report

2023-12-29

Dataset

source : 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.

A human activity recognition (HAR) dataset. Participants formed biceps curls in 5 different forms (Classe variable) while wearing a sensor in a glove, armband, and belt. There is also a sensor attached to the dumbbell.

The aim of this report is to create a predictor for the form based on the readings of the sensors.

```
dir.create("data", showWarnings=FALSE)
if (!file.exists("data/training.csv")) {
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "data/trainin
}
if (!file.exists("data/testing.csv")) {
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "data/testing.
}

training <- read.csv("data/training.csv")
testing <- read.csv("data/testing.csv")
```

Preprocessing

Some columns are almost all NAs, so I will first remove these.

```
nas <- colSums(is.na(training))
cols <- which(nas > 19000)
training <- training[, -cols]
testing <- testing[, -cols]
```

Remove variables with very little variance

```
nzvs <- nearZeroVar(training)
training <- training[, -nzvs]
testing <- testing[, -nzvs]
```

Remove highly correlated variables with over 0.9 correlation

```
cors <- cor(training[sapply(training, is.numeric)])
highcor <- findCorrelation(cors, cutoff=0.9)
training <- training[, -highcor]
testing <- testing[, -highcor]
```

My intuition is that we want to predict the activity based on the sensor output only, so I remove the id, name, and timestamp/sliding window related info, leaving just the sensor data and activity class.

```
training <- training[, -c(1, 2, 3, 4, 5, 6)]
testing <- testing[, -c(1, 2, 3, 4, 5, 6)]
```

Use preprocess to normalise data.

```
preProc <- preProcess(training, method=c("center", "scale"))
training <- predict(preProc, training)
testing <- predict(preProc, testing)
```

Model training

Keep aside 20% of the data so that we can measure accuracy on out-of-sample data.

```
tr_idx <- createDataPartition(training$classe, p=0.8, list=FALSE)
tr <- training[tr_idx,]
val <- training[-tr_idx,]
```

Use 5-fold cross validation.

```
fitControl <- trainControl(method = "cv", number = 5)
```

Fit using random forest, using doParallel to parallelize the cross validation, resulting in a big speed up.

```
library(doParallel)
cl <- makePSOCKcluster(5)
registerDoParallel(cl)
fit <- train(classe~., data=tr, method="rf", trControl=fitControl)
stopCluster(cl)
```

Training outcome

A simple random forest is quite successful, able to achieve a 99% accuracy on the validation data.

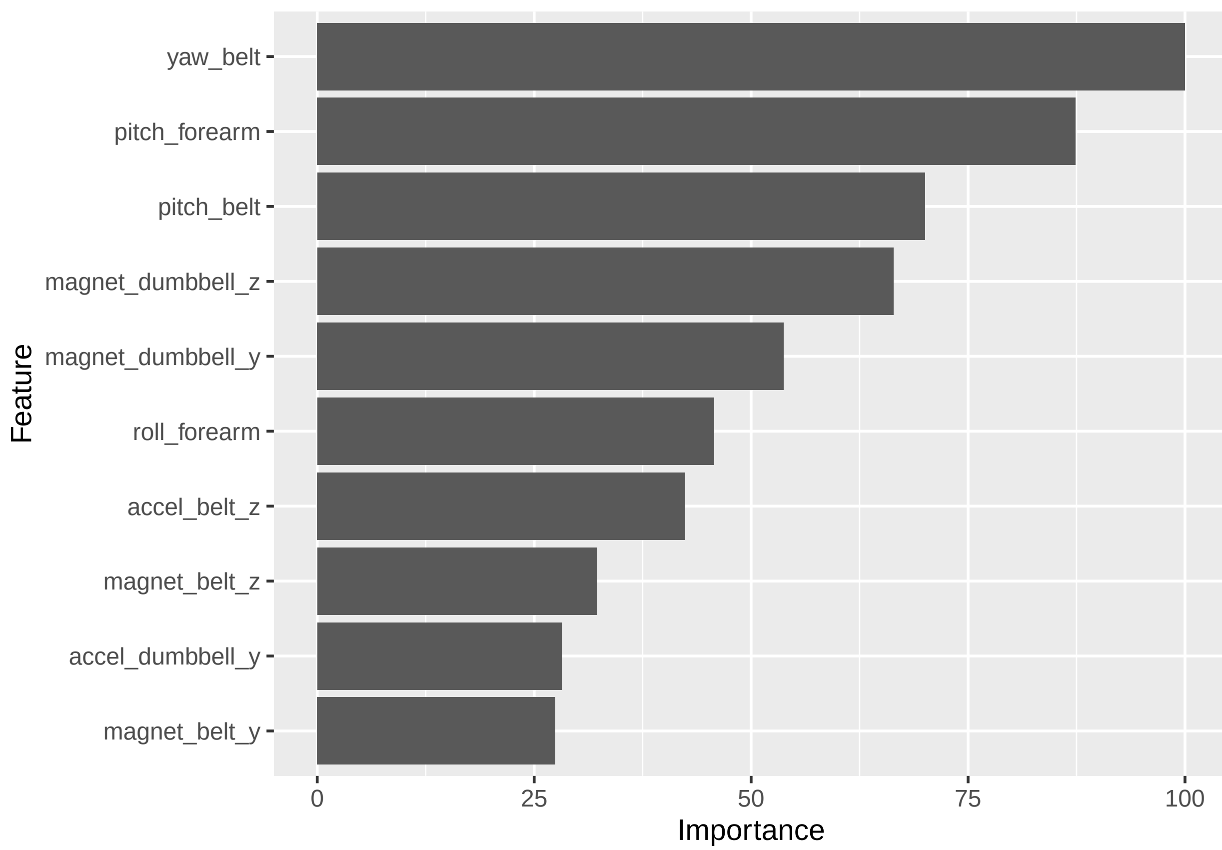
```
preds <- predict(fit, newdata = val)
confusionMatrix(preds, factor(val$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    4    0    0    0
##          B    0  754    2    0    0
##          C    0    1  680    9    2
##          D    0    0    2  634    4
##          E    0    0    0    0  715
##
## Overall Statistics
```

```
##
##                Accuracy : 0.9939
##                  95% CI : (0.9909, 0.9961)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9923
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9934   0.9942   0.9860   0.9917
## Specificity            0.9986   0.9994   0.9963   0.9982   1.0000
## Pos Pred Value         0.9964   0.9974   0.9827   0.9906   1.0000
## Neg Pred Value         1.0000   0.9984   0.9988   0.9973   0.9981
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1922   0.1733   0.1616   0.1823
## Detection Prevalence   0.2855   0.1927   0.1764   0.1631   0.1823
## Balanced Accuracy      0.9993   0.9964   0.9952   0.9921   0.9958
```

We can plot the importance of the variables found by the random forest, the yaw of the belt sensor seems to be most important.

```
ggplot(varImp(fit), top=10)
```



Predict

Apply the model to the test dataset to get the predictions.

```
predict(fit, newdata=testing)
```

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