Comparison of Classification Methods for Celestial Objects

Lucas Ben

In this project, I used a variety of classification methods to classify celestial objects using their spectral properties. The data set is from the Sloan Digital Sky Survey. A formatted version of the data was used for this project.

```
set.seed(2002) # reproducibility
library(caret)
```

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

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

```
data = read.csv("stars_train.csv") # 1000x9
```

K-Nearest Neighbours

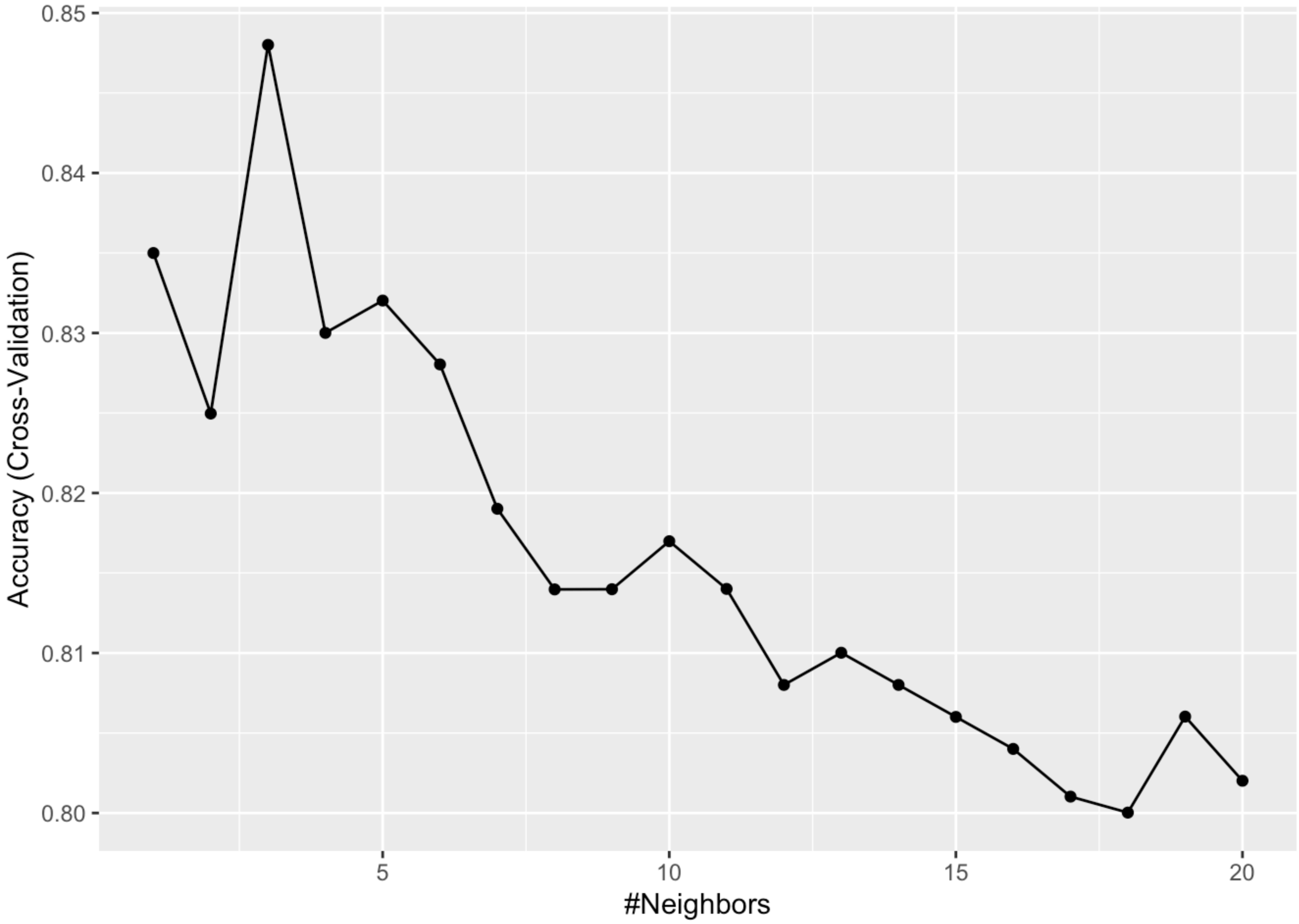
First, I will train a K-NN classifier with 10-fold CV using caret. The model will be trained to automatically select the number of neighbours k ∊ {1,2,3,…,20}.

```
formula = class ~ . # output variable is class, all other variables are used as input variables
ctrl = trainControl(method = "cv", number = 10) # 10-fold cross-validation
tune_knn = data.frame(k = 1:20) #tuning parameters
fit_knn = train(
  formula,
  data = data,
  method = "knn",
  preProc = c("center", "scale"),
  trControl = ctrl,
  tuneGrid = tune_knn
) # training K-NN classifier
```

Now I’ll plot the estimated accuracy of the classifier against the number of neighbours k. By visually inspecting this plot, the predictive accuracy is highest for k = 3 so this is the best value of the tuning parameter for this data set. You should choose whichever value of k leads to the highest accuracy.

For example, if I chose k = 15 the algorithm would underfit the data because for large k k-NN generates more linear classification boundaries. Underfitting leads to high bias and low predictive performance.

```
ggplot(fit_knn)
```



Linear & Quadratic Discriminant Analysis

Now I’ll train an LDA classifier and a QDA classifier. Before training a model, clearly describing each parameter is a necessary component of analysis.

LDA classifier has the following parameters:

Prior class probability π_k in each class k. This has length K =3 but π_3 = 1 - π_2 - π_1 therefore there are only two parameters.

Mean vector of inputs μ_k in each class k. Each vector has length 8 thus leading to 24 parameters (3x8 = 24).

Covariate matrix of inputs $\mathbf{\Sigma}$. This is a square matrix with 8 rows and 8 columns. Covariance matrices are symmetric thus there are 36 parameters (8x9/2 = 36).

```
Therefore there are 62 parameters to estimate.
```

QDA classifier has the following parameters:

Prior class probability π_k in each class k. This has length K =3 but π_3 = 1 - π_2 - π_1 therefore there are only two parameters (same as LDA).

Mean vector of inputs μ_k in each class k. Each vector has length 8 thus leading to 24 parameters (same as LDA).

Covariate matrix of inputs $\mathbf{\Sigma}$. QDA has one covariance matrix $\mathbf{\Sigma}_k$ for each class. Thus there are 108 parameters (3x36).

```
 Therefore there are 134 parameters to estimate.
```

```
fit_lda = train(
  formula,
  data = data,
  method = "lda",
  trControl = ctrl
) # training LDA classifier

fit_qda = train(
  formula,
  data = data,
  method = "qda",
  trControl = ctrl
) # training QDA classifier
```

Now I’ll calculate the predictive accuracy of the two classifiers.

```
accuracy_lda = fit_lda$results$Accuracy
cat("The mean predictive accuracy was approximately", round(accuracy_lda*100), "% for LD
A.")
```

```
## The mean predictive accuracy was approximately 82 % for LDA.
```

```
accuracy_qda = fit_qda$results$Accuracy
cat("\nThe mean predictive accuracy was approximately", round(accuracy_qda*100), "% for
QDA.")
```

```
##
## The mean predictive accuracy was approximately 94 % for QDA.
```

QDA performs significantly better which suggests that the LDA assumption (the classification boundaries are linear) is not satisfied.

Now I will make predictions given the spectral properties of three unidentified celestial bodies. According to LDA, the predicted classes for the three unidentified objects are galaxy (probability = 0.65), galaxy (probability = 0.88), and quasar (probability = 0.998). According to QDA, the predicted classes are star (probability = 0.99), star (probability = 0.97), and quasar (probability = 1).

```
newdata = read.csv("stars_new.csv") # unidentified celestial bodies

predict(fit_lda, newdata = newdata, type = "prob") # most likely class according to LDA
```

```
##        GALAXY          QSO         STAR
## 1 0.648074111 5.227091e-04 0.3514031798
## 2 0.884358972 1.072245e-05 0.1156303057
## 3 0.002207452 9.976584e-01 0.0001341257
```

```
predict(fit_qda, newdata = newdata, type = "prob") # most likely class according to QDA
```

```
##         GALAXY          QSO      STAR
## 1 6.077354e-03 1.538875e-09 0.9939226
## 2 2.973225e-02 2.937947e-07 0.9702675
## 3 2.010688e-20 1.000000e+00 0.0000000
```

Naive Bayes

Finally, I'll train a naive Bayes classifier and make predictions on the three unidentified celestial bodies from earlier. When the conditional distributions of input variables are assumed to be normal, the naive Bayes classifier assumes the conditional independence of the input variables while QDA estimates their dependence. The naive Bayes classifier with normal input distributions is a unique case of the QDA classifier where the covariance matrices are assumed to be diagonal.

The predicted classes are star (probability = 0.998), star (probability = 0.993), and quasar (probability = 0.999).

```
fit_nb = train(
  formula,
  data = data,
  method = "naive_bayes",
  trControl = ctrl
) # training naive Bayes classifier

predict(fit_nb, newdata = newdata, type = "prob") # most likely class according to naive
Bayes
```

```
##         GALAXY          QSO         STAR
## 1 2.026011e-04 1.731646e-21 9.997974e-01
## 2 6.707994e-04 2.022039e-13 9.993292e-01
## 3 1.958473e-06 9.999441e-01 5.390547e-05
```