Prediction of Exercise Manner

Duo Sun

12 August 2018

1 Introduction

This goal of this study is to predict the manner (noted as A, B, C, D, E in the dataset) in which 6 participants did the exercise.

The features used as predictors for this purpose include x-,y-,z-accelerometers on the belt, forearm, arm, and dumbell. Three models including LDA, CART, and kNN are built, and repeated cross-validation method is used to evaluate the performance and select the optimal parameters for each model.

Among the built models, kNN gave the best performanc with an average prediction accuracy of 89.6%. The performance of the kNN model is further evaluated on validation data, which gave a prediction accuracy of 91.5%.

2 Load data

The dataset "pml-training.csv" containing 19622 observations with 160 variables are used for training and validating models. Among these 159 variable, 12 features that are believed to be directly related with exercise manner including x-,y-,z-accelerometers on the belt, forearm, arm, and dumbell of 6 participants are selected to predict the manner of the exercise.

```
# Load libraries
library(caret)
```

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

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

```
library(ggplot2)
library(lattice)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
# Load dataset
pmldata <- read.csv("pml-training.csv",header = TRUE)

# Remove variables containing NA
pmldata <- pmldata[,colSums(is.na(pmldata))==0]

# Extract features including accelerometers x,y,z on the belt, forearm, arm, and dumbell of 6 participa
var <- c("accel_belt_x","accel_belt_y","accel_belt_z",
         "accel_arm_x","accel_arm_y","accel_arm_z",
         "accel_dumbbell_x","accel_dumbbell_y","accel_dumbbell_z",
         "accel_forearm_x","accel_forearm_y","accel_forearm_z",
         "classe")
dataset <- pmldata[,var]
```

3 Split the data into training/testing dataset and validation dataset

The dataset are split into two part: 80% dataset is used for training models and estimating their prediction accuracy; 20% dataset is used to evaluate the built model. The distributions of different manners in training dataset and validation dataset are shown below and are very well matched.

```
set.seed(0)
inTrainTest  <- createDataPartition(y=dataset$classe,p=0.80,list=FALSE)
validation   <- dataset[-inTrainTest,]
dataset      <- dataset[inTrainTest,]
dim(dataset)
```

```
## [1] 15699    13
```

```
cbind(freq=table(dataset$classe), percentage=prop.table(table(dataset$classe)) * 100)
```

```
##   freq percentage
## A 4464   28.43493
## B 3038   19.35155
## C 2738   17.44060
## D 2573   16.38958
## E 2886   18.38334
```

```
dim(validation)
```

```
## [1] 3923   13
```

```
cbind(freq=table(validation$classe), percentage=prop.table(table(validation$classe)) * 100)
```

```
##   freq percentage
## A 1116   28.44762
## B  759   19.34744
## C  684   17.43564
## D  643   16.39052
## E  721   18.37879
```

4 Data summary and visualization

The follow figures show range of each selected variable under different manner, and the correlation between variables, respectively.

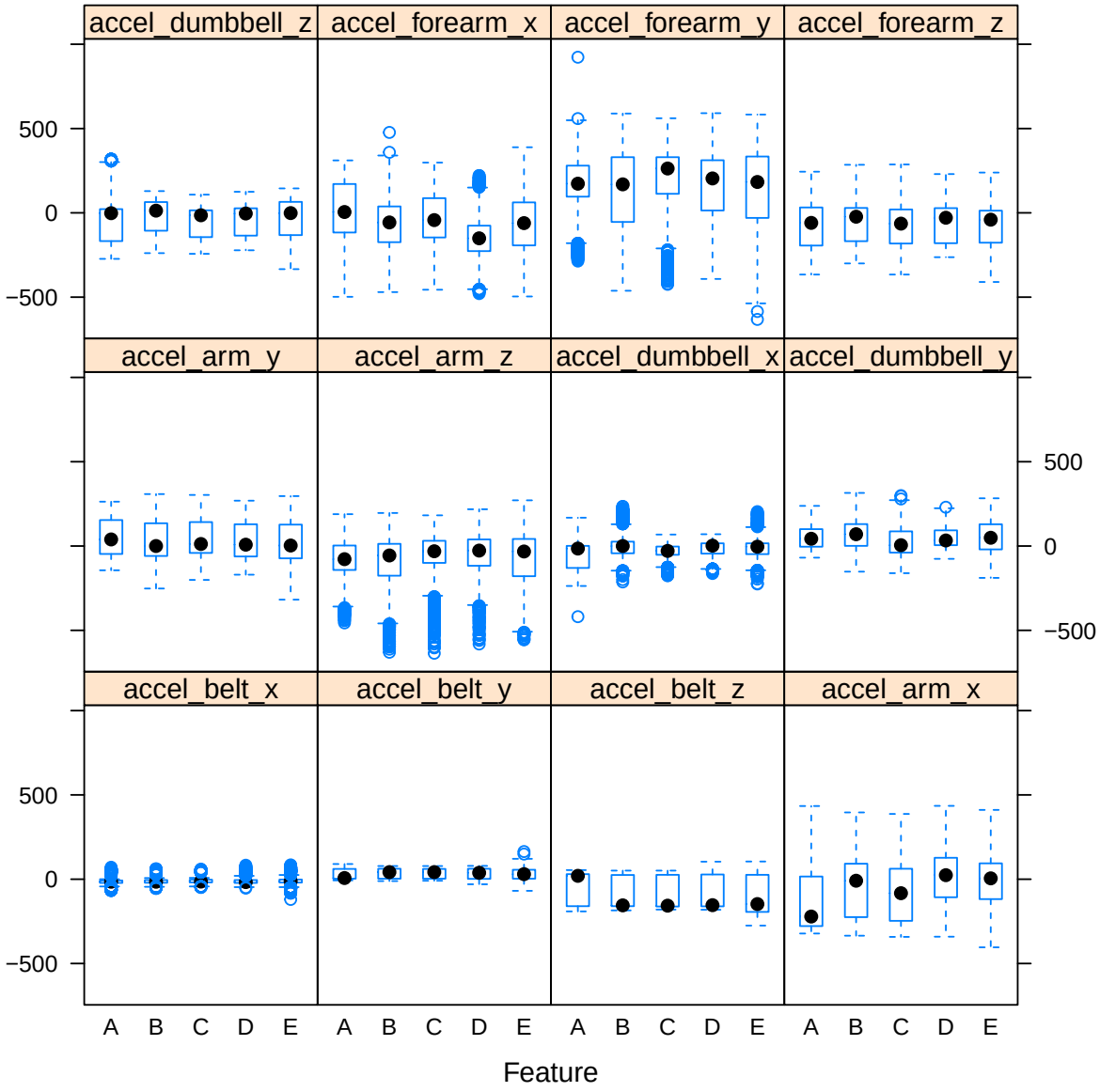
```
# Summarize data
dim(dataset)
```

```
## [1] 15699    13
```

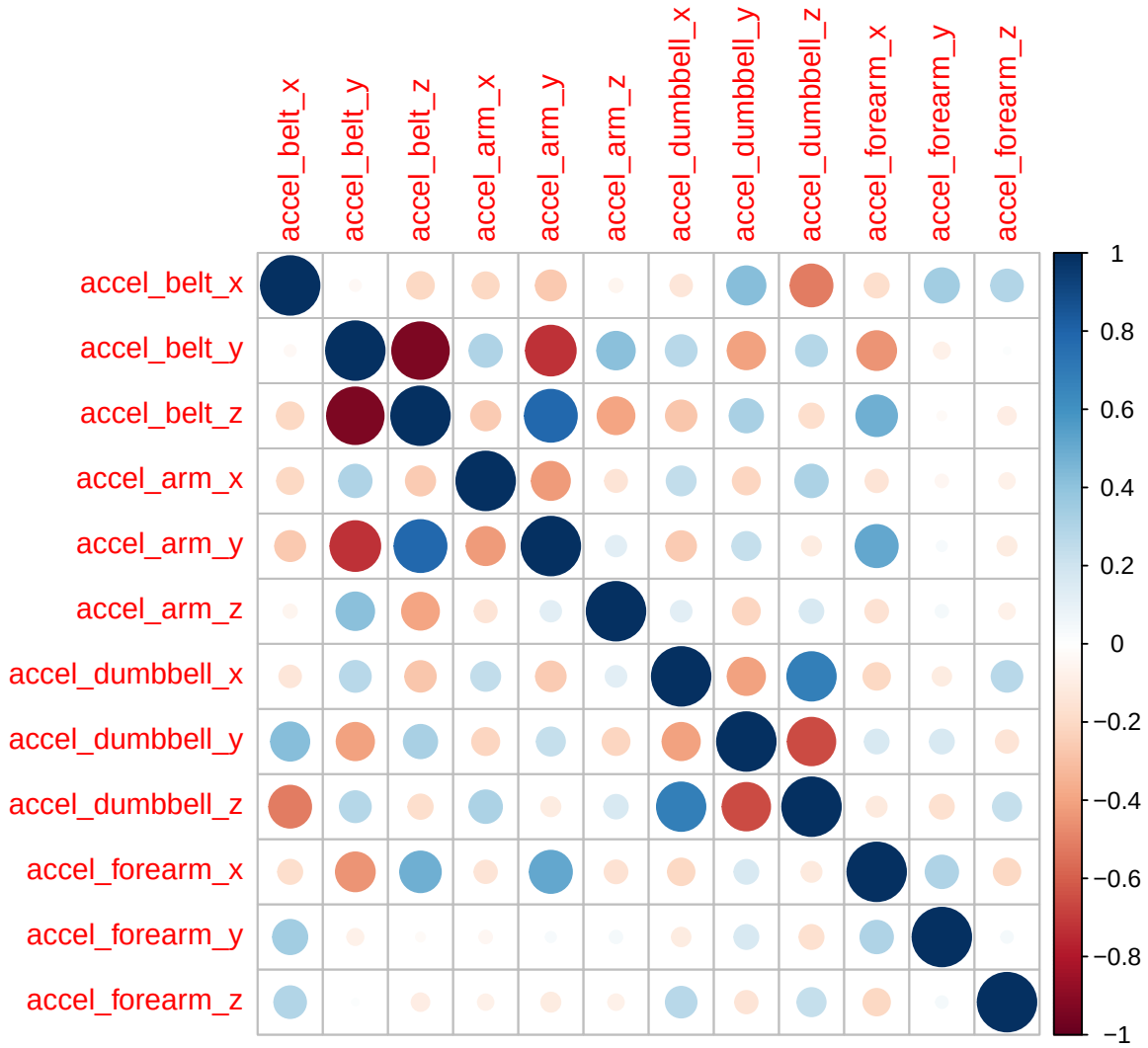
```
str(dataset)
```

```
## 'data.frame':    15699 obs. of  13 variables:
##  $ accel_belt_x    : int  -22 -20 -22 -21 -21 -22 -22 -20 -21 -21 ...
##  $ accel_belt_y    : int  4 5 3 2 4 3 4 2 4 2 ...
##  $ accel_belt_z    : int  22 23 21 24 21 21 21 24 22 23 ...
##  $ accel_arm_x     : int  -290 -289 -289 -289 -289 -289 -289 -288 -288 -290 ...
##  $ accel_arm_y     : int  110 110 111 111 111 111 111 109 110 110 ...
##  $ accel_arm_z     : int  -125 -126 -123 -123 -122 -125 -124 -122 -124 -123 ...
##  $ accel_dumbbell_x: int  -233 -232 -232 -233 -234 -232 -234 -232 -235 -233 ...
##  $ accel_dumbbell_y: int  47 46 48 48 48 47 46 47 48 47 ...
##  $ accel_dumbbell_z: int  -269 -270 -269 -270 -269 -270 -272 -269 -270 -269 ...
##  $ accel_forearm_x : int  192 196 189 189 193 195 193 193 190 193 ...
##  $ accel_forearm_y : int  203 204 206 206 203 205 205 204 205 205 ...
##  $ accel_forearm_z : int  -216 -213 -214 -214 -215 -215 -213 -214 -215 -214 ...
##  $ classe          : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Data visualization
featurePlot(x=dataset[,1:12], y=dataset[,13], plot="box")
```




```
# Calculate correlations between predictors
correlations <- cor(dataset[,1:12])
# Create correlation plot
corrplot(correlations, method="circle")
```



5 Model building

Three different models including LDA, CART, and kNN are built. Repeated cross-validation is used to evaluate the performance of models and select the optimal parameters for each model.

```
# Run algorithms using 10-fold cross validation, repeated 3 times
fitControl <- trainControl(method = "repeatedcv",number = 10,repeats = 3)
metric <- "Accuracy"
tuneLength <- 20
# LDA
set.seed(0);fit.lda  <- train(classe~., data=dataset, preProcess = c("center", "scale"), method="lda", r
# CART
set.seed(0);fit.cart <- train(classe~., data=dataset, preProcess = c("center", "scale"), method="rpart"
```

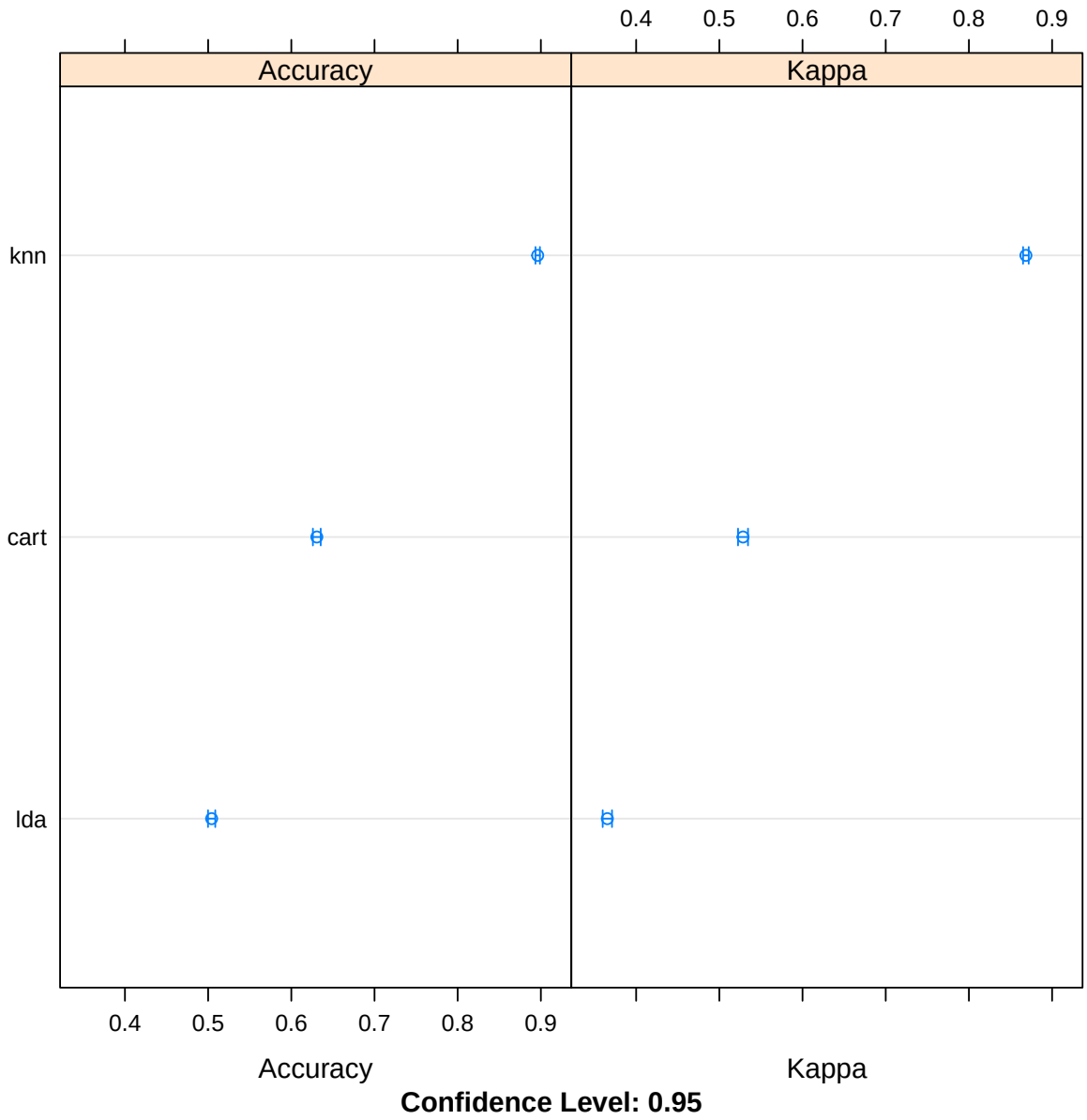
```r
# kNN
set.seed(0);fit.knn  <- train(classe~., data=dataset, preProcess = c("center", "scale"), method="knn", r
```

The cross-validation results show that kNN gives the best performace, an average accuracy of 89.6%.

```r
# Compare LDA, CART, and kNN
results <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, cart, knn
## Number of resamples: 30
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.4754621 0.4961808 0.5068448 0.5042155 0.5117872 0.5222930    0
## cart 0.6101911 0.6225722 0.6291808 0.6306528 0.6401393 0.6611465    0
## knn  0.8826531 0.8912768 0.8961125 0.8961714 0.8998090 0.9115213    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.3279636 0.3547642 0.3684229 0.3653663 0.3748799 0.3881079    0
## cart 0.5021500 0.5171881 0.5272838 0.5284496 0.5393940 0.5677434    0
## knn  0.8512546 0.8621681 0.8683352 0.8685006 0.8731306 0.8880339    0
```

```r
dotplot(results)
```

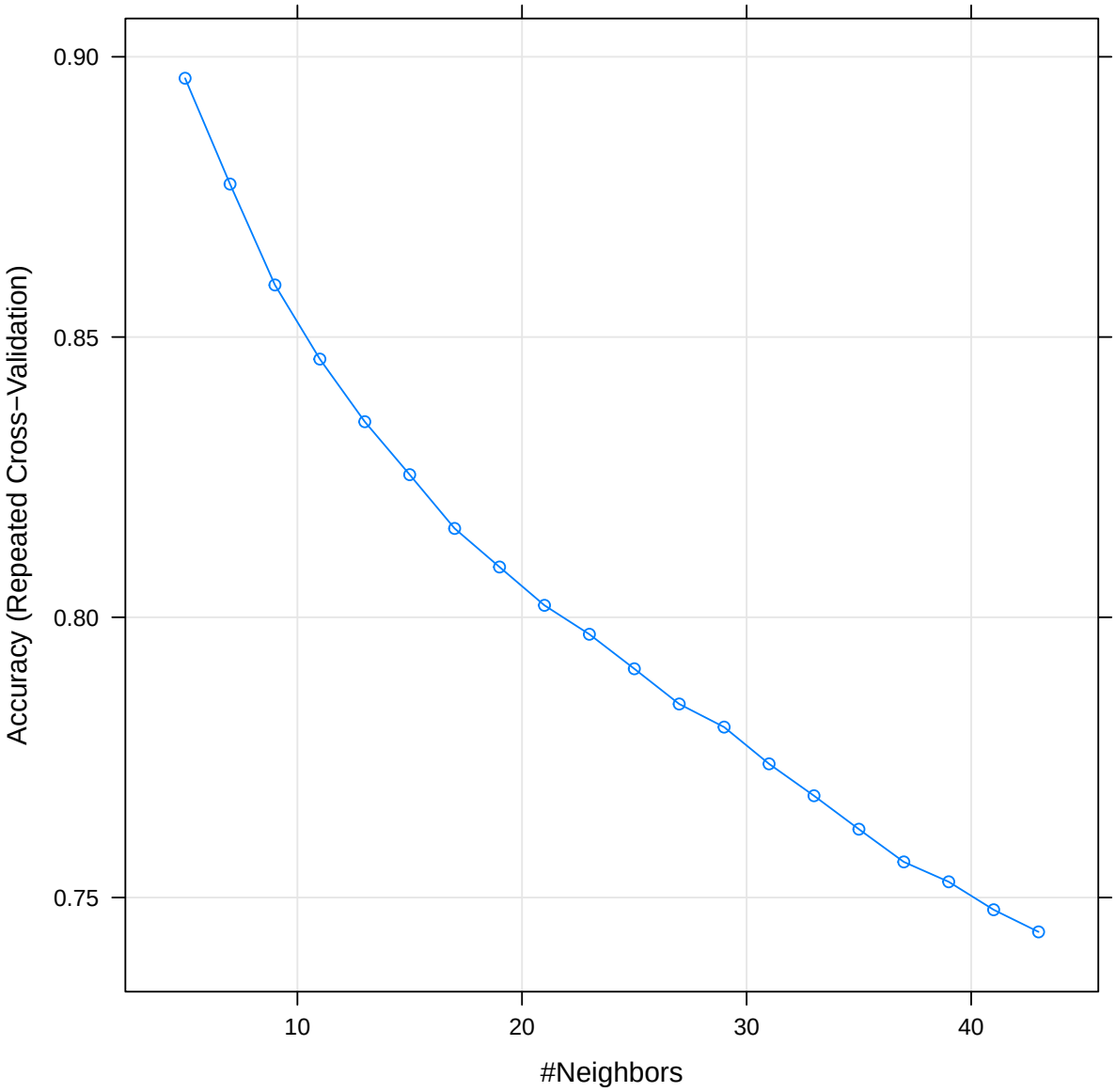



```
# Summarize Best Model
print(fit.knn)
```

```
## k-Nearest Neighbors
##
## 15699 samples
##    12 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (12), scaled (12)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 14128, 14130, 14129, 14128, 14130, 14130, ...
## Resampling results across tuning parameters:
```

```
##
##   k   Accuracy   Kappa
##    5  0.8961714  0.8685006
##    7  0.8772948  0.8445533
##    9  0.8592895  0.8216414
##   11  0.8460613  0.8048270
##   13  0.8348929  0.7906594
##   15  0.8254440  0.7786513
##   17  0.8158468  0.7664814
##   19  0.8089675  0.7577215
##   21  0.8021301  0.7490317
##   23  0.7969700  0.7424513
##   25  0.7908122  0.7345764
##   27  0.7845280  0.7265758
##   29  0.7804085  0.7213520
##   31  0.7738480  0.7130188
##   33  0.7681578  0.7057980
##   35  0.7621911  0.6982320
##   37  0.7563519  0.6908396
##   39  0.7528062  0.6863443
##   41  0.7478167  0.6800009
##   43  0.7438680  0.6749945
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
plot(fit.knn)
```



6 Model evaluation

The built kNN model is further evaluated on validation dataset, which shows a predication accuracy of 91.5%, close to the estimated result shown in section 5.

```
# Validate model accuracy using testing data
validation_pred <- predict(fit.knn, newdata = validation)
confusionMatrix(validation_pred, validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    A    B    C    D    E
##          A 1068   51   25   26    9
##          B    9  662   19    7   36
##          C   20   38  622   18   15
##          D   16    2   15  588   12
##          E    3    6    3    4  649
##
## Overall Statistics
##
##                Accuracy : 0.9149
##                  95% CI : (0.9057, 0.9234)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8921
##  Mcnemar's Test P-Value : 1.156e-12
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9570   0.8722   0.9094   0.9145   0.9001
## Specificity            0.9605   0.9776   0.9719   0.9863   0.9950
## Pos Pred Value         0.9059   0.9031   0.8724   0.9289   0.9759
## Neg Pred Value         0.9825   0.9696   0.9807   0.9833   0.9779
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2722   0.1687   0.1586   0.1499   0.1654
## Detection Prevalence   0.3005   0.1868   0.1817   0.1614   0.1695
## Balanced Accuracy      0.9587   0.9249   0.9406   0.9504   0.9476
```

```
mean(validation_pred == validation$classe)
```

```
## [1] 0.9148611
```

7 Predict new data with the model

The built kNN model is used to predict the new dataset "pml-testing.csv". The prediction result is given below.

```
newdata <- read.csv("pml-testing.csv",header = TRUE)
newdata <- newdata[,var[-13]]
newdata_pred <- predict(fit.knn, newdata = newdata)
newdata_pred
```

```
##  [1] B C C A A E D B A A A C B A E E A B B B
## Levels: A B C D E
```