared to whcih households do not own a riding lawnmower as indicated by the + sign.

Then our new household is indicated by the word "New" at the intersection of $60k in income and a 20k sq foot lot size.

Since we are preparing a KNN model, we are particularly curious about the points closest to our "New" household.

```
plot(Lot_Size~ Income, data=train.df, pch=ifelse(train.df$Ownership=='Owner', 1, 3), main= "Households w
#text(train.df$Income, train.df$Lot_Size, rownames(train.df), pos=4)
text(60,20, "New")
legend('topright', c("Own Mower", "No Mower", "New Household"), pch = c(1, 3, 4))
```



Normalizing Datasets

Variables with larger ranges (ie. Year or Salary) make similar homes further apart than if you were measuring variables with smaller ranges (ie. Number of Occupants)

```
#Create Backups of Original Data Frames
train.norm.df <- train.df
valid.norm.df <- valid.df
mower.norm.df <- mower.df

#Normalize the Income and Lot Size
norm.values <- preProcess(train.df[, 1:2], method=c("center", "scale"))
```

```
train.norm.df[, 1:2] <- predict(norm.values, train.df[, 1:2])
valid.norm.df[, 1:2] <- predict(norm.values, valid.df[, 1:2])
mower.norm.df[, 1:2] <- predict(norm.values, mower.df[, 1:2])
new.norm.df <- predict(norm.values, new.df)

#New Values are on smaller scale
summary(train.norm.df)
```

```
##      Income           Lot_Size         Ownership
##  Min.   :-1.6335   Min.   :-1.99660   Length:16
##  1st Qu.:-0.7218   1st Qu.:-0.61282   Class :character
##  Median :-0.1064   Median :-0.01977   Mode  :character
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.3799   3rd Qu.: 0.69189
##  Max.   : 2.2717   Max.   : 1.79892
```

Creating K-NN Model & Deciding K

For each instance of K we find the K nearest neighbors to the "New" point and then predict whether the household will own a riding lawn mower or not.

1-NN Classifier: Compares the position of "New" to the single nearest point, and predicts the new household will own a riding lawn mower. **3-NN Classifier:** Compares the position of "New" to 3 closest points, 2 of which are owners and 1 of which is not and predicts the new household will own a riding lawn mower.

Best Practices - If K is to low we might be overfitting the data. - If K is to high we miss the ability of K to capture the structure of data. - Generally K is between 1 and 20 - Generally K is odd to avoid a tie - Choose K that maximizes accuracy in the validation or testing dataset

```
#3-NN Model where K equals 3
Neighbors <- knn(train = train.norm.df[, 1:2], test = new.norm.df, cl = train.norm.df[, 3], k = 3)

print(row.names(train.df)[attr(Neighbors, "Neighbors.index")])
```

```
## character(0)
```

```
print(Neighbors)
```

```
## [1] Owner
## attr(,"nn.index")
##      [,1] [,2] [,3]
## [1,]    9   12    3
## attr(,"nn.dist")
##           [,1]     [,2]      [,3]
## [1,] 0.3252905 0.455852 0.4827363
## Levels: Owner
```

Measuring Accuracy

```r
library(caret)
library(class)
```

```
##
## Attaching package: 'class'
```

```
## The following objects are masked from 'package:FNN':
##
##     knn, knn.cv
```

```r
#Initaliza a dataframe with two columns
accuracy.df <- data.frame(k = seq(1, 16, 1), accuracy = rep(0, 16))

#Convert data type from character to factor
class(valid.norm.df$Ownership)
```

```
## [1] "character"
```

```r
valid.norm.df$Ownership <- as.factor(valid.norm.df$Ownership)
class(valid.norm.df$Ownership)
```

```
## [1] "factor"
```

```r
#compute knn for different k values on validation
for(i in 1:16){
  knn.pred <- knn(train.norm.df[,1:2], valid.norm.df[,1:2], cl= train.norm.df[,3], k = i)
  accuracy.df[i, 2] <- confusionMatrix(knn.pred, valid.norm.df$Ownership)$overall[1]
}

#Show the Accuracy of K-NN models for all possible values of K
accuracy.df
```

```
##     k accuracy
## 1   1    0.625
## 2   2    0.625
## 3   3    0.625
## 4   4    0.625
## 5   5    0.625
## 6   6    0.500
## 7   7    0.375
## 8   8    0.500
## 9   9    0.750
## 10 10    0.625
## 11 11    0.750
## 12 12    0.625
## 13 13    0.500
## 14 14    0.500
## 15 15    0.375
## 16 16    0.375
```

Classify the New Household

Based on our model, the new household is predicted to own a riding lawn mower.

```
knn.pred.new <- knn(mower.norm.df[,1:2], new.norm.df, cl = mower.norm.df[,3], k = 9)
row.names(train.df)[attr(Neighbors, "Neighbors.index")]
```

```
## character(0)
```

```
print(knn.pred.new)
```

```
## [1] Owner
## Levels: Nonowner Owner
```