Course 8 project

R Markdown

The goal of your project is to predict the manner in which people did the exercise. the data is provided by: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har

data load and preprocessing

below I load the data. I see that there are a lot of NA's within variables.

data load and preprocessing

there are still 60 variables which is a lot. To see if this could be reduced I perform a test for zero covariates.

```
##    freqRatio      percentUnique        zeroVar            nzv
##  Min.   : 1.000   Min.   :  0.01019   Mode :logical   Mode :logical
##  1st Qu.: 1.022   1st Qu.:  1.25497   FALSE:60        FALSE:59
##  Median : 1.069   Median :  3.36102                   TRUE :1
##  Mean   : 6.192   Mean   : 11.08467
##  3rd Qu.: 1.135   3rd Qu.:  7.96937
##  Max.   :87.256   Max.   :100.00000
```

apply on test set

these changes are aso performed with the test dataset

create training and validation set

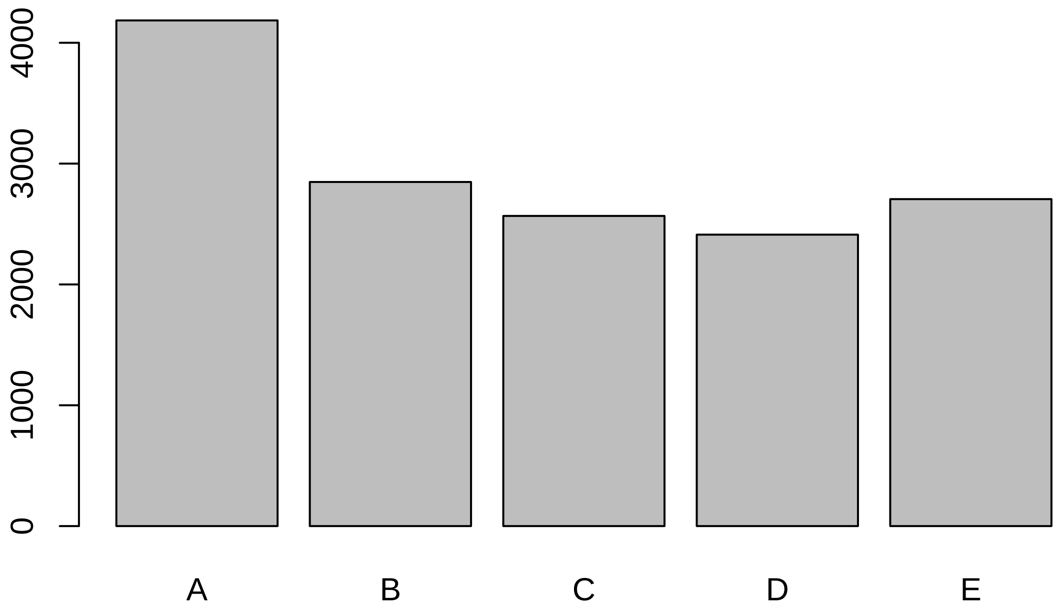
the dataset is seperated into 2 dataset, the training dataset is used to train different models. The validation dataset is used to estimate how accurate the different models are for out sample errors.

```
set.seed(22)
inTrainIndex <- createDataPartition(df_train$classe, p=0.75)[[1]]
df_training <- df_train[inTrainIndex,]
df_validation <- df_train[-inTrainIndex,]
```

exploration

before perfroming Machine learning models, I do some explorative analysis to get to know the data

```
library(ggplot2)
df_training$classe <- as.factor(df_training$classe)
plot(df_training$classe)
```



most of the datapoints are assigned to classe A. but it seems no problem for further analysis because there are no rare events.

train machine learning models

to start i perform a " normal " decision tree.

```
decisiontree <- train(classe ~., data= df_training, method="rpart")
decisiontree
```

```
## CART
##
## 14718 samples
##    58 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14718, 14718, 14718, 14718, 14718, 14718, ...
## Resampling results across tuning parameters:
##
##   cp         Accuracy   Kappa
##   0.2437102  0.7654753  0.7022441
##   0.2569069  0.5796089  0.4633778
##   0.2703883  0.3840259  0.1706814
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.2437102.
```

```
dt_predict <- predict(decisiontree, newdata = df_validation)
df_validation$classe <- as.factor(df_validation$classe)
cf_matrix <- confusionMatrix(df_validation$classe, dt_predict)
cf_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    0    0    0    0
##          B    0  949    0    0    0
##          C    0    0    0    0  855
##          D    0    0    0    0  804
##          E    0    0    0    0  901
##
## Overall Statistics
##
##                Accuracy : 0.6617
##                  95% CI : (0.6483, 0.6749)
##     No Information Rate : 0.522
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5694
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000       NA       NA   0.3520
## Specificity            1.0000   1.0000   0.8257   0.8361   1.0000
## Pos Pred Value         1.0000   1.0000       NA       NA   1.0000
## Neg Pred Value         1.0000   1.0000       NA       NA   0.5856
## Prevalence             0.2845   0.1935   0.0000   0.0000   0.5220
## Detection Rate         0.2845   0.1935   0.0000   0.0000   0.1837
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      1.0000   1.0000       NA       NA   0.6760
```

The accurancy of the final model on the training set is 76.5% which is not that good the accurancy of the final mode on the validation set is 66 %%

model selection

There is no cross validation performed due to high compution time.

apply model to test set

apply model to the test set to make predicitons

```
test_predict <- predict(decisiontree, newdata= df_test)
```