Practical machine learning project (Human Activity Recognition)

Aim

1.Use variables to predict "classe" variable in the training set
2.Use your prediction model to predict 20 different test cases.

DATA

- Training data https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
- Test data https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
- Data source http://groupware.les.inf.puc-rio.br/har.

Load data

```
library(ggplot2); library(caret);
setwd("~/R/Cousera R/Machine learning")
har <- read.csv("pml-training.csv", header=TRUE, na.strings = c("NA",""," "))
#summary(har)
```

Many columns have NA's = 19216. Remove columns occupied with NA over 90%. Remove columns which is irrelevant to human activity: "X","user_name", "raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp", "new_window", "num_window".

```
NAnum <-apply(is.na(har),2,sum)
NAcol <- NAnum > nrow(har)*0.9
har <- har[,!NAcol]
har <- har[,8:60]
```

Genarate training and testing dataset

```
inTrain <- createDataPartition(y = har$classe, p = 0.4, list = FALSE)
training <- har[inTrain,]
testing <- har[-inTrain,]
```

First I tried 4 models (Decision tree, Naive Bayes, Random forest, Gradient boosting model) to choose best classification model for this dataset.

```
library(doSNOW)
cls<-makeCluster(6) # Number of cores you want to use
registerDoSNOW(cls) # Register the cores.

moddt <- train(classe ~ ., method= "rpart", data= training)             #Decision tree
modnb <- train(classe ~ ., data=training, method="nb")                  #Naive Bayes
modrf <- train(classe ~ ., data= training, method= "rf", prox= TRUE)    #Random forest
modgbm <- train(classe ~ ., method= "gbm",data= training, verbose=FALSE) #Gradient boosting model

stopCluster(cls)  # Free up your cores again.
```

Random forest model(modrf) was highest accuracy >0.95. Next I tuned this model by the number of folds in K-fold cross-validation

```
cls<-makeCluster(6) # Number of cores you want to use
registerDoSNOW(cls) # Register the cores.

modrf5 <- train(classe ~ ., data = training, method = "rf", trControl = trainControl(method = "repeatedcv", number = 5, repeats = 3
))
# Random forest with 5 fold cross validation
modrf10 <- train(classe ~ ., data = training, method = "rf", trControl = trainControl(method = "repeatedcv", number = 10, repeats =
 3))
# Random forest with 10 fold cross validation
modrf20 <- train(classe ~ ., data = training, method = "rf", trControl = trainControl(method = "repeatedcv", number = 20, repeats =
 3))
# Random forest with 20 fold cross validation

modrf5
```

```
## Random Forest
##
## 7850 samples
##   52 predictor
##    5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 3 times)
##
## Summary of sample sizes: 6282, 6280, 6279, 6280, 6279, 6281, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9822073  0.9774840  0.003404175  0.004310298
##   27    0.9824187  0.9777565  0.003987770  0.005047087
##   52    0.9729092  0.9657239  0.003914626  0.004952017
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
modrf10
```

```
## Random Forest
##
## 7850 samples
##   52 predictor
##    5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
##
## Summary of sample sizes: 7066, 7066, 7065, 7065, 7065, 7063, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9842877  0.9801180  0.005388643  0.006819348
##   27    0.9859856  0.9822703  0.004290732  0.005429966
##   52    0.9764305  0.9701817  0.006151434  0.007786966
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
modrf20
```

```
## Random Forest
##
## 7850 samples
##   52 predictor
##    5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (20 fold, repeated 3 times)
##
## Summary of sample sizes: 7457, 7457, 7459, 7457, 7457, 7457, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9850512  0.9810840  0.006514347  0.00824525
##   27    0.9862833  0.9826464  0.006245716  0.00790302
##   52    0.9761345  0.9698087  0.008408949  0.01063913
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

20 fold cross-validation model(modrf20) was the highest accuracy in training dataset. Higher number of corss validation reduced the error. Then I tested these models using tesing dataset(part of training dataset).

Check random forest model – best

```
predrf5 <- predict(modrf5,testing)
table(predrf5,testing$classe)
```

```
##
## predrf5    A    B    C    D    E
##       A 3342   27    1    0    2
##       B    1 2241   27    6    5
##       C    1   10 2016   31    5
##       D    0    0    9 1891   10
##       E    4    0    0    1 2142
```

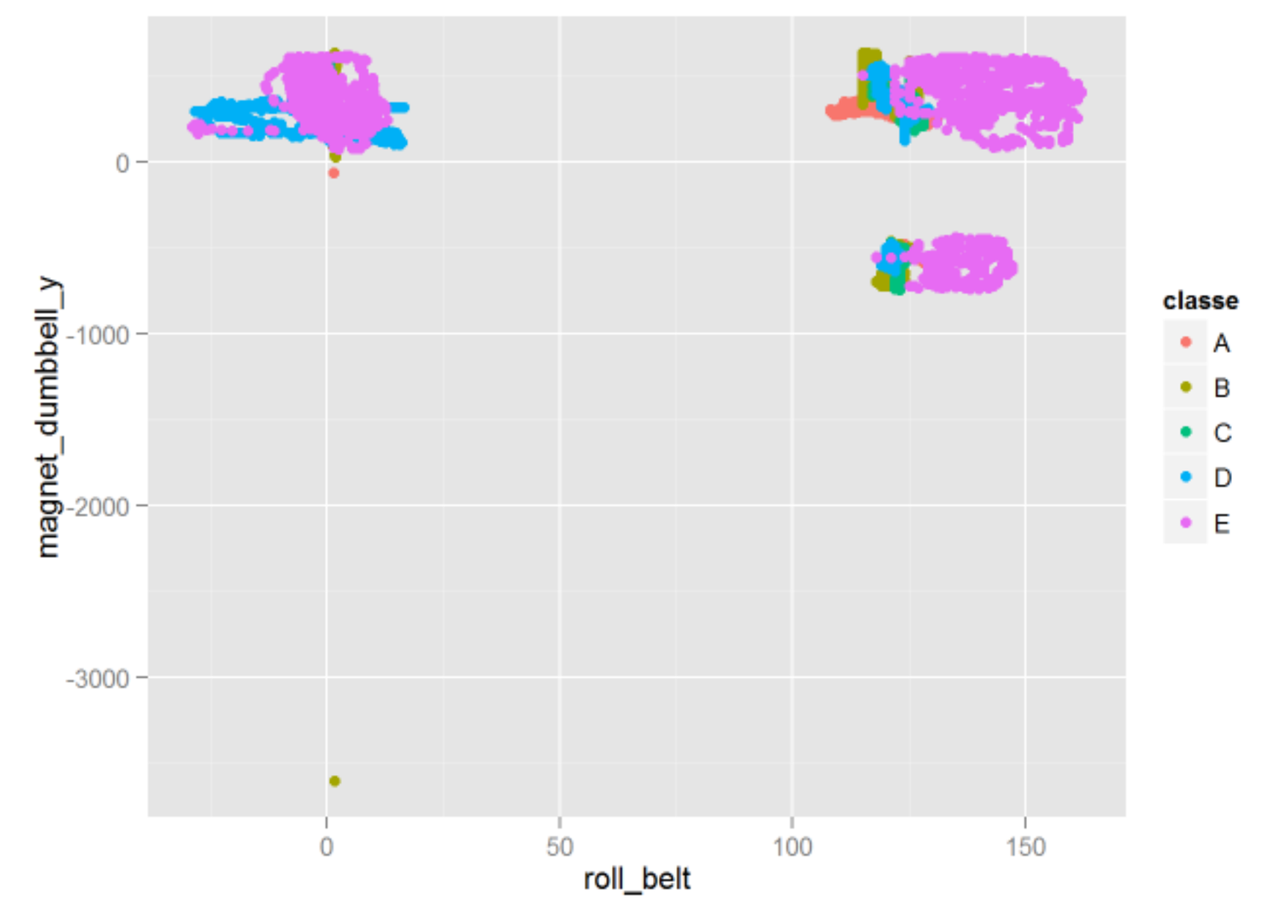
```
predrf10 <- predict(modrf10,testing)
table(predrf10,testing$classe)
```

```
##
## predrf10    A    B    C    D    E
##        A 3342   26    0    0    2
##        B    1 2240   27    3    4
##        C    1   12 2018   32    6
##        D    0    0    8 1892   10
##        E    4    0    0    2 2142
```

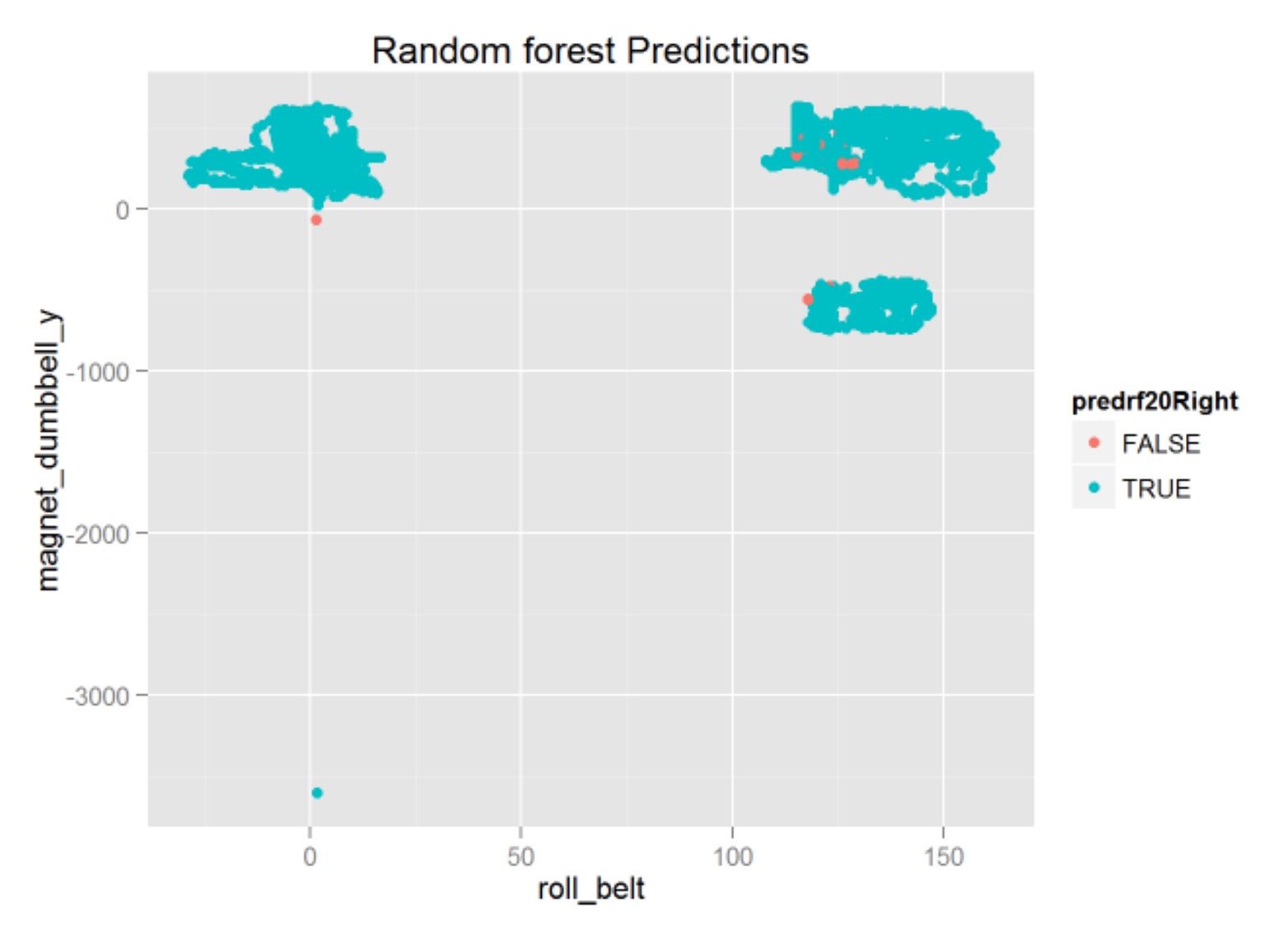
```
predrf20 <- predict(modrf20,testing)
table(predrf20,testing$classe)
```

```
##
## predrf20    A    B    C    D    E
##        A 3344   28    0    0    2
##        B    0 2238   28    4    4
##        C    1   12 2015   30    6
##        D    0    0   10 1894   10
##        E    3    0    0    1 2142
```

```
testing$predrf20Right <- predrf20==testing$classe
qplot(roll_belt, magnet_dumbbell_y, colour=classe, data=testing)
```




```
qplot(roll_belt, magnet_dumbbell_y,colour=predrf20Right,data=testing,main="Random forest Predictions")
```



Check other models

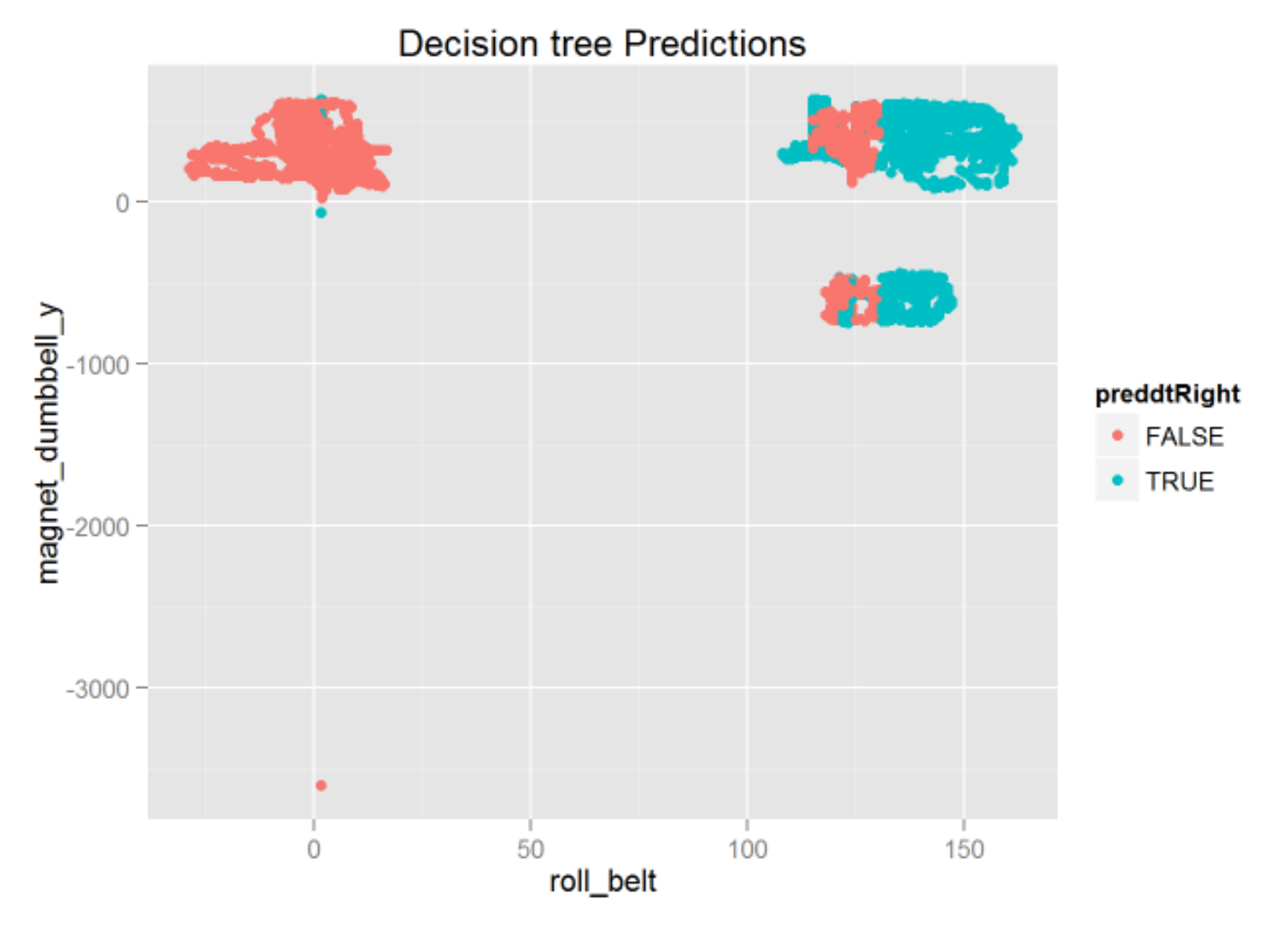
Other models are also visually inspected. To estimate the out of sample error the plots of prediction results were generated.

Check decision tree model – worst

```
#library(rattle)
#library(rpart.plot)
#fancyRpartPlot(moddt$finalModel)
preddt <- predict(moddt,testing); testing$preddtRight <- preddt==testing$classe
table(preddt,testing$classe)
```

```
##
## preddt    A    B    C    D    E
##      A 3043  989  971  878  316
##      B   54  585   32  336  128
##      C  240  704 1050  715  747
##      D    0    0    0    0    0
##      E   11    0    0    0  973
```

```
qplot(roll_belt, magnet_dumbbell_y,colour=preddtRight,data=testing,main="Decision tree Predictions")
```

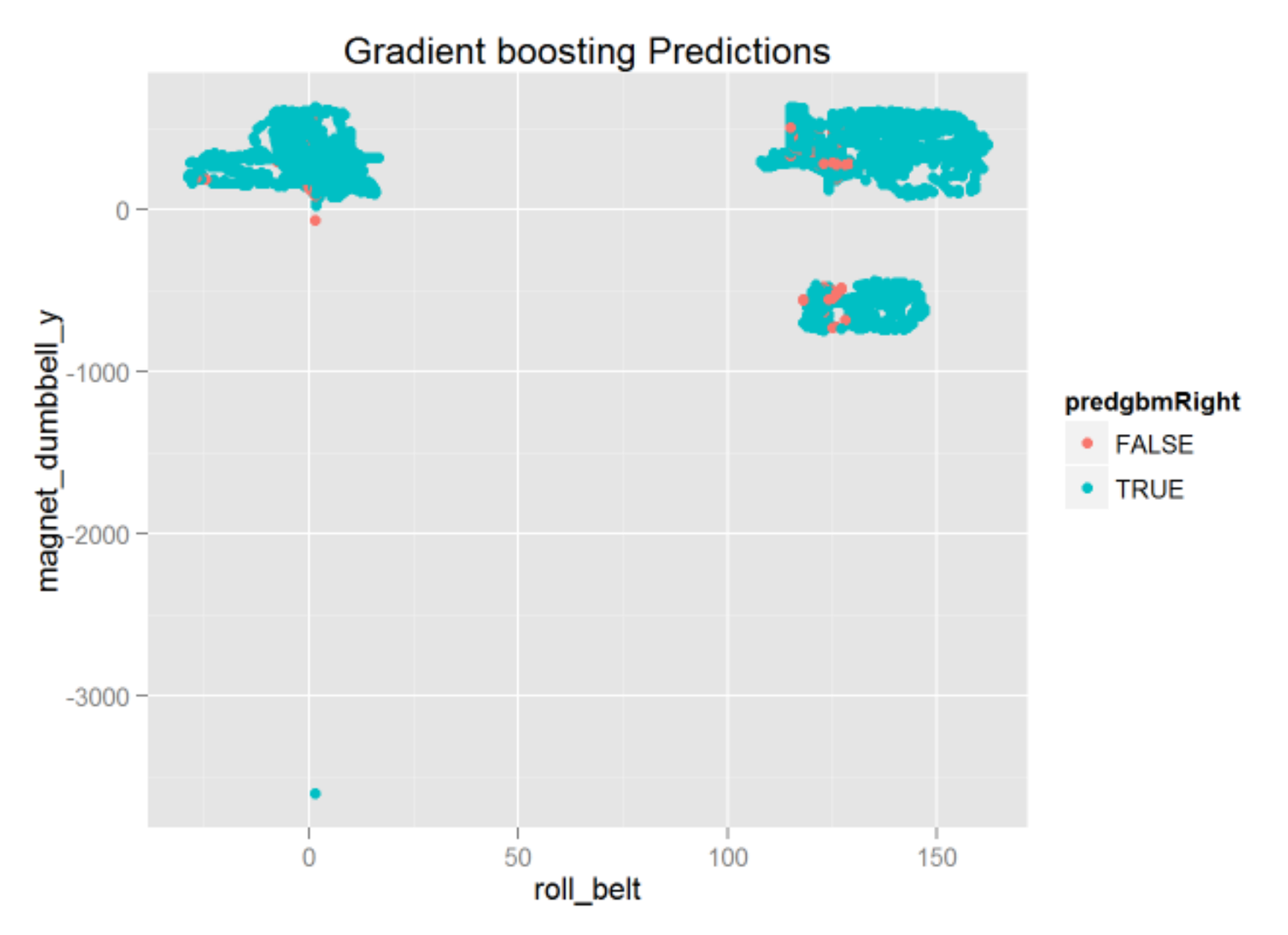


Check gradient boosting model – work

```
predgbm <- predict(modgbm,testing); testing$predgbmRight <- predgbm==testing$classe
table(predgbm,testing$classe)
```

```
##
## predgbm    A    B    C    D    E
##       A 3284   98    0    3    5
##       B   32 2123   83   13   22
##       C   10   49 1934   57   27
##       D    7    5   31 1842   39
##       E   15    3    5   14 2071
```

```
qplot(roll_belt, magnet_dumbbell_y,colour=predgbmRight,data=testing,main="Gradient boosting Predictions")
```

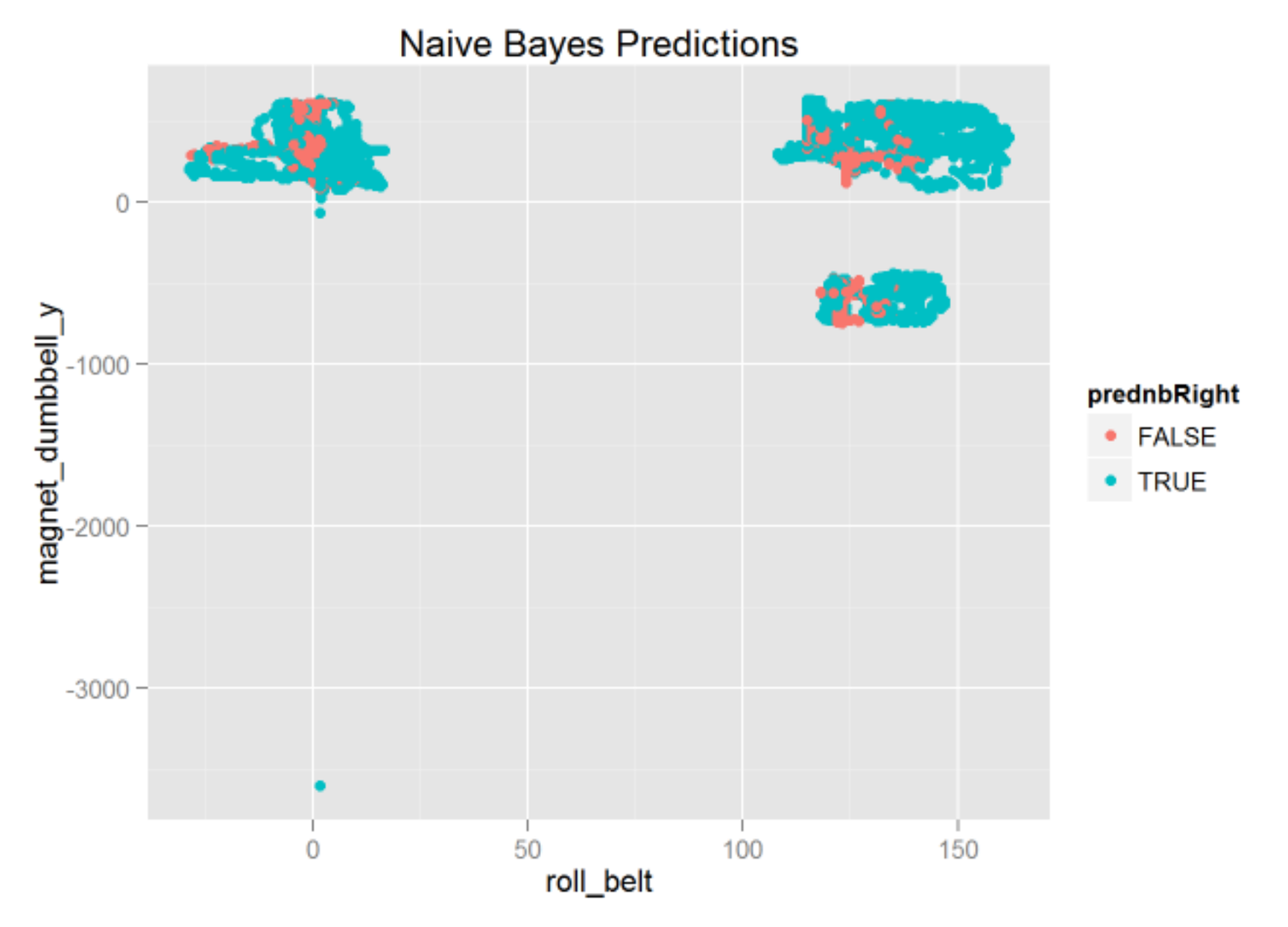


Check Naive Bayes model – work

```
prednb <- predict(modnb,testing); testing$prednbRight <- prednb==testing$classe
table(prednb,testing$classe)
```

```
##
## prednb    A    B    C    D    E
##      A 2422  198   91  109   54
##      B  130 1548  157   14  218
##      C  371  326 1639  346  131
##      D  406  188  161 1355   91
##      E   19   18    5  105 1670
```

```
qplot(roll_belt, magnet_dumbbell_y,colour=prednbRight,data=testing,main="Naive Bayes Predictions")
```




```
stopCluster(cls)
```

Final model

From the results above, I decided to use random forest "modrf20" model for the prediction.