K-NN Roxbury

Eduardo Ortiz

2023-10-08

Introduction:

Understanding whether homes have been remodeled or not is crucial for various stakeholders in the real estate market, including buyers, sellers, and investors. In this project, I aim to utilize a k-Nearest Neighbors (kNN) model to predict whether homes have undergone remodeling based on key features such as total price, number of rooms, and living area.

Purpose of the Project:

The primary objective of this project is to develop a predictive model that can accurately classify homes into two categories: remodeled and non-remodeled. By leveraging machine learning techniques, specifically the kNN algorithm, we seek to provide valuable insights into the housing market.

Variables Used in the Model:

1. Total Price: The total price of the home, which serves as a crucial indicator of its overall value and investment potential.
2. Number of Rooms: The number of rooms in the home, including bedrooms, bathrooms, and other living spaces, which can provide insights into the size and functionality of the property.
3. Living Area: The total living area of the home, measured in square feet or square meters, representing the amount of usable space available to occupants.

Approach:

To achieve the objectives, I will normalize the numerical features and encode categorical variables into factors. I will then split the data into training and testing sets, train the kNN model on the training data, and evaluate its performance on the testing data using metrics such as accuracy, precision, and recall.

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.3.1

## Loading required package: lattice
```

```r
library(tinytex)
```

```
## Warning: package 'tinytex' was built under R version 4.3.3
```

```r
# read dataset and inspect
housing.df <- read.csv("C:\\Users\\eddie\\Downloads\\Roxbury.csv")
head(housing.df)
```

```
##   TOTAL_VALUE LIVING_AREA ROOMS REMODEL
## 1       344.2        1352     6      NO
## 2       412.6        1976    10     YES
## 3       330.1        1371     8      NO
## 4       498.6        2608     9      NO
## 5       331.5        1438     7      NO
## 6       337.4        1060     6     YES
```

```r
str(housing.df)
```

```
## 'data.frame':    390 obs. of  4 variables:
##  $ TOTAL_VALUE: num  344 413 330 499 332 ...
##  $ LIVING_AREA: int  1352 1976 1371 2608 1438 1060 1916 1200 1092 2992 ...
##  $ ROOMS      : int  6 10 8 9 7 6 7 6 5 8 ...
##  $ REMODEL    : chr  "NO" "YES" "NO" "NO" ...
```

```r
# Split data into training and testing sets
set.seed(123) # Setting seed for reproducibility
train.index <- sample(nrow(housing.df), 0.6 * nrow(housing.df))
train.df <- housing.df[train.index, ]
test.df <- housing.df[-train.index, ]
```

```r
library(class)

# Create new data frames to preserve original data
train.norm.df <- train.df
test.norm.df <- test.df

# Standardize the data
norm.values <- preProcess(train.df[, 1:3], method = c("center", "scale"))
train.norm.df[, 1:3] <- predict(norm.values, train.df[, 1:3])
test.norm.df[, 1:3] <- predict(norm.values, test.df[, 1:3])

# Perform k-NN classification on arbitrary number to get baseline performance
knn<-knn(train=train.norm.df[,1:3], test=test.norm.df[,1:3], cl=train.norm.df[, 4], k=3)

# Calculate accuracy
accuracy <- mean(knn == test.norm.df[, 4])
accuracy
```

```
## [1] 0.7435897
```

```r
# Initialize a data frame to store accuracy values for different k
accuracy.df <- data.frame(k = 1:15, accuracy = rep(0, 15))

# Compute k-NN for different values of k
for (i in 1:15) {
  knn.pred <- knn(train.norm.df[, 1:3], test.norm.df[, 1:3], cl = train.norm.df[, 4], k = i)
  accuracy.df[i, "accuracy"] <- confusionMatrix(knn.pred, factor(test.norm.df[, 4]))$overall[1]
}

accuracy.df
```

```
##     k  accuracy
## 1   1 0.6858974
## 2   2 0.6923077
## 3   3 0.7435897
## 4   4 0.7051282
## 5   5 0.7692308
## 6   6 0.7500000
## 7   7 0.7628205
## 8   8 0.7435897
## 9   9 0.7884615
## 10 10 0.7820513
## 11 11 0.8012821
## 12 12 0.8012821
## 13 13 0.8141026
## 14 14 0.7948718
## 15 15 0.8076923
```

Results:

The analysis revealed that by establishing an initial baseline kNN model with a k value of 3, I obtained a baseline accuracy of 74%. This baseline accuracy served as a reference point for evaluating the effectiveness of the kNN model with different values of k.

A for loop was used to evaluate the model for values of k ranging from 1 to 15, and determined that the kNN model achieved the highest accuracy at a k value of 13, with an accuracy of 81%. This indicates that considering the 13 nearest neighbors for classification yielded the most accurate predictions for determining whether homes were remodeled or not.

These findings underscore the significance of parameter tuning in machine learning models, as the choice of hyperparameters such as the value of k can have a significant impact on model performance. By systematically evaluating the model with different values of k, I was able to identify the optimal configuration that maximized predictive accuracy.