Homework 3

Charles Hanks

01/25/2023

Directions:

Please turn in a knitted HTML or pdf file on WISE.

1. Setup (1pt)

Change the author of this RMD file to be yourself and modify the below code so that you can successfully load the 'pinot.rds' data file from your own computer.

```r
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)
knitr::opts_knit$set(root.dir = "/Users/charleshanks/Desktop/MSDS/SPRING_23/ML")
setwd("/Users/charleshanks/Desktop/MSDS/SPRING_23/ML")
library(tidyverse)
library(caret)
library(fastDummies)
wine = read_rds("pinot.rds")
```

2. KNN Concepts (5pts)

Explain how the choice of K affects the quality of your prediction when using a K Nearest Neighbors algorithm.

Answer: A K that is too small can result in overfitting the model to the training data. The model would have low bias, but high variance. This means that the model would perform very well on the training data, but not be able to correctly predict outcomes from new data, and would be greatly affected by outliers. A sign of overfitting is when the accuracy of predictions from training data is higher than accuracy of predictions on testing data. If K is too large, the model will have high bias, low variance. In an instance of classification, the model will be more likely to choose the category with the most data points in the set (like California in the pinot dataset). The predictions of the model will be based on the majority category, and not an actual pattern in the data. This is underfitting the model. Depending the data, there will be a "goldilocks zone" for choice K - not too small, not too big, for producing the most accurate prediction from training and test data.

3. Feature Engineering (3pts)

1. Create a version of the year column that is a factor (instead of numeric)
2. Create dummy variables that indicate the presence of "cherry", "chocolate" and "earth" in the description, allowing for capital letters.

3. Create 3 new features that represent the interaction between time and the cherry, chocolate and earth inidicators
4. Remove the description column from the data

```
wine = wine %>% mutate(
          factor_year = as.factor(year),
          cherry = as.integer(str_detect(description,"[Cc]herry")),
          chocolate = as.integer(str_detect(description,"[Cc]hocolate")),
          earth = as.integer(str_detect(description,"[Ee]arth")),
          cherry_year = year*cherry,
          chocolate_year = year*chocolate,
          earth_year = year*earth) %>%
      select(-description)
```

4. Preprocessing (3pts)

1. Preprocess the dataframe that you created in the previous question using centering and scaling of the numeric features
2. Create dummy variables for the year factor column

```
#centering and scaling price and points
wine = wine %>% mutate(
  price_cs = (price - mean(price))/sd(price),
  points_cs = (points - mean(points))/sd(points))

#adding dummy cols for year factor column
dummies = wine %>%
  select(factor_year) %>%
  dummy_cols(remove_selected_columns = T)

wine =  wine %>% cbind(dummies)

#removing cols I don't need for model:
wine = wine %>%
  select(-id,-price,-points,-year,-factor_year)
```

5. Running KNN (5pts)

1. Split your data into an 80/20 training and test set
2. Use Caret to run a KNN model that uses your engineered features to predict province

- use 5-fold cross validated subsampling
- allow Caret to try 15 different values for K

3. Display the confusion matrix on the test data

```
set.seed(504)
pinot_index = createDataPartition(wine$province,p =0.8, list = FALSE)
p_train = wine[pinot_index, ]
p_test = wine[-pinot_index, ]
```

```r
control <- trainControl(method = "cv", number = 5)

#training my knn model
fit = train(province~.,
            data = p_train,
            method = "knn",
            tuneLength = 15,
            trControl = control)


#Confusion matrix
confusionMatrix(predict(fit,p_test),factor(p_test$province))
```

```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction          Burgundy California Casablanca_Valley Marlborough New_York
##   Burgundy               104         25                 0           4        1
##   California              59        666                11          20       12
##   Casablanca_Valley        0          0                 0           0        0
##   Marlborough              1          2                 2           2        1
##   New_York                 0          1                 1           1        1
##   Oregon                  74         97                12          18       11
##                     Reference
## Prediction          Oregon
##   Burgundy              31
##   California           240
##   Casablanca_Valley      0
##   Marlborough            1
##   New_York               3
##   Oregon               272
##
## Overall Statistics
##
##                 Accuracy : 0.6246
##                   95% CI : (0.6009, 0.6479)
##      No Information Rate : 0.4728
##      P-Value [Acc > NIR] : < 2.2e-16
##
##                    Kappa : 0.3809
##
##  Mcnemar’s Test P-Value : NA
##
## Statistics by Class:
##
##                       Class: Burgundy Class: California Class: Casablanca_Valley
## Sensitivity                   0.43697            0.8420                  0.00000
## Specificity                   0.95749            0.6122                  1.00000
## Pos Pred Value                0.63030            0.6607                      NaN
## Neg Pred Value                0.91114            0.8120                  0.98446
## Prevalence                    0.14226            0.4728                  0.01554
## Detection Rate                0.06216            0.3981                  0.00000
## Detection Prevalence          0.09863            0.6025                  0.00000
## Balanced Accuracy            0.69723           0.7271                  0.50000
##                      Class: Marlborough Class: New_York Class: Oregon
## Sensitivity                    0.044444       0.0384615        0.4973
## Specificity                    0.995700       0.9963570        0.8117
## Pos Pred Value                 0.222222       0.1428571        0.5620
## Neg Pred Value                 0.974159       0.9849940        0.7687
## Prevalence                     0.026898       0.0155409        0.3270
## Detection Rate                 0.001195       0.0005977        0.1626
## Detection Prevalence           0.005380       0.0041841        0.2893
## Balanced Accuracy              0.520072       0.5174093        0.6545
```

```
#Kappa = 0.38

#791 californian samples in ptest
#
p_test_sample_size = nrow(p_test)
p_test %>% filter(province == "California") %>% nrow() /1673
```

```
## [1] 0.4728033
```

6. Kappa (2pts)

Is this a good value of Kappa? Why or why not?

Answer: It's a fair value of Kappa (between .21 and .4). This value is more useful than the standard measure of accuracy (.62) in assessing the performance of this model because the data is imbalanced.

7. Improvement (2pts)

Looking at the confusion matrix, where do you see room for improvement in your predictions?

Answer: The model did not make any corrections predictions of province as Casablanca Valley, and only got 2 correct predictions for Marlborough and 1 for New York. As expected, it was most accurate in predicting California, which makes up 47% of the test set. Better features could be engineered to predict the 3 provinces with the least amount of observations in the train and test data.