Prediction Assignment Writeup

Human Activity Recognition

Executive Summary

To predict what exercise was performed based upon wearable tech data. Data obtained from http://groupware.les.inf.puc-rio.br/har

One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the manner in which they did the exercise according to the "classe" variable.

The following analysis was conducted…

- Get and clean the data: download, read and process
- Explore the data: find useful parameters
- Model development: extract, apply, and assess knowledge
- Conclusion: based on the data and analysis

Get the data

Download and read in the data.

```r
# URLs
train.url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test.url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

# File names
train.name <- "./data/pml-training.csv"
test.name <- "./data/pml-testing.csv"

# If non-existent create
if (!file.exists("./data")) {
  dir.create("./data")
}

# If non-existent download
if (!file.exists(train.name)) {
  download.file(train.url, destfile = train.name, method="curl")
}
if (!file.exists(test.name)) {
  download.file(test.url, destfile = test.name, method="curl")
}

# Load the data
train <- read.csv("./data/pml-training.csv")
test <- read.csv("./data/pml-testing.csv")
```

Clean the data

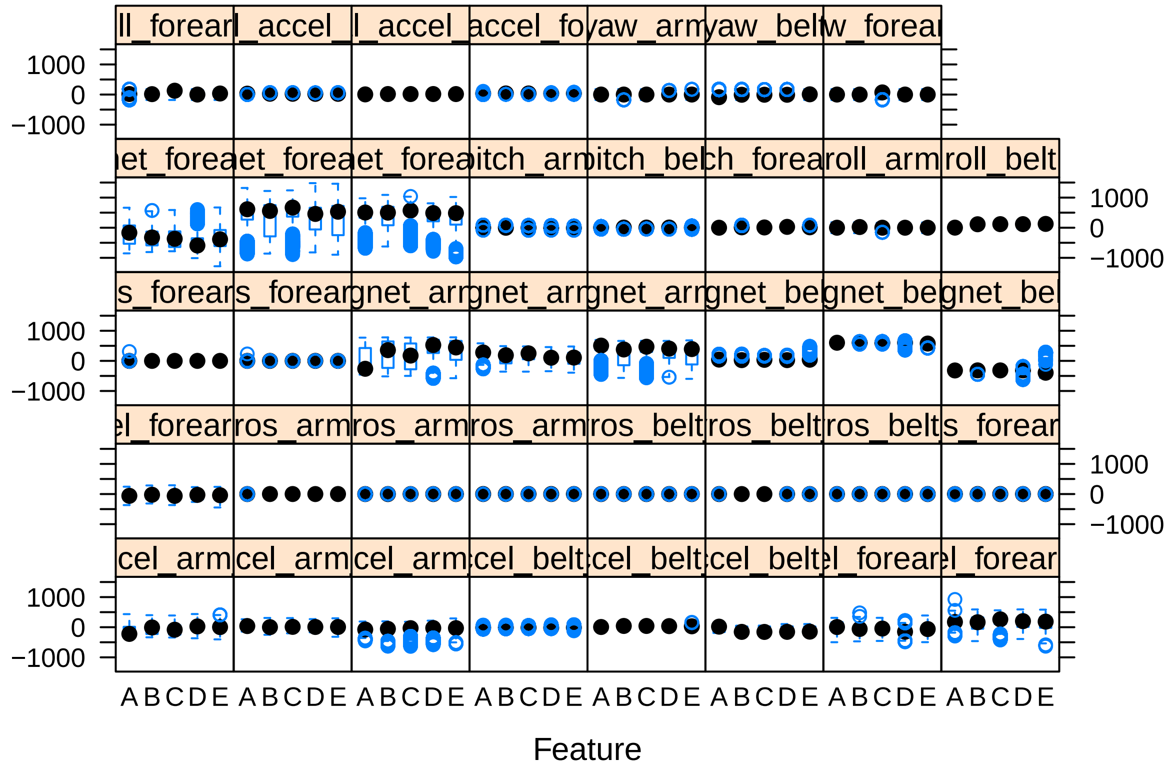
Process the data.

```r
# Inspect the data
# str(train)
# names(train)

# Identify prediction target
target.ori <- train[, "classe"]
target <- target.ori
# levels(target)

# Set character levels as numeric
num.class <- length(levels(target))
levels(target) <- 1:num.class
# head(target)

# Set prediction target as NULL (MLNB. t-Distributed Stochastic Neighbor Embedding)
train$classe <- NULL

# Select belt, forearm, arm, and dumbell variables (as per assignment instructions)
select <- grepl("belt|arm|dumbell", names(train))
train <- train[, select]
test <- test[, select]

# Clean the data by removing NAs (MLNB. t-Distributed Stochastic Neighbor Embedding)
cols.without.na <- colSums(is.na(test)) == 0
train <- train[, cols.without.na]
test <- test[, cols.without.na]
```

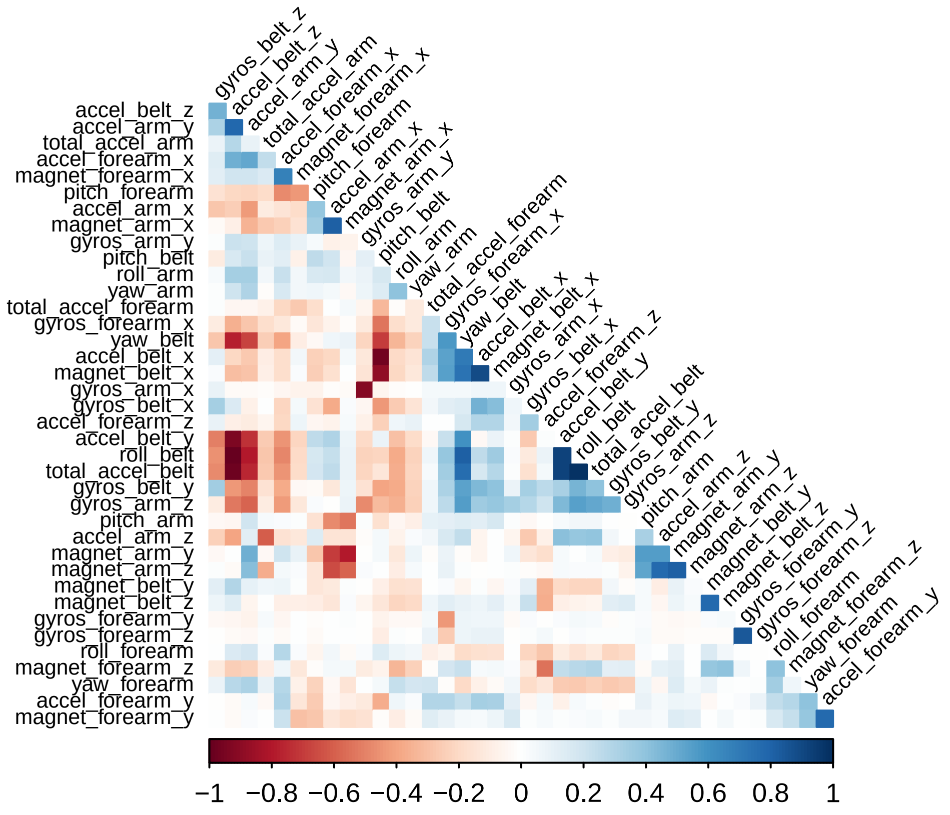
Explaoratory analysis

Iidentification, visualization and removal of near zero variance predictors followed by t-Distributed stochastic neighbor embedding. https://lvdmaaten.github.io/tsne/

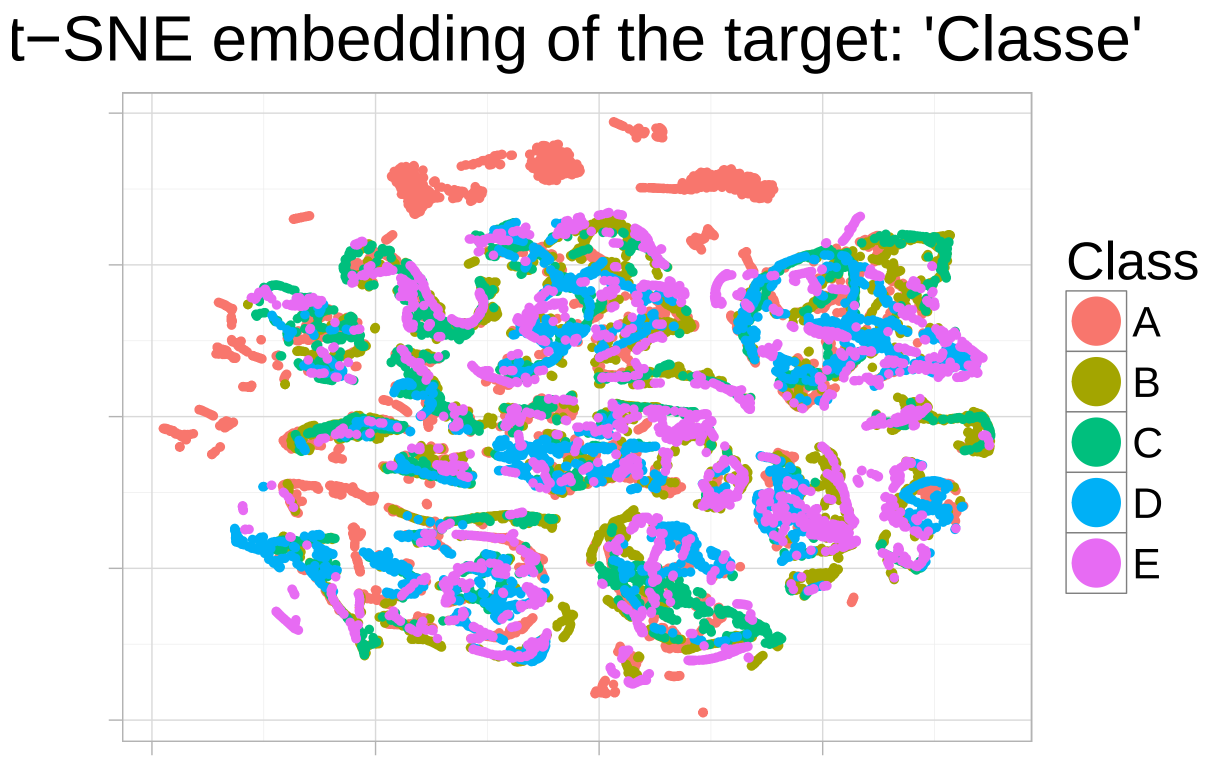
```r
# Check for zero variance
zero.var <- nearZeroVar(train, saveMetrics = TRUE)
# zero.var

# Correlation between features and the prediction target
featurePlot(train, target.ori, "box")
```



```
corrplot(cor(train), method = "color", type = "lower", order = "hclust", tl.cex = 0.70, tl.col="black",
```



```r
# The t-Distributed Stochastic Neighbor Embedding method
tDSNE <- Rtsne(as.matrix(train), check_duplicates = FALSE, pca = TRUE, perplexity = 50, theta = 0.25, di
embedding <- as.data.frame(tDSNE$Y)
embedding$Class <- target.ori
fig <- ggplot(embedding, aes(x = V1, y = V2, color = Class)) +
  geom_point(size = 1) +
  guides(colour = guide_legend(override.aes = list(size = 8))) +
  xlab("") +
  ylab("") +
  ggtitle(" t-SNE embedding of the target: 'Classe' ") +
  theme_light(base_size = 20) +
  theme(axis.text.x = element_blank(), axis.text.y = element_blank())
print(fig)
```



t-SNE is a (prize-winning) technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets (implemented via Barnes-Hut approximations).

Model development

Prepare the data

The extreme gradient boosting tree method requires numeric matrix data.https://cran.r-project.org/web/packages/xgboost/

```
# Format train dataset as a matrix and set as numeric
Mtrain <- as.matrix(train)
mode(Mtrain) = "numeric"

# Format test dataset as a matrix and set as numeric
Mtest = as.matrix(test)
mode(Mtest) = "numeric"

# Set target dataset as numeric
y <- as.matrix(as.integer(target)-1)
```

Train, cross-validate, and evaluate the model

Set the parameters for training and cross validation, as well as setting a multiple classification objective as the learning function. Evaluate with merror, confusion matrix, and test data.

```r
# xgboost parameters
xgbp <- list("objective" = "multi:softprob", "num_class" = num.class, "eval_metric" = "merror", "nthreac

# Set the seed
set.seed(4567)

# Timed k-fold cross validation
system.time( bst.cv <- xgb.cv(param = xgbp, data = Mtrain, label = y, nfold = 5, nrounds = 100, predicti
```

```
##    user  system elapsed
##  433.28   12.66  121.37
```

```r
# head(bst.cv$dt)
# tail(bst.cv$dt)

# Index of minimum merror
min.merror.idx <- which.min(bst.cv$dt[, test.merror.mean])
# min.merror.idx

# Minimum merror
bst.cv$dt[min.merror.idx,]
```

```
##    train.merror.mean train.merror.std test.merror.mean test.merror.std
## 1:                 0                0         0.005504        0.001987
```

```r
# Prediction decoding
pred.cv <- matrix(bst.cv$pred, nrow = length(bst.cv$pred)/num.class, ncol = num.class)
pred.cv <- max.col(pred.cv, "last")

# Confusion matrix
confusionMatrix(factor(y+1), factor(pred.cv))
```

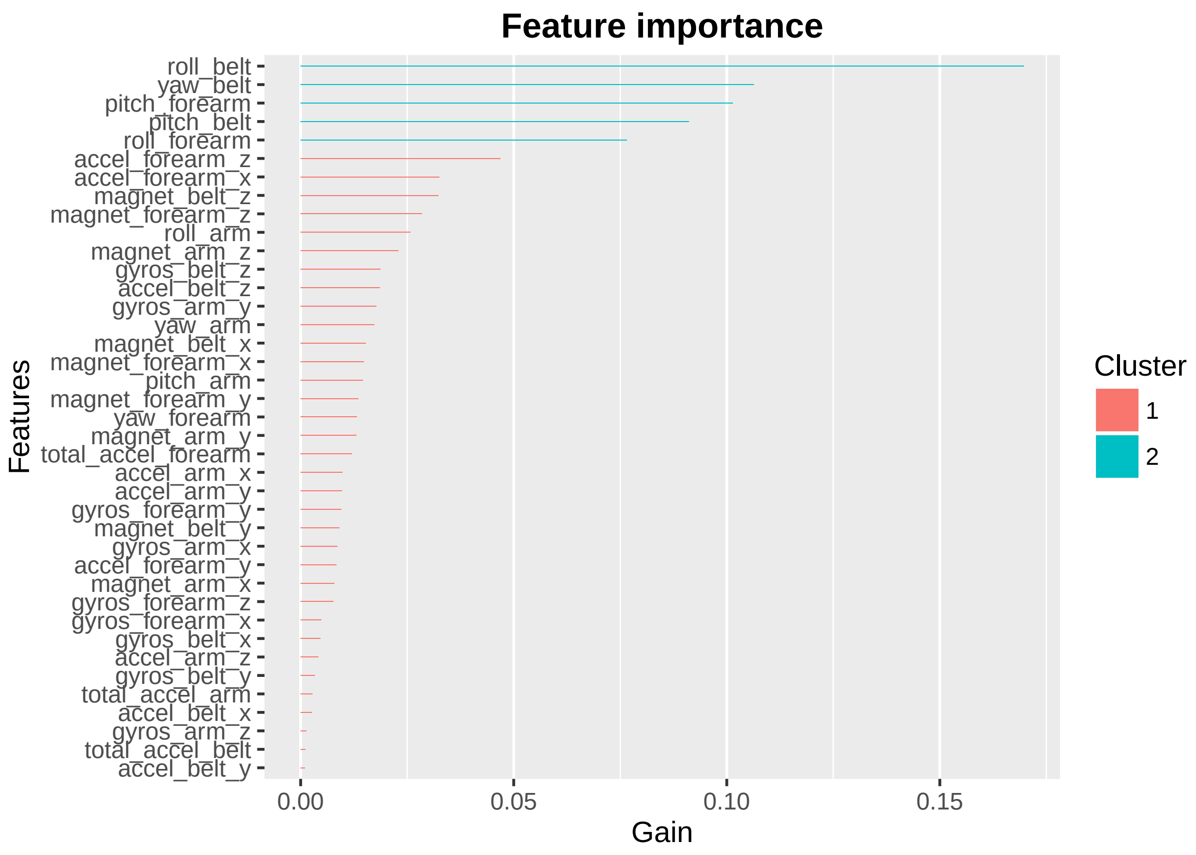
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    2    3    4    5
##          1 5570    4    4    2    0
##          2    8 3770   19    0    0
##          3    0   27 3380   15    0
##          4    0    1   21 3191    3
##          5    0    1    1    4 3601
##
## Overall Statistics
##
##                Accuracy : 0.9944
##                  95% CI : (0.9932, 0.9954)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9929
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            0.9986   0.9913   0.9869   0.9935   0.9992
## Specificity            0.9993   0.9983   0.9974   0.9985   0.9996
## Pos Pred Value         0.9982   0.9929   0.9877   0.9922   0.9983
## Neg Pred Value         0.9994   0.9979   0.9972   0.9987   0.9998
## Prevalence             0.2843   0.1938   0.1745   0.1637   0.1837
## Detection Rate         0.2839   0.1921   0.1723   0.1626   0.1835
## Detection Prevalence   0.2844   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9989   0.9948   0.9921   0.9960   0.9994
```

```r
# Model fit
system.time( bst <- xgboost(param = xgbp, data = Mtrain, label = y, nrounds = min.merror.idx, verbose =
```

```
##    user  system elapsed
##   88.08    2.39   24.96
```

```r
# xgboost predict test data using the trained model
pred <- predict(bst, Mtest)
# head(pred, 10)

# Prediction set as character
pred <- matrix(pred, nrow = num.class, ncol = length(pred)/num.class)
pred <- t(pred)
pred <- max.col(pred, "last")
pred.char <- toupper(letters[pred])

# Get the trained model
model <- xgb.dump(bst, with.stats = TRUE)

# Get the feature real names
names <- dimnames(Mtrain)[[2]]

# Compute feature importance matrix
importance_matrix <- xgb.importance(names, model = bst)

# Plot
gp <- xgb.plot.importance(importance_matrix)
print(gp)
```



The above feature importance plot ranks features in order of the highest correlation to the prediction target.

Write output

Print out and write to file the predictions.

```
# Print
pred.char
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
path = "./data/"
pml_write_files = function(x) {
  n = length(x)
  for(i in 1: n) {
    filename = paste0("problem_id_", i, ".txt")
    write.table(x[i], file=file.path(path, filename),
                quote=FALSE, row.names=FALSE, col.names=FALSE)
  }
}
pml_write_files(pred.char)
```

Conclusion

The confusion matrix of the xgb-model (5-fold cross validation with 100 cycles) shows an overall accuracy of ~ 99.5%. Computational analysis completed with an elapsed time ~ 2mins. To potentially improve the model (both in terms of computation time and reduce possible overfitting), low ranking features may be removed (cluster 1).