Machine Learning Class Project

Kristy Wedel

December 4, 2016

Background

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). In this project, the goal was to develop a model using data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to determine which how the activity was performed.

Cleaning the data

```r
pml.training <- read.csv("C:/Users/kmw014/Desktop/Machine Learning Project/pml-training.csv")
pml.testing <- read.csv("C:/Users/kmw014/Desktop/Machine Learning Project/pml-testing.csv")
set.seed(1234)
#str(pml.training)
```

Some of the fields have a large percentage of NAs, and may not be the best predictors. They are removed for the creation of the model. Also, the index field and timestamps are not good predictors for this model.

```r
#removes fields that have high percentage of NA
pml.training <- pml.training[lapply(pml.training, function(x) sum(is.na(x)) / length(x) ) < 0.1 ]
#index
pml.training$X <- NULL
pml.training$raw_timestamp_part_1 <- NULL
pml.training$raw_timestamp_part_2 <- NULL
pml.training$user_name <- NULL
pml.training$num_window <- NULL
```

Fields that are highly correlated to others in the data set are not needed for the model. They are dropped below to limit the number of predictors. Remaining NAs are also removed.

```r
#make all numeric
drop = c("user_name","cvtd_timestamp", "new_window", "classe")
dat <- as.data.frame(sapply(pml.training[,!(names(pml.training) %in% drop)], as.numeric))
#remove highly correlated fields
list <- findCorrelation(cor(dat), cutoff=0.9, verbose=FALSE)
pml.training[list] <- NULL
#add classe back in
dat$classe <- pml.training$classe
pml.training <- dat
#remove NAs
pml.training <- na.omit(pml.training)
```

The random forest model ranks the importance of fields in the model. It can identify some of the more useful predictors.

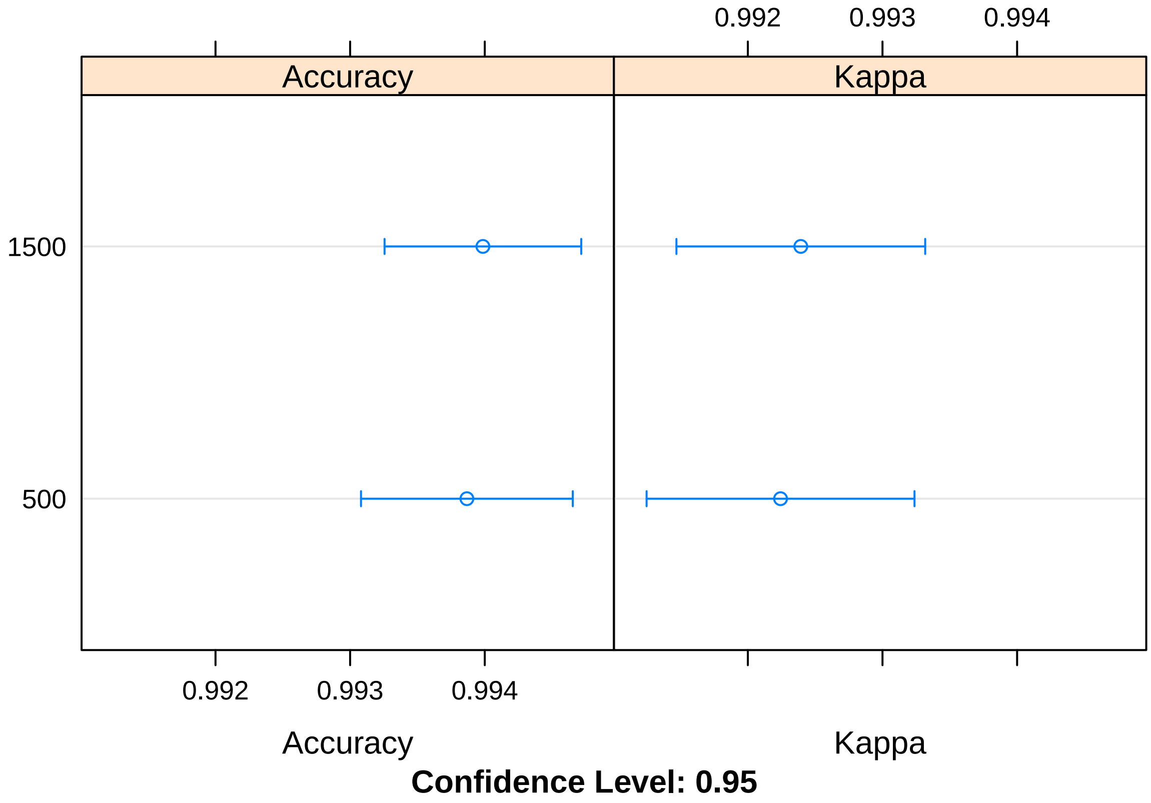
```
#remove lower importance fields

fit<-randomForest(classe~., data=pml.training, ntree=500, keep.forest=FALSE, importance = TRUE)
imp <- importance(fit)
impvar <- rownames(imp)[order(imp [,1], decreasing = TRUE)]
#limiting to the top 15
impvar <-head(impvar, 15)
pml.training<- pml.training[impvar]
pml.training$classe <- dat$classe
```

Through a comparison of the accuracy and kappas of each model, the best number of trees can be selected.

```
# Manual Search
x <- pml.training[,1:15]
y <- pml.training[,16]
metric <- "Accuracy"
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="grid")
tunegrid <- expand.grid(.mtry=c(sqrt(ncol(x))))
modellist <- list()
for (ntree in c(500, 1500)) {
    set.seed(1234)
    fit <- train(classe~., data=pml.training, method="rf", metric=metric, tuneGrid=tunegrid, trControl=c
    key <- toString(ntree)
    modellist[[key]] <- fit
}
# compare results
results <- resamples(modellist)
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: 500, 1500
## Number of resamples: 30
##
## Accuracy
##         Min.  1st Qu. Median   Mean 3rd Qu.   Max. NA's
## 500  0.9883  0.9926 0.9944 0.9939  0.9954 0.9969    0
## 1500 0.9888  0.9929 0.9946 0.9940  0.9954 0.9969    0
##
## Kappa
##         Min.  1st Qu. Median   Mean 3rd Qu.   Max. NA's
## 500  0.9852  0.9907 0.9929 0.9922  0.9942 0.9961    0
## 1500 0.9858  0.9910 0.9932 0.9924  0.9942 0.9961    0
```

```
dotplot(results)
```



Creating a validation set

The training set is split 75/25 to allow for a validation set of the model.

```
#split into data validation
inTrain <- createDataPartition(y=pml.training$classe, p=0.75, list = FALSE)
training <- pml.training[inTrain,]
pml.validation <- pml.training[-inTrain,]
pml.training <- training
```

Creating the model and testing on the validation set

The random forest model is created using the number of trees that gives the highest accuracy. It is then tested on the validation data. The accuracy is .9916395.

```
fit<-randomForest(classe~., data=pml.training, ntree=1500, importance = TRUE)
print(fit)
```

```
##
## Call:
```

```
##  randomForest(formula = classe ~ ., data = pml.training, ntree = 1500,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 1500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 0.79%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4180    4    0    0    1 0.001194743
## B   15 2805   24    4    0 0.015098315
## C    0   18 2537   12    0 0.011686794
## D    0    1   21 2388    2 0.009950249
## E    0    2    5    7 2692 0.005173688
```

```
pred_val_test<-predict(fit, pml.validation)
confusionMatrix(pred_val_test, pml.validation$classe)$overall[1]
```

```
##  Accuracy
## 0.9916395
```

Testing the model with the test set

First, the testing set is limited to include only the fields in the model, and then the model is tested to predict the class.

```
pml.training.labels<- colnames(pml.testing) %in% colnames(pml.training)
pml.testing<- pml.testing[pml.training.labels]

pred_test<-predict(fit, pml.testing)
pred_test
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

References

Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.