stsci3740 final project modeling

Fiona Huang

2025-04-25

```
# import datasets
red <- read.csv("C:/Users/xinya/Downloads/Cornell Classes/STSCI 3740/final project/winequality-red.csv"

white <- read.csv("C:/Users/xinya/Downloads/Cornell Classes/STSCI 3740/final project/winequality-white.

wine <- read.csv("C:/Users/xinya/Downloads/Cornell Classes/STSCI 3740/final project/wine-quality-white-
```

Fitting a KNN Model

```
# normalize the data using z-score
normalize <- function(x) {
  return((x - mean(x)) / sd(x))
}

# wine_norm <- wine %>%
#   mutate(across(where(is.numeric) & !where(is.factor), normalize))

all_columns <- names(wine)
columns_to_normalize <- all_columns[all_columns != "quality" & sapply(wine, is.numeric)]
wine_norm <- wine
wine_norm[columns_to_normalize] <- lapply(wine[columns_to_normalize], normalize)



# change type of wine to white=1, red=2
wine_norm$type <- as.numeric(factor(wine_norm$type))

# split the dataset into train/test
set.seed(1)
index <- sample(1:nrow(wine_norm), size=nrow(wine_norm)*0.7, rep=FALSE)
training <- wine_norm[index, ]
testing <- wine_norm[-index, ]

training_X <- training %>% select(-quality)
testing_X <- testing %>% select(-quality)
```

```
# try different values of k
k.values <- 1:20
```

```r
knn.errors <- sapply(k.values, function(k) {
  knn.pred <- knn(training_X, testing_X, training$quality, k=k)
  mean(knn.pred != testing$quality)
})

print(knn.errors)
```

```
##  [1] 0.3989744 0.4794872 0.4707692 0.4574359 0.4482051 0.4456410 0.4507692
##  [8] 0.4528205 0.4451282 0.4528205 0.4364103 0.4420513 0.4317949 0.4379487
## [15] 0.4358974 0.4358974 0.4461538 0.4415385 0.4389744 0.4405128
```

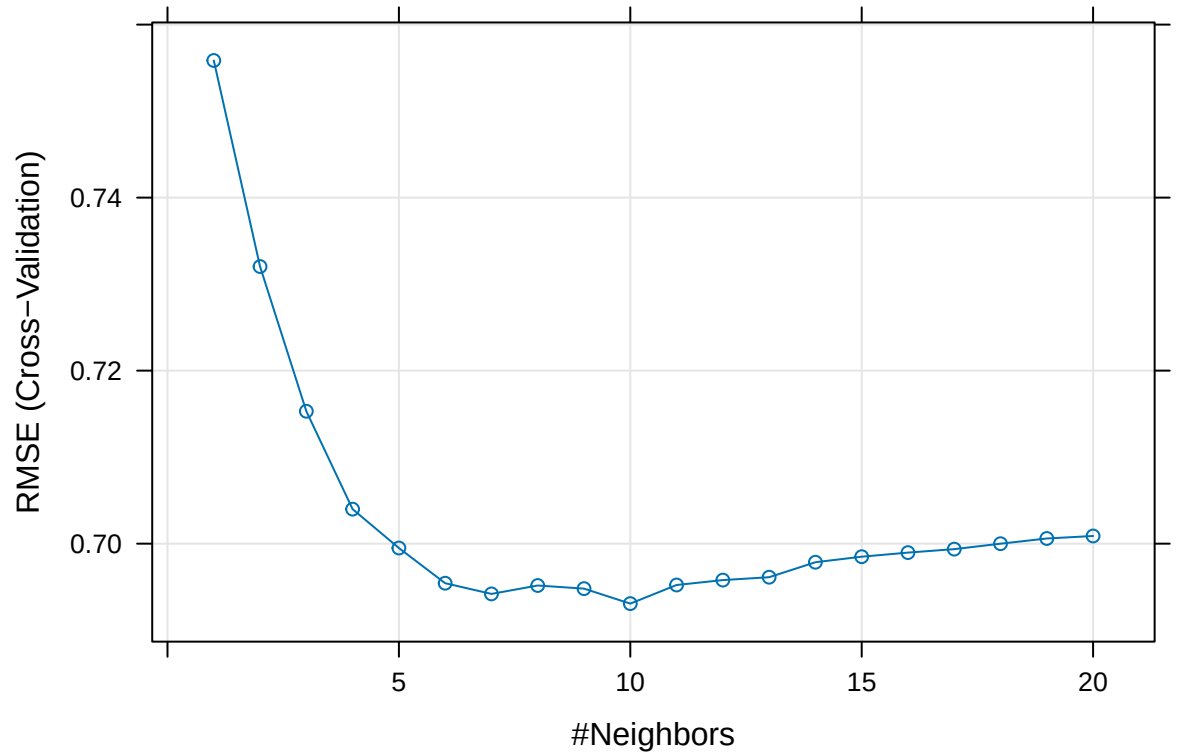
The value of k that seems to perform the best on this data is k=1.

```r
set.seed(1)
# 10-fold cross validation
control <- trainControl(method = "cv", number = 10)

knn_cv <- train(
  quality ~ .,
  data = wine_norm,
  method = "knn",
  trControl = control,
  tuneGrid = expand.grid(k = 1:20)
)

knn_cv
```

```
## k-Nearest Neighbors
##
## 6497 samples
##   12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5847, 5846, 5847, 5847, 5847, 5848, ...
## Resampling results across tuning parameters:
##
##   k   RMSE       Rsquared   MAE
##    1  0.7558513  0.3795189  0.4239693
##    2  0.7320306  0.3612061  0.4939238
##    3  0.7153067  0.3654421  0.5125768
##    4  0.7039897  0.3718375  0.5170525
##    5  0.6994976  0.3727748  0.5246780
##    6  0.6954288  0.3756623  0.5260905
##    7  0.6941926  0.3755005  0.5286038
##    8  0.6951634  0.3725416  0.5314023
##    9  0.6948018  0.3720174  0.5344289
##   10  0.6930709  0.3743660  0.5342596
##   11  0.6952228  0.3699285  0.5365115
##   12  0.6957883  0.3683444  0.5382253
##   13  0.6961228  0.3676329  0.5401778
##   14  0.6978546  0.3643808  0.5417053
```

```
##    15  0.6984920  0.3632550  0.5426139
##    16  0.6989681  0.3623560  0.5431027
##    17  0.6993684  0.3614560  0.5446430
##    18  0.6999960  0.3602518  0.5456382
##    19  0.7005973  0.3591466  0.5467203
##    20  0.7008890  0.3587130  0.5473792
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 10.
```

```r
plot(knn_cv)
```




```r
model <- knn(training_X, testing_X, training$quality, k=10)
```

```r
# fitting the knn models using less number of variables
X <- c("alcohol", "volatile.acidity", "residual.sugar", "total.sulfur.dioxide")
```