ML Project

Summary:

- *Model: Random Forest from library* `randomForest`
- *Expected out of sample error: 0.008*
- *Number of correct predicted observations: 19/20*

Loading data

First, we start with loading basic libraries and the datasets:

```
library(caret); library(kernlab); library(ggplot2)

# Training-Data
fileURL_train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(fileURL_train, destfile="./train.csv",method="curl")
train<-read.csv("train.csv")

# Testing-Data
fileURL_test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(fileURL_test, destfile="./test.csv",method="curl")
testset<-read.csv("test.csv")

dim(train)
```

```
## [1] 19622   160
```

Data Splitting

We continue with building the training and test set. (I decided to split training and test set 50:50 to reduce the calculation time for the tree-model.):

```
set.seed(1974)
library(caret)
inTrain <- createDataPartition(y=train$classe,p=0.5, list=FALSE)
training <- train[inTrain,]; testing <- train[-inTrain, ]
```

Preprocessing

The Datasets consists of 160 variables. First of all, we should get rid of all variables with too many na's in the `training` dataset:

```
for(i in ncol(training):1){
    if (sum(is.na(training[i]))/length(training[,i])>0.9) {training[i]<-NULL}
}
```

Now we have 93 variables left.

The first 7 variables obviously seems to have no predictive value. So we get rid of them, too:

```
training <- training[,-(1:7)]
```

A quick look at the structure of the dataset (`str(training)`) shows that some variables are of type factor. So we convert all variables but `training$classe` to numeric types:

```
for (i in ncol(training)-1:1){training[,i]<-as.numeric(training[,i])}
```

The dataset still contains of a lot of variables which doesn't help to build a predictive algorithm. Clearly speaking, let's get rid of all variables with near zero variation:

```
nzv <- nearZeroVar(training[,-ncol(training)-1],saveMetrics = TRUE)
    subtrain <- training[,!as.logical(nzv$nzv)]
```

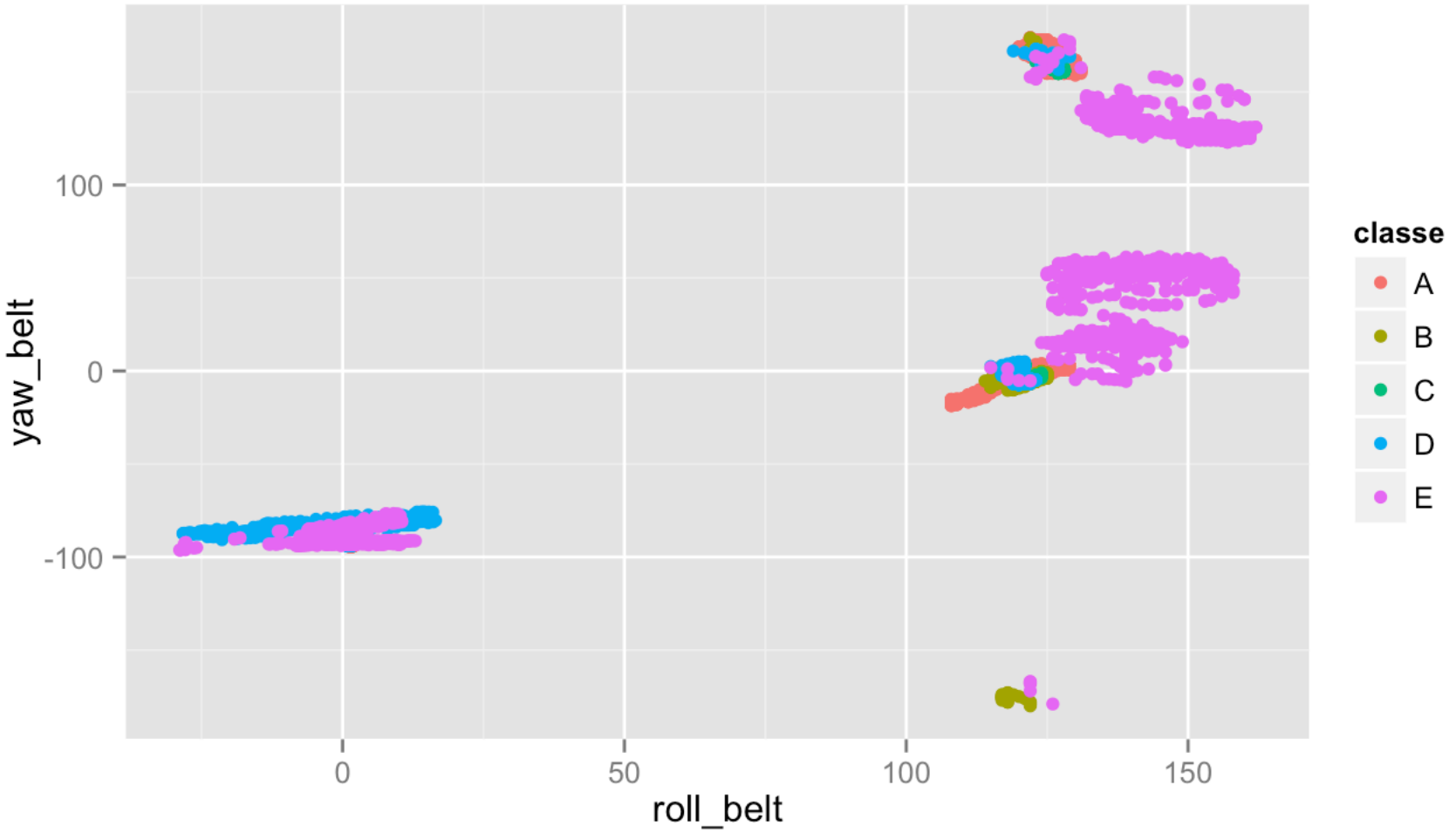
Some Plotting

A deeper look at the variables showed that 4 predictors, namely

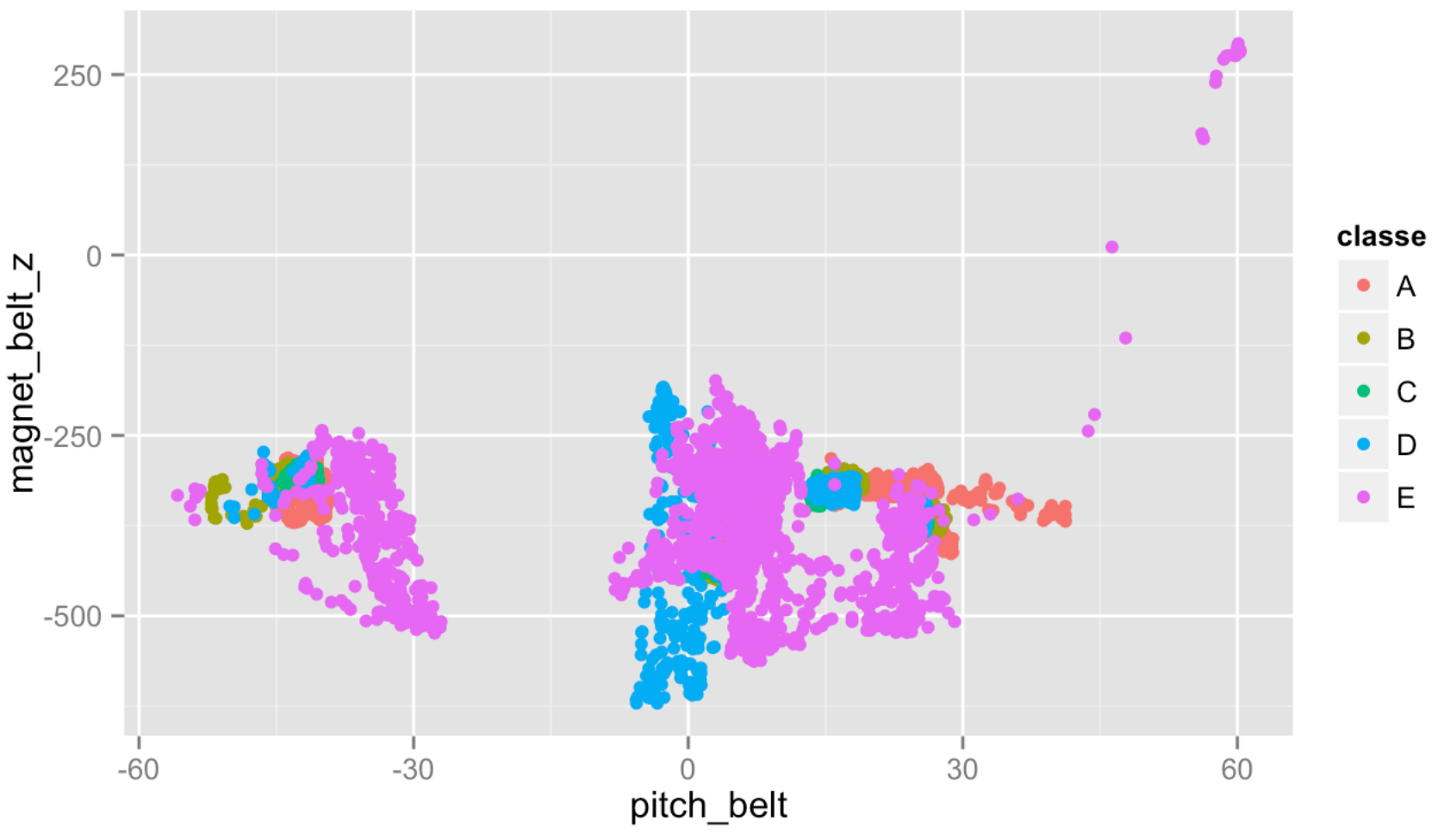
- `roll_belt+yaw_belt`
- `pitch_belt` , and
- `magnet_belt_z`

seems to have the strongest predictive influence. Let's plot these variables:

```
qplot(roll_belt,yaw_belt, color=classe, data=subtrain)
```




```
qplot(pitch_belt, magnet_belt_z, color=classe, data=subtrain)
```



Prediction algorithm (too slow)

Let's continue with training. I chose the **random forest** alghorithm as this - as mentioned in the lecture - is *"… usually one of the two top performing algorithms along with boosting in prediction contests".* ^(see Lecture)^

At first, I used the alghorithm from the `caret`-package. The results were quite good and I actually succeeded in predicting all 20 test-observations. Anyway, the drawback of this algorithm was that it took overnight to calculate.

Accidentally, I deleted all variables in the workspace. And I finished this document on the last day prior to the submission deadline. So, I needed to switch to the `randomForest()`-function from the `randomForest`-library. (And it appeared to be that this function is much faster but a little bit less accurate.).

Anyway, here are the results:

```
library(randomForest)
fit.rf2 = randomForest(classe ~ ., data=subtrain)
fit.rf2
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = subtrain)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 1.04%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 2783    6    0    0    1    0.002509
## B   18 1873    8    0    0    0.013691
## C    0   16 1689    6    0    0.012858
## D    1    0   35 1571    1    0.023010
## E    0    0    3    7 1794    0.005543
```

We can see, that the error rate is **1.04%**.

Cross Validation

We now use the test-set to check the model:

```
prediction <- predict(fit.rf2,testing)
CM <- confusionMatrix(testing$classe, prediction)
CM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2787    3    0    0    0
##          B    7 1880    9    2    0
##          C    0   24 1685    2    0
##          D    0    0   20 1584    4
##          E    0    0    4    0 1799
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.99, 0.994)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.99
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.997    0.986    0.981    0.997    0.998
## Specificity             1.000    0.998    0.997    0.997    1.000
## Pos Pred Value          0.999    0.991    0.985    0.985    0.998
## Neg Pred Value          0.999    0.997    0.996    1.000    1.000
## Prevalence              0.285    0.194    0.175    0.162    0.184
## Detection Rate          0.284    0.192    0.172    0.161    0.183
## Detection Prevalence    0.284    0.193    0.174    0.164    0.184
## Balanced Accuracy       0.999    0.992    0.989    0.997    0.999
```

The estimated out of sample error is: 1-Accuracy, e.g.: **0.008**.

Prediction for Write-up Assignment

```
fileURL_test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(fileURL_test, destfile="./test.csv",method="curl")
testset<-read.csv("test.csv")
predict(fit.rf2,testset)
```

```
##  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  D  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

Out of the 20 oberservations 19 were correctly predicted, e.g.: **95% accuracy**!