Practical machine learning project

MRIOTH

December 27, 2015

Loading packages:

```
library(e1071)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.2
```

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

```
library(ranger)
```

```
## Warning: package 'ranger' was built under R version 3.2.2
```

Data cleaning:

```
db <- as.data.frame(lapply(dat, function(x){
  x <- replace(x, x %in% c("n", "N", "","#DIV/0!"), NA)
  x <- as.factor(x)})) #normalizing missing or blank values
dat2 <- as.data.frame(lapply(db, as.numeric))#converting levels to numeric
dat2<- dat2[sample(nrow(dat2)),]#randomizing rows
```

Assessing correlated variables:

```
M<-abs(cor(dat2[,-159]))
diag(M)<-0
which(M>0.9,arr.ind=T)
```

```
##                 row col
## accel_belt_z      41   7
## accel_belt_x      39   8
## accel_belt_y      40  10
## accel_belt_z      41  10
## pitch_belt         8  39
## magnet_belt_x     42  39
## total_accel_belt  10  40
## accel_belt_z      41  40
## roll_belt          7  41
## total_accel_belt  10  41
## accel_belt_y      40  41
## accel_belt_x      39  42
## gyros_arm_y       60  59
## gyros_arm_x       59  60
```

Removing variables with less than 3% unique information:

```
dat3<-dat2[,(colSums(is.na(dat2))/nrow(dat2)) < 0.97]
```

Splitting data into 75% training 25% test sets:

```
inTrain<-createDataPartition(dat3$classe, p = 0.75)[[1]]
tra = dat3[ inTrain,]
test = dat3[-inTrain,]
```

Preprocessing training set with principle componenet analysis:

```
preProc<- preProcess(log10(tra[,-59]+1), method="pca")
trainPC<-predict(preProc, log10(tra[,-59]+1))
```

Building linear descriminant analysis model (LDA):

```
tra$classe<-as.factor(tra$classe)
modelf<-train(tra$classe~., method="lda", data=trainPC)
```

```
## Loading required package: MASS
```

```
testf<- predict(preProc, log10(test[,-59]+1))
```

LDA Accuracy using test data for out of sample error:

```
confusionMatrix(test$classe, predict(modelf,testf))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3   4   5
##          1 968 128 152 119  26
##          2 151 530 154  81  31
##          3 105  82 575  79  18
##          4  39  82 132 483  68
##          5  20 199 133 129 420
##
## Overall Statistics
##
##                Accuracy : 0.6069
##                  95% CI : (0.593, 0.6206)
##     No Information Rate : 0.2616
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5046
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            0.7545   0.5191   0.5017  0.54209  0.74600
## Specificity            0.8826   0.8926   0.9244  0.92001  0.88920
## Pos Pred Value         0.6949   0.5597   0.6694  0.60075  0.46615
## Neg Pred Value         0.9103   0.8759   0.8588  0.90049  0.96428
## Prevalence             0.2616   0.2082   0.2337  0.18169  0.11480
## Detection Rate         0.1974   0.1081   0.1173  0.09849  0.08564
## Detection Prevalence   0.2841   0.1931   0.1752  0.16395  0.18373
## Balanced Accuracy      0.8186   0.7059   0.7131  0.73105  0.81760
```

The out of sample error for LDA is 48.4%, pretty high. So we will need to try a different model.

Building a random forrest model (via Ranger):

```
rf<-ranger(classe~., tra, num.trees = 500, write.forest = TRUE, classification=TRUE)
```

```
## Growing trees.. Progress: 72%. Estimated remaining time: 11 seconds.
```

```
prd<- predict(rf, dat=test)
confusionMatrix(prd$predictions, as.factor(test$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    2    3    4    5
##          1 1393    0    0    0    0
##          2    0  947    1    0    0
##          3    0    0  856    0    0
##          4    0    0    2  804    2
##          5    0    0    0    0  899
##
## Overall Statistics
##
##                Accuracy : 0.999
##                  95% CI : (0.9976, 0.9997)
##     No Information Rate : 0.2841
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9987
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            1.0000   1.0000   0.9965   1.0000   0.9978
## Specificity            1.0000   0.9997   1.0000   0.9990   1.0000
## Pos Pred Value         1.0000   0.9989   1.0000   0.9950   1.0000
## Neg Pred Value         1.0000   1.0000   0.9993   1.0000   0.9995
## Prevalence             0.2841   0.1931   0.1752   0.1639   0.1837
## Detection Rate         0.2841   0.1931   0.1746   0.1639   0.1833
## Detection Prevalence   0.2841   0.1933   0.1746   0.1648   0.1833
## Balanced Accuracy      1.0000   0.9999   0.9983   0.9995   0.9989
```

Using the cross-validated testing set, the out of sample error for random forrest is less than 0.001–much better So we will apply this to the project's prediction set.

repeating the data cleaning for the 20-item prediction set:

Predicting the value using the random forrest classifier:

```
predict(rf, dat=fin)
```

```
## Ranger prediction
##
## Type:                             Classification
## Sample size:                      20
## Number of independent variables:  58
```