# Asteroid_KNN

Annika Lin

2023-04-13

```
df <- read.csv("~/Documents/Georgetown/Spring23/Statistical Learning & Data Science/Proj
ect/NASA-asteroid-Classification-master/final/nasa.csv")
df <- df[ , !(names(df) %in% c("X"))]

df <- df[-695,]
```

```
library(caret)
```

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

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

```
# function to normalize data
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }

arr.norm <- apply(df[,-21], 2, normalize)

arr.norm <- data.frame(arr.norm, df$Hazardous)

colnames(arr.norm)[colnames(arr.norm) == "df.Hazardous"] ="Hazardous"
```

## (3.b) Fit kNN using 5-fold CV over a grid of values between 1 and 21 for the number of neighbors k,

using set.seed(1). How many neighbors are used in the final model?

```
# 5-fold CV to choose k

set.seed(1)

arr.norm$Hazardous <- as.factor(arr.norm$Hazardous)

fit.knn <- train(Hazardous ~ .,
  method = "knn",
  tuneGrid = expand.grid(k = 1:21),
  trControl = trainControl(method="cv", number=5, savePredictions = TRUE, classProbs = T
RUE),
  metric = "Accuracy",
  data = arr.norm)

fit.knn
```

```
## k-Nearest Neighbors
##
## 3078 samples
##   20 predictor
##    2 classes: 'False', 'True'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 2462, 2463, 2463, 2462, 2462
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.8645222  0.4227282
##    2  0.8534785  0.3734312
##    3  0.8804429  0.4254110
##    4  0.8749203  0.3926299
##    5  0.8814180  0.4071912
##    6  0.8778418  0.3926183
##    7  0.8801167  0.3847148
##    8  0.8797931  0.3796854
##    9  0.8775209  0.3589633
##   10  0.8804466  0.3685311
##   11  0.8801193  0.3552413
##   12  0.8804440  0.3575683
##   13  0.8797936  0.3410263
##   14  0.8797936  0.3371958
##   15  0.8801177  0.3317969
##   16  0.8762200  0.3033070
##   17  0.8788180  0.3181018
##   18  0.8781718  0.3167519
##   19  0.8794684  0.3145760
##   20  0.8768699  0.2957819
##   21  0.8791442  0.3117543
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

5 neighbors are used in the final model.

# (3.c) Which are the 10 most important variables using kNN? Is there any overlap with the variables you

selected using the lasso penalized logistic regression?

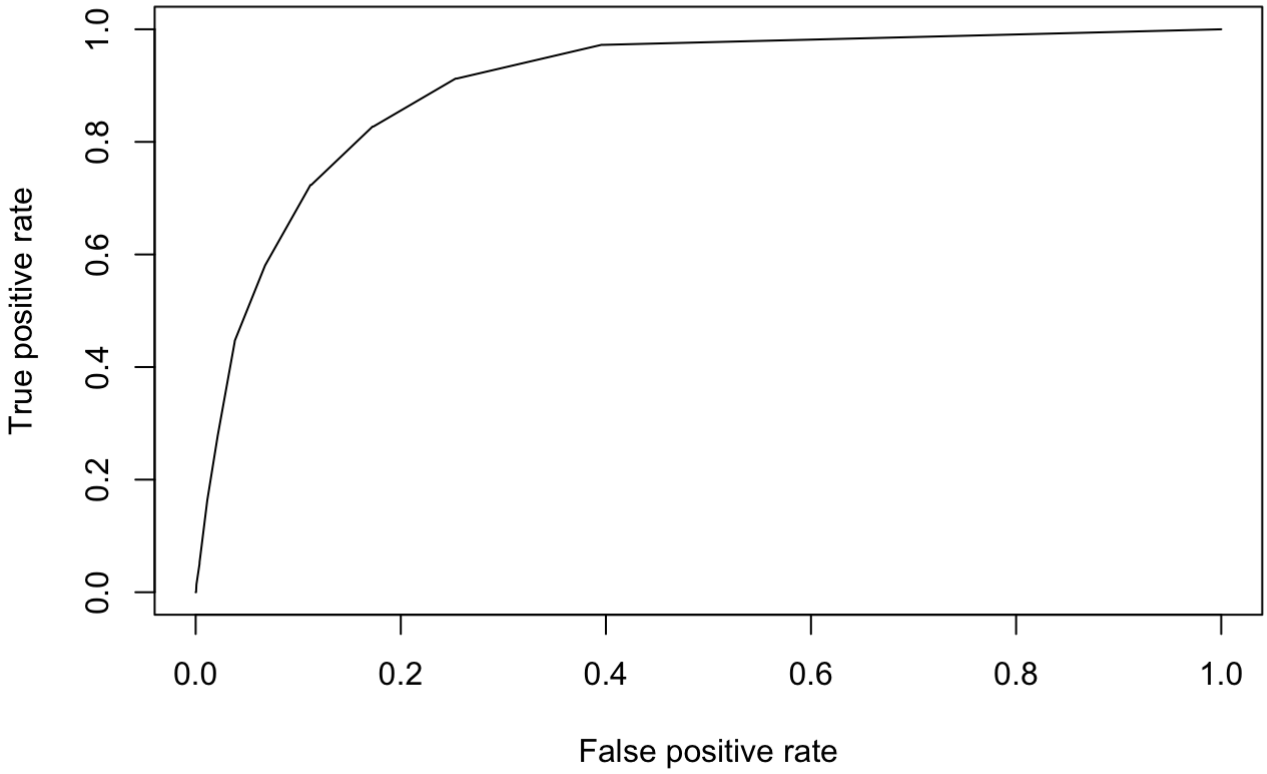
```
imp2 <- varImp(fit.knn)$importance
imp2 <- imp2[order(imp2$True, decreasing=T)[1:10],]
imp2
```

```
##                                    False      True
## Absolute.Magnitude              100.00000 100.00000
## Est.Dia.in.KM.min.              100.00000 100.00000
## Est.Dia.in.KM.max.              100.00000 100.00000
## Est.Dia.in.KM.range             100.00000 100.00000
## Orbit.Uncertainity               93.70653  93.70653
## Minimum.Orbit.Intersection       76.29926  76.29926
## Relative.Velocity.in.KM.per.sec  58.85243  58.85243
## Perihelion.Distance              58.33520  58.33520
## Eccentricity                     53.13038  53.13038
## Close.Approach.Date              26.30449  26.30449
```

```
# colnames(imp2)[1] <- "Importance"
# imp2[-2]
#
# intersect(rownames(imp2), sel.names)
```

# (3.d) Provide the cross-validated ROC curve and its AUC.

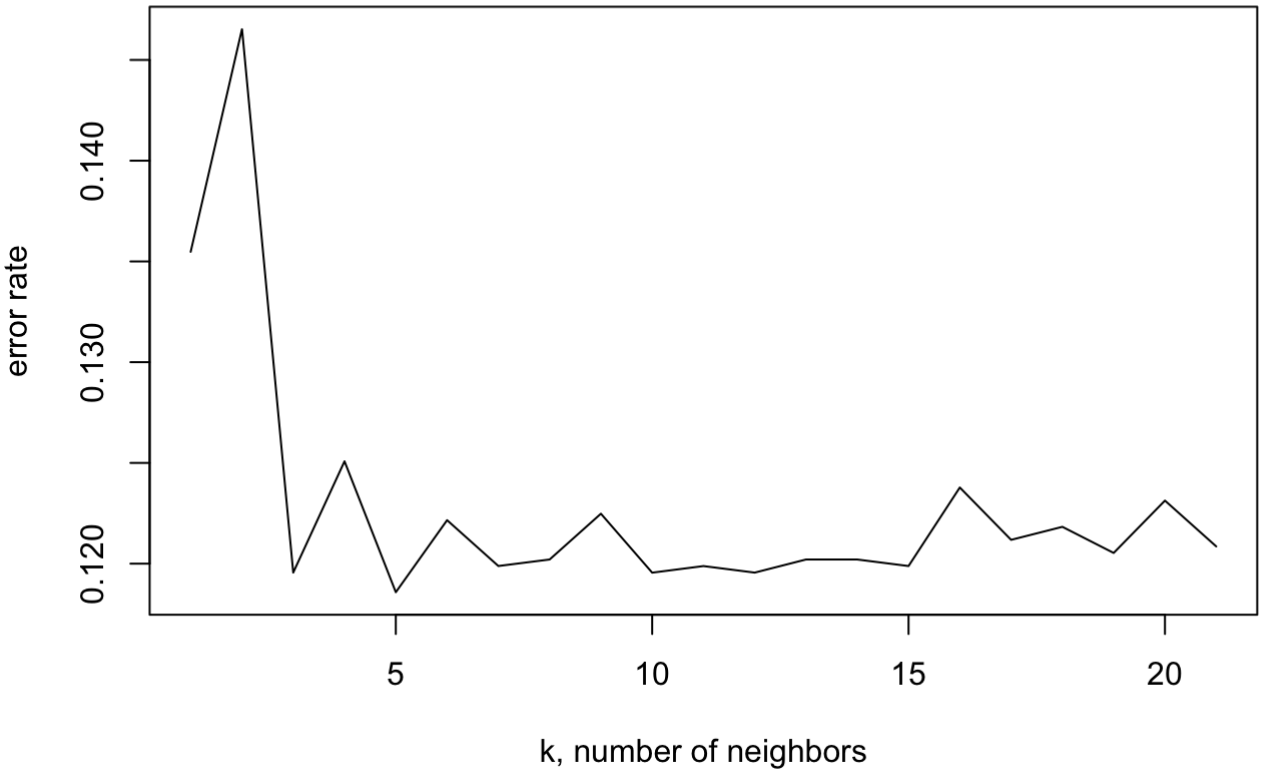
```
library(ROCR)
pihatcv.knn <- fit.knn$pred[fit.knn$pred$k == 13,]
pred <- prediction(pihatcv.knn$True, pihatcv.knn$obs)
perf <- performance(pred, "tpr", "fpr")
plot(perf)
```




```
# Area under ROC curve (AUC) = concordance index
auc.perf = performance(pred, "auc")
knn_auc <- auc.perf@y.values
knn_auc
```

```
## [[1]]
## [1] 0.900462
```

```
plot(fit.knn$results[,1], 1-fit.knn$results[,2], type="l",
xlab="k, number of neighbors", ylab="error rate")
```




```
#The predictions based on the selected k are:
pihat.fin <- predict(fit.knn, type="prob")
tail(pihat.fin)
```

```
##      False True
## 3073     1    0
## 3074     1    0
## 3075     1    0
## 3076     1    0
## 3077     1    0
## 3078     1    0
```

```
yhat.fin <- predict(fit.knn)
tail(yhat.fin)
```

```
## [1] False False False False False False
## Levels: False True
```

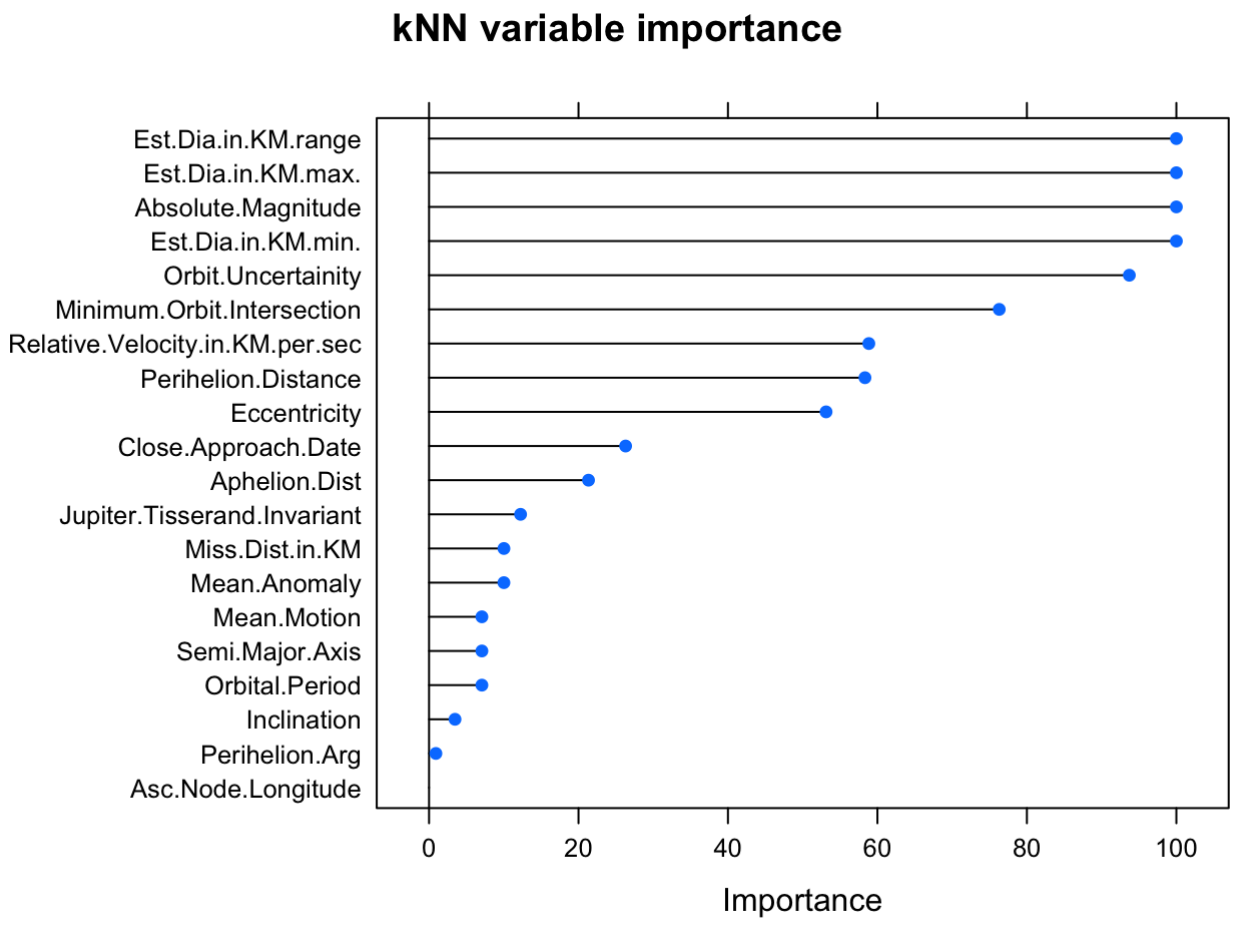
```
table(yhat.fin, df$Hazardous)
```

```
##
## yhat.fin False True
##    False  2591  197
##    True     53  237
```

```
tail(fit.knn$pred)
```

```
##        pred   obs     False       True rowIndex  k Resample
## 64633 False False 1.0000000 0.00000000     3057 21    Fold5
## 64634 False False 0.9047619 0.09523810     3061 21    Fold5
## 64635 False False 1.0000000 0.00000000     3062 21    Fold5
## 64636 False False 0.9523810 0.04761905     3069 21    Fold5
## 64637 False False 1.0000000 0.00000000     3077 21    Fold5
## 64638 False False 1.0000000 0.00000000     3078 21    Fold5
```

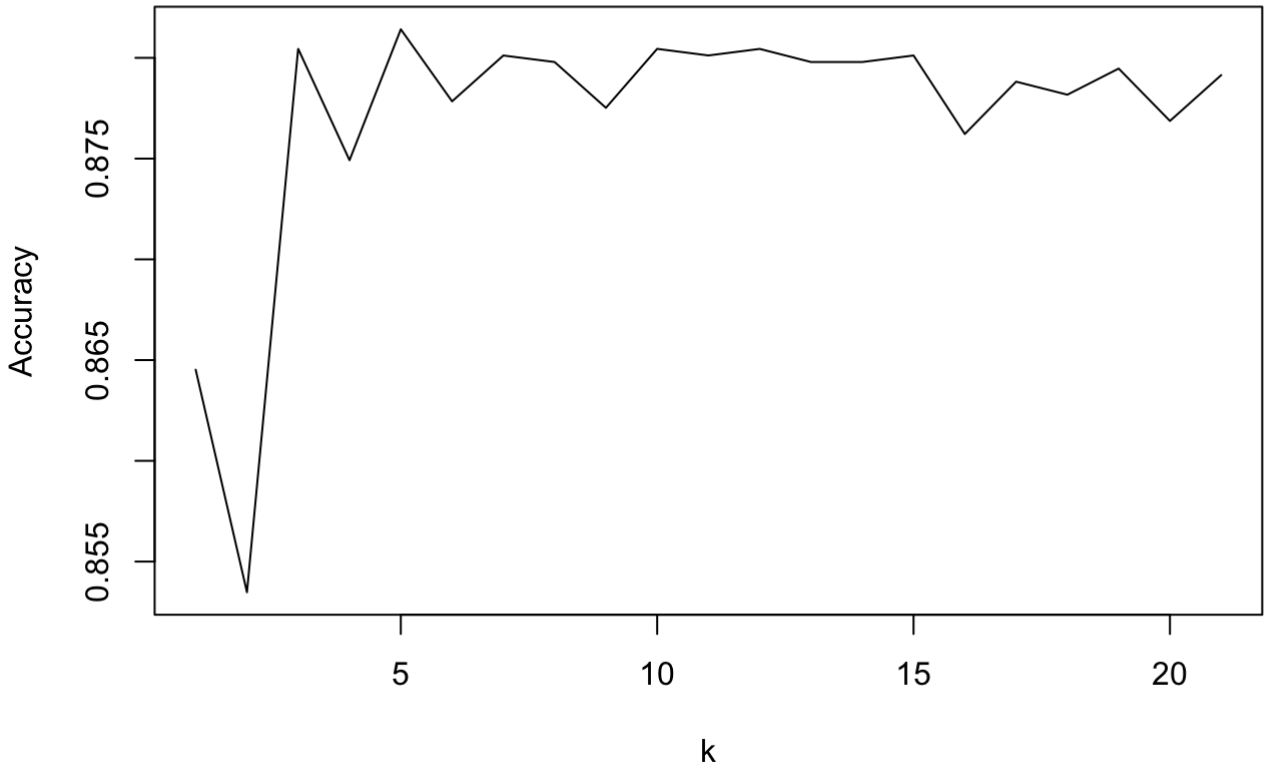
```
plot(varImp(fit.knn), main="kNN variable importance")
```




```
varImp(fit.knn)
```

```
## ROC curve variable importance
##
##                                 Importance
## Est.Dia.in.KM.range                100.0000
## Est.Dia.in.KM.max.                 100.0000
## Est.Dia.in.KM.min.                 100.0000
## Absolute.Magnitude                 100.0000
## Orbit.Uncertainity                  93.7065
## Minimum.Orbit.Intersection          76.2993
## Relative.Velocity.in.KM.per.sec     58.8524
## Perihelion.Distance                 58.3352
## Eccentricity                        53.1304
## Close.Approach.Date                 26.3045
## Aphelion.Dist                       21.3395
## Jupiter.Tisserand.Invariant         12.2465
## Miss.Dist.in.KM                     10.0205
## Mean.Anomaly                        10.0178
## Orbital.Period                       7.0710
## Mean.Motion                          7.0710
## Semi.Major.Axis                      7.0710
## Inclination                          3.4734
## Perihelion.Arg                       0.9156
## Asc.Node.Longitude                   0.0000
```

```
plot(fit.knn$results[,1:2], type="l")
```


Accuracy
0.855
0.865
0.875
5
10
15
20
k