Practical Machine Learning - Course Project

Ignacio Ojea

INSTRUCTIONS

What you should submit

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Downloading and Normalizing Data

Let us start by loading the relevant libraries

```
library(caret); library(ggplot2); library(forecast); library(elasticnet)
```

Now I proceed to domwload and read the data and look at the relevant variable:

```
training <- read.csv("pml-training.csv", na.strings=c('NA','','#DIV/0!'))
testing <- read.csv("pml-testing.csv", na.strings=c('NA','','#DIV/0!'))
class(training$classe)
```

```
## [1] "factor"
```

```
summary(training$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Reading the basics about the information provided via http://groupware.les.inf.puc-rio.br/har, we get that the "classe" variable corresponds to:

- 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).

A glance at the data shows that there are 160 variables, and plenty of missing values (NAs and also #DIV/0!) which I coerced to NAs before). So it is reasonable to begin by considering only variables which have no NAs in them.

```
NAindex <- apply(training,2,function(x) {sum(is.na(x))})
training <- training[,which(NAindex == 0)]
NAindex <- apply(testing,2,function(x) {sum(is.na(x))})
testing <- testing[,which(NAindex == 0)]
table(complete.cases(training))
```

```
##
##  TRUE
## 19622
```

```r
table(complete.cases(testing))
```

```
##
## TRUE
##   20
```

Furthermore:

- The first column is not really a variable, it just contains the row number.
- The second is the user_name variable, which should not be relevant in our study.
- Third to seventh column correspond to the variables related to the time window for that particular sensor reading.

In a nutshell, we want to select only the variables that correspond to the sensors.

```r
sensorColumns <- grep(pattern = "_belt|_arm|_dumbbell|_forearm|classe", names(training))
training <- training[, sensorColumns]
testing <- testing[, sensorColumns]
dim(training)
```

```
## [1] 19622    53
```

So in fact we are left with all but the first seven columns. We also notice that all variables except classe are numeric.

```r
table(sapply(training[1,], class))
```

```
##
##  factor integer numeric
##       1      25      27
```

Data Splitting

We split the data with the usual 75% so that we can estimate the out of sample error of our predictor. Seed is set for reproducibility.

```r
set.seed(23122018)
raw.training <- training #rename to preserve the original data
inTrain <- createDataPartition(raw.training$classe, p=0.75, list=FALSE)
training <- raw.training[inTrain,]
crossvalidation <- raw.training[-inTrain,]
```

Model Construction and Selection

Model Construction

For the sake of exhaustion, I will be using using a random forest (“rf”), boosted trees (“gbm”), linear discriminant analysis (“lda”) models, linear support vector machines (“svmLinear”), and finally recursive partitioning for classification, regression and survival trees (“rpart”). This might be computationally demanding, but it shows knowledge (and was the suggested strategy in the final Quiz).

In a final addenda at the end I show that combining models by stacking predictions together using random forests (“rf”) does not present a significant improvement.

To reduce the risk of overfitting, a 10-fold cross validation is employed during model building.

The Kappa metric is selected as the comparison criteria.

```
mod.rf <- train(classe ~., method="rf", data=training, metric="Kappa", trControl=trainControl(method='c
mod.gbm <- train(classe~., data=training, method="gbm", metric="Kappa",trControl=trainControl(method='c
mod.lda <- train(classe~., data=training, method="lda", metric="Kappa",trControl=trainControl(method='c
mod.svmLinear <- train(classe~., data=training, method="svmLinear", metric="Kappa",trControl=trainContr
mod.rpart <- train(classe~., data=training, method="rpart", metric="Kappa",trControl=trainControl(metho
```

Now let us combine the models using the crossvalidation data set. Using the original testing set here would be methodologically incorrect. The predictions are stacked together using random forests.

Model Selection

The models are then compared using the resamples function from the Caret package.

```
library(lattice)
r.values <- resamples(list(rf=mod.rf,gbm=mod.gbm,lda=mod.lda,mod.rpart,mod.svmLinear))
summary(r.values)
```

```
##
## Call:
## summary.resamples(object = r.values)
##
## Models: rf, gbm, lda, Model4, Model5
## Number of resamples: 10
##
## Accuracy
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf     0.9904891 0.9918492 0.9928622 0.9930022 0.9943964 0.9959211    0
## gbm    0.9470468 0.9595580 0.9616168 0.9607980 0.9643044 0.9680923    0
## lda    0.6841033 0.6988791 0.7032267 0.7027450 0.7080218 0.7221467    0
## Model4 0.4813306 0.4917633 0.5042453 0.5006147 0.5111260 0.5118967    0
## Model5 0.7690217 0.7733835 0.7794018 0.7812228 0.7892377 0.7960571    0
##
## Kappa
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf     0.9879652 0.9896911 0.9909721 0.9911478 0.9929109 0.9948406    0
## gbm    0.9330135 0.9488432 0.9514512 0.9504055 0.9548334 0.9596174    0
## lda    0.6005490 0.6189099 0.6243686 0.6237255 0.6301638 0.6476692    0
## Model4 0.3217382 0.3353032 0.3538971 0.3475330 0.3610632 0.3622148    0
## Model5 0.7065232 0.7118995 0.7194314 0.7217535 0.7318425 0.7404078    0
```

Based on the display above, it can be determined that the Random Forest model outperforms the others by having a Kappa mean value of 0.991 and a mean accuracy of 0.993. I will therefore select it.

Accuracy on training set and cross validation set

```
rf.pred <- predict(mod.rf, training)
confusionMatrix(training$classe,rf.pred)$overall['Accuracy']
```

```
## Accuracy
##        1
```

Now let us do the same but with the crossvalidation set and without subsetting so that the corresponding statistics and error rates are shown.

```
rf.cross.pred <- predict(mod.rf, crossvalidation)
confusionMatrix(crossvalidation$classe,rf.cross.pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    0    0    0    0
##          B    4  944    0    1    0
##          C    0    5  849    1    0
##          D    0    0   10  794    0
##          E    0    1    0    2  898
##
## Overall Statistics
##
##                Accuracy : 0.9951
##                  95% CI : (0.9927, 0.9969)
##     No Information Rate : 0.2853
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9971   0.9937   0.9884   0.9950   1.0000
## Specificity            1.0000   0.9987   0.9985   0.9976   0.9993
## Pos Pred Value         1.0000   0.9947   0.9930   0.9876   0.9967
## Neg Pred Value         0.9989   0.9985   0.9975   0.9990   1.0000
## Prevalence             0.2853   0.1937   0.1752   0.1627   0.1831
## Detection Rate         0.2845   0.1925   0.1731   0.1619   0.1831
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9986   0.9962   0.9934   0.9963   0.9996
```

Not bad! We have an accuracy of 0.995 and a Kappa of 0.993. So the predictor seems to have a low out of sample error rate.

Results

Finally, we shall use the selected model to predict the classification of the testing set provided. In addition, in accordance to submission instructions.

```
final.prediction <- predict(mod.rf, testing)
print(as.data.frame(final.prediction))
```

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

Addenda: Combining models presents no significant improvement

Now let us combine the models using the crossvalidation data set. We will later compare the combined model with our Random Forest model using resampling and in the crossvalidation data set.

```
pred.rf <- predict(mod.rf, crossvalidation)
pred.gbm <- predict(mod.gbm, crossvalidation)
pred.lda <- predict(mod.lda, crossvalidation)
pred.rpart <- predict(mod.rpart,crossvalidation)
pred.svmLinear <- predict(mod.svmLinear,crossvalidation)
predDF <- data.frame(pred.rf, pred.gbm, pred.lda, pred.rpart, pred.svmLinear, classe = crossvalidation$
combMod <- train(classe ~ ., method = "rf", data = predDF,metric="Kappa",trControl=trainControl(method=
```

Once again, the two models are then compared using the resamples function from the Caret package.

```
library(lattice)
r.values <- resamples(list(rf=mod.rf,comb=combMod))
summary(r.values)
```

```
##
## Call:
## summary.resamples(object = r.values)
##
## Models: rf, comb
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf   0.9904891 0.9918492 0.9928622 0.9930022 0.9943964 0.9959211    0
## comb 0.9918200 0.9938776 0.9938900 0.9951074 0.9974511 1.0000000    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf   0.9879652 0.9896911 0.9909721 0.9911478 0.9929109 0.9948406    0
## comb 0.9896533 0.9922509 0.9922725 0.9938105 0.9967767 1.0000000    0
```

The difference between between the two models in median accuracy and in median Kappa is negligible.

```r
rf.cross.pred <- predict(mod.rf, crossvalidation)
combM.cross.prediction <- predict(combMod,crossvalidation)
confusionMatrix(crossvalidation$classe,rf.cross.pred)$overall['Accuracy']
```

```
## Accuracy
## 0.995106
```

```r
confusionMatrix(crossvalidation$classe,combM.cross.prediction)$overall['Accuracy']
```

```
##  Accuracy
## 0.9955139
```

Once again, this shows that combining models does not present a significant improvement.

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.

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz5aVvDq0zU