**Homework 4 – Sadiya Amreen**

Problem 2 (15 points):

In this question we will use the Sacramento data, which covers available housing in the region of that city. The variables include numerical information about the size of the housing and its price, as well as categorical information like zip code (there are a large but limited number in the area), and the type of unit (condo vs house (coded as residential)). a. Load the data from the tidyverse library with the data(“Sacramento”) command and you should have a variable Sacramento. Because we have categoricals, convert them to dummy variables.

library(lattice)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(caret)

## Loading required package: ggplot2

library(tidyverse)

## ── Attaching packages  
## ───────────────────────────────────────  
## tidyverse 1.3.2 ──

## ✔ tibble 3.1.8 ✔ purrr 0.3.4  
## ✔ tidyr 1.2.1 ✔ stringr 1.4.1  
## ✔ readr 2.1.2 ✔ forcats 0.5.2  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ purrr::lift() masks caret::lift()

data("Sacramento")  
df<-Sacramento  
sac\_df <- data.frame(df)  
modify\_sac\_df<-select(sac\_df,-c(latitude,longitude,zip,city))  
head(modify\_sac\_df)

## beds baths sqft type price  
## 1 2 1 836 Residential 59222  
## 2 3 1 1167 Residential 68212  
## 3 2 1 796 Residential 68880  
## 4 2 1 852 Residential 69307  
## 5 2 1 797 Residential 81900  
## 6 3 1 1122 Condo 89921

dummy\_data <- dummyVars( ~ ., data = modify\_sac\_df)  
dummies <- as.data.frame(predict(dummy\_data, newdata = modify\_sac\_df))  
final\_sac\_d <- dummies  
final\_sac\_d$type <-dummies$type  
head(final\_sac\_d)

## beds baths sqft type.Condo type.Multi\_Family type.Residential price  
## 1 2 1 836 0 0 1 59222  
## 2 3 1 1167 0 0 1 68212  
## 3 2 1 796 0 0 1 68880  
## 4 2 1 852 0 0 1 69307  
## 5 2 1 797 0 0 1 81900  
## 6 3 1 1122 1 0 0 89921

2.B.With kNN, because of the high dimensionality, which might be a good choice for the distance function?

#To put them into the same range, high-dimension data should be harmonized.  
#The common and straightforward to use Euclidean distance measure can then be used.

2.C.Use kNN to classify this data with type as the label. Tune the choice of k plus the type of distance function. Report your results – what values for these parameters were tried, which were chosen, and how did they perform with accuracy?

library(kknn)

## Warning: package 'kknn' was built under R version 4.2.2

##   
## Attaching package: 'kknn'

## The following object is masked from 'package:caret':  
##   
## contr.dummy

set.seed(123)  
tc2c <- trainControl(method="cv", number = 10)  
tgknn <- expand.grid(kmax = 3:7,  
kernel = c("rectangular", "cos"),  
distance = 1:3)  
knn2c <- train(type ~ .,data = modify\_sac\_df, method = 'kknn',trControl = tc2c, preProcess = c('center', 'scale'),  
tuneGrid = tgknn)  
# Printing trained model provides report  
knn2c

## k-Nearest Neighbors   
##   
## 932 samples  
## 4 predictor  
## 3 classes: 'Condo', 'Multi\_Family', 'Residential'   
##   
## Pre-processing: centered (4), scaled (4)   
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 838, 838, 839, 839, 839, 838, ...   
## Resampling results across tuning parameters:  
##   
## kmax kernel distance Accuracy Kappa   
## 3 rectangular 1 0.9259417 0.3648830  
## 3 rectangular 2 0.9356080 0.4291293  
## 3 rectangular 3 0.9377471 0.4359111  
## 3 cos 1 0.9302199 0.4608879  
## 3 cos 2 0.9280694 0.4374066  
## 3 cos 3 0.9280577 0.4582539  
## 4 rectangular 1 0.9259417 0.3648830  
## 4 rectangular 2 0.9356080 0.4291293  
## 4 rectangular 3 0.9377471 0.4359111  
## 4 cos 1 0.9312835 0.4325855  
## 4 cos 2 0.9313069 0.4397001  
## 4 cos 3 0.9366718 0.4635936  
## 5 rectangular 1 0.9345215 0.3564663  
## 5 rectangular 2 0.9377359 0.4095237  
## 5 rectangular 3 0.9377359 0.4095237  
## 5 cos 1 0.9355965 0.4418362  
## 5 cos 2 0.9377816 0.4437250  
## 5 cos 3 0.9356311 0.4413530  
## 6 rectangular 1 0.9345215 0.3564663  
## 6 rectangular 2 0.9377359 0.4095237  
## 6 rectangular 3 0.9377359 0.4095237  
## 6 cos 1 0.9345213 0.4250141  
## 6 cos 2 0.9367064 0.4416205  
## 6 cos 3 0.9356311 0.4413530  
## 7 rectangular 1 0.9387994 0.3763733  
## 7 rectangular 2 0.9388226 0.3981448  
## 7 rectangular 3 0.9366604 0.4061491  
## 7 cos 1 0.9355965 0.4185840  
## 7 cos 2 0.9367064 0.4416205  
## 7 cos 3 0.9345673 0.4294551  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were kmax = 7, distance = 2 and kernel  
## = rectangular.