Practical Machine Learning - Prediction Assignment

Intro

The purpose of this project is to build a Machine Learning model for the PML data set. The data was provided in two sets a training set (about 20K records) and a testing set (20 records). The original data is taken from: 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#ixzz3yr1SiiZ0

Exploratory Analysis

The first thing I did was to look at the data in it's raw form. I've noticed that it has missing values marked either as NA or as "". I've read the data into r accordingly.

```
csvdata <- read.csv('pml-training.csv', stringsAsFactors = FALSE, na.strings=c("","NA"))
```

There are 160 variables in the input file and it seems that over a half are empty fields. This means that after cleanning we will get about 60 variables to use for estimation. I have two options - I can either analyze those 60 variables and choose the ones most usfull for a thin model or I can choose a model that takes all of them into considaration. I choose to use all of them in order to save time.

Cleanning the data

Since I want to use all the relevant variables for the analysis the first step would be to clean the data.

Missing data

I started by cleanning all empty columns. Some columns were almost empty (had over 19K NAs) Due to the large precentage of empty data I cleaned these as well. A quick way to get these column names would be:

```
colnames(summary(csvdata)[,complete.cases(summary(csvdata)[7,])])
```

Is this a time series?

An important question to answer since it determins the type of model and type of data you will need. On the first look it seems like this might be a time series since time stamps are available and the data is split into windows. Since the model will be used to test the original testing data (20 records) I looked at that data. Each record is a test on it's own, it's not a compliation of 20 different time series which means that every test record is only based on the sensors. I've decided not to treat this as a time series. This means I can also clean all the time related data.

Other

I've also cleaned the row numbers.

```
cleandata <- csvdata[,-c(12:36, 50:59, 69:83, 87:100)]
cleandata <- cleandata[,-c(37,39:48, 61:75,77:86)]
cleandata <- cleandata[,-c(3:7)]
cleandata <- cleandata[,-c(1)]
```

Factorizing

```
cleandata$user_name <- factor(cleandata$user_name)
cleandata$classe <- factor(cleandata$classe)
```

I've decided to keep the user_name since it might be a good predictor (I assume each user has it's unique way to do all of the 5 classes)

Model Selection

Now that the data is ready I began on choosing the model. The following points were important to choose the model.

- The predicted value is a class not a continues value.
- There are more than 2 classes.
- It is not a time series - each record is an independent test.

Relevant models would be classification models like classification trees, boosting and lda. I will train a few models and stack the results for the final model.

Initial Testing

At first I've tested rpart, rf, gbm and lda on 10% of the trainning data in order to get a quick result. Based on that test I've put aside the traditional rpart (output did not include all classes!) and lda (low accuracy).

Cross Validation

Since I've chosen Random Forset and Generalized Boosted Regression Model I don't need to do any extra work. The implementation already uses bootstrapping (resampling with replacments the trainning set).

Partiotioning

Since I've chosen methods that use bootstrapping I can actualy use the entire training. I will still chose only random 75% of it for the model trainning because this is big enough and can give me a very large initial test set.

```r
set.seed(1)
inTrain <- createDataPartition(y = cleandata$classe, p = 0.75, list = FALSE)
training <- cleandata[inTrain,]
testing <- cleandata[-inTrain,]
```

Random Forest

```r
set.seed(2)
modFit_rf <- train(classe ~ ., data = training, method = 'rf')
# See expected accuracy
print(modFit_rf$finalModel)
# See actual accuracy
rf_prediction  <- predict(modFit_rf,testing)
confusionMatrix(rf_prediction, testing$classe)
confusionMatrix(rf_prediction, testing$classe)$overall['Accuracy']
```

The out of bag(OOB) estimate of error rate is 0.6% which is very close to the accuracy we get when we test the data. This model already achieves an accuracy of 99.5%!

Generalized Boosted Regression Model (gbm)

```r
set.seed(3)
modFit_gbm <- train(classe ~ ., data = training, method = 'gbm', verbose = FALSE)
# See expected accuracy
print(modFit_gbm$finalModel)
# See actual accuracy
gbm_prediction  <- predict(modFit_gbm,testing)
confusionMatrix(gbm_prediction, testing$classe)
confusionMatrix(gbm_prediction, testing$classe)$overall['Accuracy']
```

This model achieves an accuracy of 96.4%.

Stacked model

I've taken the output of both models and trained them together for a stacked model in hope it will produce an even better model.

```r
StackedPredictions <- data.frame(rf_prediction, gbm_prediction, classe = testing$classe)
modFit_StackedModel <- train(classe ~ . , data = StackedPredictions, method = "rf")
StackedModel_prediction <- predict(modFit_StackedModel, StackedPredictions[,-c(3)])
confusionMatrix(StackedModel_prediction, testing$classe)
confusionMatrix(StackedModel_prediction, testing$classe)$overall['Accuracy']
```

Accuracy remained at 99.5%.

Predictions

Since the stacked model did not aid I will remain with the Random Forest model.

```
csvtestdata <- read.csv('pml-testing.csv', stringsAsFactors = FALSE, na.strings=c("",".","NA"))
cleantestdata <- csvtestdata[,-c(12:36, 50:59, 69:83, 87:100)]
cleantestdata <- cleantestdata[,-c(37,39:48, 61:75,77:86)]
cleantestdata <- cleantestdata[,-c(3:7)]
cleantestdata <- cleantestdata[,-c(1)]
# This time we also have an extra column to clean at the end of the data (problem ID)
cleantestdata <- cleantestdata[,-c(54)]
# Factorize username
cleantestdata$user_name <- factor(cleantestdata$user_name)
# Predict it on RF model
TestPrediction <- predict(modFit_rf, cleantestdata)
```

```
[1] B A B A A E D B A A B C B A E E A B B B
```

100% accuracy according to the quiz :)