Machine Learning project

Ale Goity

15/3/2021

#Summary We will analyzed the data using decision trees. I generated two models, the first using all the variables and a second model with only some variables. (I tried to performed a random forest analysis but my computer was not able to complete the prediction). Finally, I predicted the classe of the testing data using the model with better performance. Additionally, I compared the difference between the results obtain with the two models

Read and cleaning the data

```
setwd("/Users/alegoity/Dropbox/Cursos/Coursera/R/Course8 (Machine Learning)/project/")
training<-read.csv("pml-training.csv")
testing<-read.csv("pml-testing.csv")
```

Remove columns with no information for the analysis. No variation in the column or NA values. Also, the information from columns 1 to 5. Doing this, we reduced variables to 54.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.2
## Loading required package: lattice
## Loading required package: ggplot2
```

```
a<-nearZeroVar(training)
training<-training[,-a] #eliminate columns with no variation
training<-training[ , -(1:5)]
training<- training[,colSums(is.na(training))<nrow(training)-1000] #to eliminate columns with high numb

dim(training)
```

```
## [1] 19622    54
```

Use createDataPartition to generate training_data (70%) and testing_data (30%) group

```
set.seed(222)
inTrain<-createDataPartition(training$classe, p=0.7, list=FALSE)
training_data <- training[inTrain,]
testing_data <- training[-inTrain,]
```

Modelling

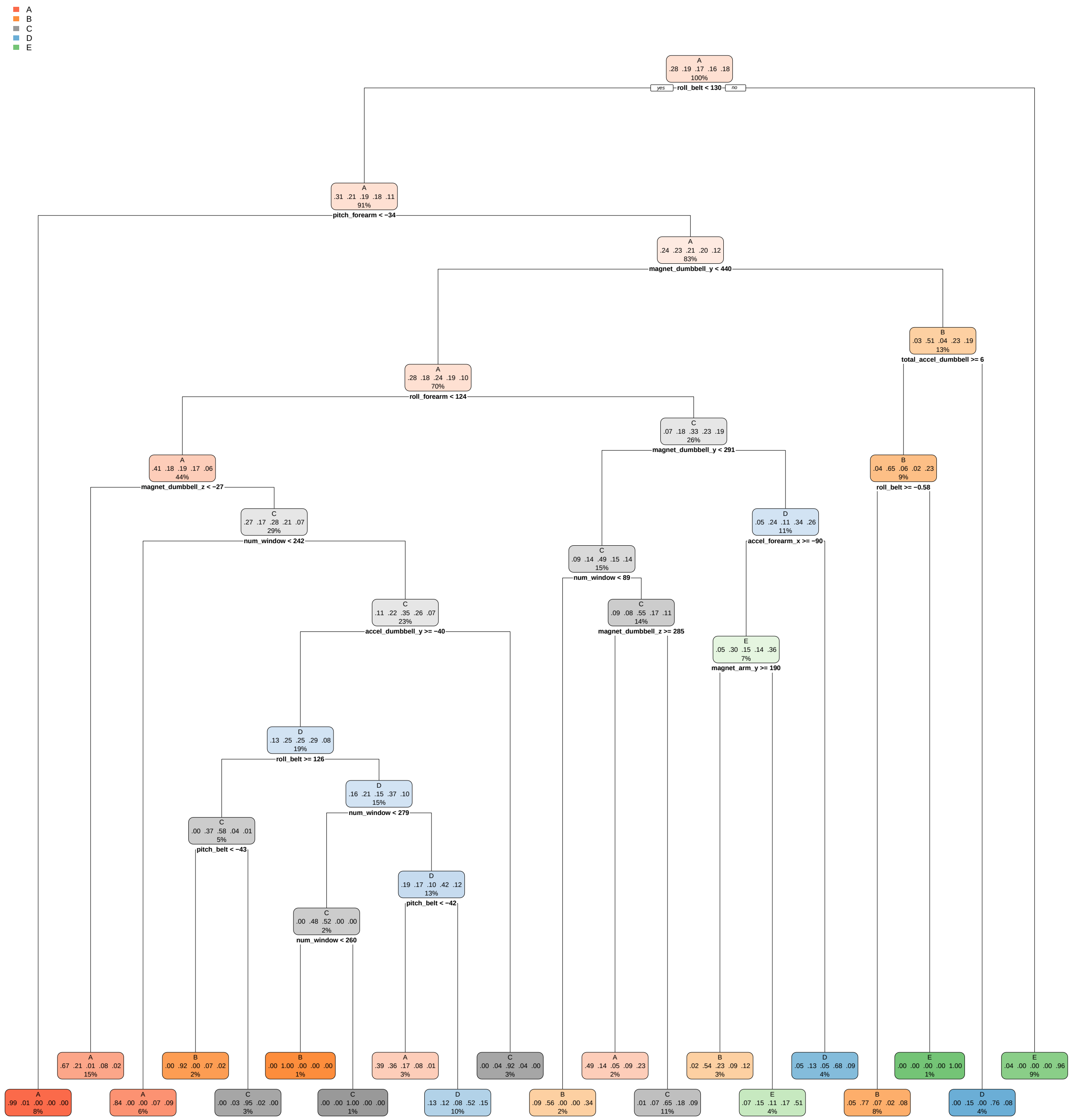
Create a decision tree model using all the variables

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.2
```

```r
set.seed(222)
model_DT1 <- rpart(classe ~., data = training_data, method = "class")

# Plot the trees
rpart.plot(model_DT1)
```




```r
testing_data$classe<-as.factor(testing_data$classe)
predict_tree <- predict(model_DT1, newdata= testing_data, type="class")
conMatrixtree <- confusionMatrix(testing_data$classe, predict_tree)
conMatrixtree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1491   38   10   87   48
##          B  251  643   63  153   29
##          C   50   71  822   54   29
##          D  107   38  132  647   40
##          E   98  104   63  120  697
##
## Overall Statistics
##
##                Accuracy : 0.7307
##                  95% CI : (0.7191, 0.742)
##     No Information Rate : 0.3393
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6573
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7466   0.7192   0.7541   0.6098   0.8268
## Specificity            0.9529   0.9006   0.9575   0.9343   0.9236
## Pos Pred Value         0.8907   0.5645   0.8012   0.6712   0.6442
## Neg Pred Value         0.8798   0.9471   0.9448   0.9159   0.9696
## Prevalence             0.3393   0.1519   0.1852   0.1803   0.1432
## Detection Rate         0.2534   0.1093   0.1397   0.1099   0.1184
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.8498   0.8099   0.8558   0.7720   0.8752
```

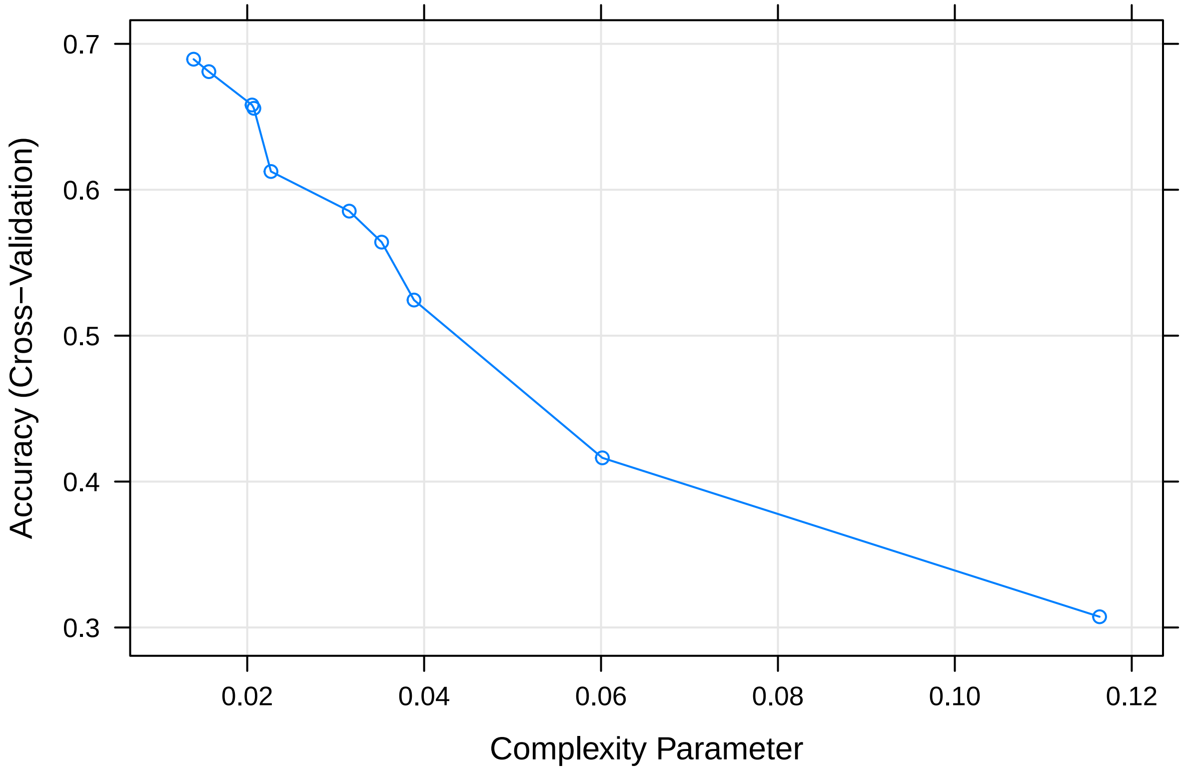
The accuracy of model_DT1 is 73.1%

Create a decision tree using cross validation to predict the accuracy of the model.

trControl is set to 10-fold cross validation and tuneLenght to 10

```r
set.seed(222)
model_DT2<- train(classe~., data = training_data, method = "rpart", trControl = trainControl("cv", numbe

plot(model_DT2)
```



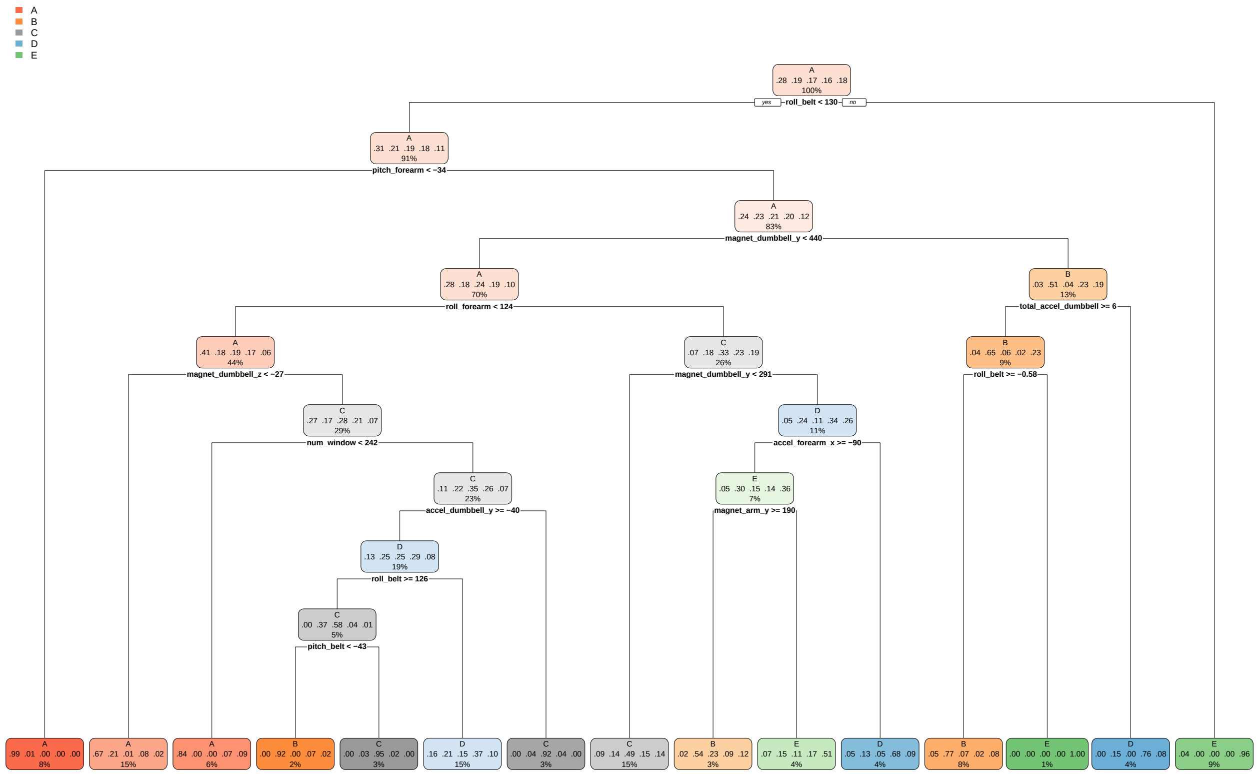
To determing the cp at which is obtained the best model accuracy

```
model_DT2$bestTune
```

```
##           cp
## 1 0.01393551
```

plot the best decision tree obtained

```
rpart.plot(model_DT2$finalModel)
```



Decisions rules the model

```
model_DT2$finalModel
```

```
## n= 13737
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##    1) root 13737 9831 A (0.28 0.19 0.17 0.16 0.18)
##      2) roll_belt< 129.5 12499 8640 A (0.31 0.21 0.19 0.18 0.11)
##        4) pitch_forearm< -33.65 1097   12 A (0.99 0.011 0 0 0) *
##        5) pitch_forearm>=-33.65 11402 8628 A (0.24 0.23 0.21 0.2 0.12)
##         10) magnet_dumbbell_y< 439.5 9640 6920 A (0.28 0.18 0.24 0.19 0.1)
##           20) roll_forearm< 123.5 6026 3573 A (0.41 0.18 0.19 0.17 0.055)
##             40) magnet_dumbbell_z< -27.5 2060  672 A (0.67 0.21 0.012 0.079 0.023) *
##             41) magnet_dumbbell_z>=-27.5 3966 2873 C (0.27 0.17 0.28 0.21 0.072)
##               82) num_window< 241.5 875  139 A (0.84 0.0011 0 0.072 0.086) *
##               83) num_window>=241.5 3091 1998 C (0.11 0.22 0.35 0.26 0.068)
##                166) accel_dumbbell_y>=-40.5 2629 1857 D (0.13 0.25 0.25 0.29 0.08)
##                  332) roll_belt>=125.5 625  261 C (0 0.37 0.58 0.038 0.0064)
##                    664) pitch_belt< -42.75 243   20 B (0 0.92 0 0.066 0.016) *
##                    665) pitch_belt>=-42.75 382   18 C (0 0.026 0.95 0.021 0) *
##                  333) roll_belt< 125.5 2004 1256 D (0.16 0.21 0.15 0.37 0.1) *
##                167) accel_dumbbell_y< -40.5 462   35 C (0 0.039 0.92 0.037 0) *
##           21) roll_forearm>=123.5 3614 2412 C (0.074 0.18 0.33 0.23 0.19)
##             42) magnet_dumbbell_y< 290.5 2105 1074 C (0.09 0.14 0.49 0.15 0.14) *
##             43) magnet_dumbbell_y>=290.5 1509  992 D (0.052 0.24 0.11 0.34 0.26)
##               86) accel_forearm_x>=-90.5 945  609 E (0.051 0.3 0.15 0.14 0.36)
##                172) magnet_arm_y>=189.5 372  172 B (0.019 0.54 0.23 0.094 0.12) *
```

```
##              173) magnet_arm_y< 189.5 573  283 E (0.072 0.15 0.11 0.17 0.51) *
##             87) accel_forearm_x< -90.5 564  179 D (0.053 0.13 0.046 0.68 0.089) *
##       11) magnet_dumbbell_y>=439.5 1762  869 B (0.031 0.51 0.044 0.23 0.19)
##         22) total_accel_dumbbell>=5.5 1255  439 B (0.043 0.65 0.06 0.018 0.23)
##           44) roll_belt>=-0.58 1053  237 B (0.051 0.77 0.071 0.021 0.082) *
##           45) roll_belt< -0.58 202    0 E (0 0 0 0 1) *
##         23) total_accel_dumbbell< 5.5 507  120 D (0 0.15 0.0039 0.76 0.081) *
##    3) roll_belt>=129.5 1238   47 E (0.038 0 0 0 0.96) *
```

Make predictions on the test data

```
testing_data$classe<-as.factor(testing_data$classe)
predicted_classe <-predict(model_DT2, testing_data)
```

Compute model accuracy rate on test data

```
mean(predicted_classe == testing_data$classe)
```

```
## [1] 0.6880204
```

The accuracy of model_DT2 is 68.8%

Predict testing cases

Based on the accuracy of the models we will use model_DT1 (use all variables to make the prediction).

```
testing_prediction1<-predict(model_DT1, newdata = testing, type="class")
testing_prediction1
```

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

EXTRA

We will compare the predictions obtain using model_DT1 and model_DT2.

Prediction using model_DT2

```
testing_prediction2<-predict(model_DT2, newdata = testing)
testing_prediction2
```

```
##  [1] C A B A A C D D A A C B C A E D A A A B
## Levels: A B C D E
```

Compare how many equal predictions we obtain using model_DT1 or model_DT2

```
sum(testing_prediction1 == testing_prediction2)
```

```
## [1] 18
```

We obtain equal results in 18 of 20 cases using any of the two models. Observing differences in two cases were model_DT1 should have a better performance.