Implementation of pointwise

Introduction

Here we implement a simple case of usage of pointwise to ranking the dataset we defined. We convert it to an classification problem and the label in our dataset is defined manually among {0,1,2,3}. The machine learning algorithm we used is random forest with 5 fold cross validation. Finally, we rank the prediction result according to the pred label.

Loading libraries and setup

```
setwd('/Users/mingjie/Desktop/[IR]ltr')
suppressMessages(require(randomForest))
library(data.table)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```
ohsumed.dt <- fread("./data/dataset.csv", sep = ",", header=TRUE)
head(ohsumed.dt)
```

```
##    label qid docid c1 c2 c3
## 1:     0   1     1  0  0  0
## 2:     1   1     2  1  0  0
## 3:     2   1     3  1  1  0
## 4:     0   1     4  0  0  0
## 5:     0   1     5  0  0  0
## 6:     1   1     6  1  0  0
```

Name the dataset:

```
## rename
col.names <- paste(rep("C", 3), seq(1:3), sep = "")
col.names <- c("r", "qid", "docid", col.names)
setnames(ohsumed.dt, col.names)
## use the benefits of data.table to quicly prepare the data.
letor <- ohsumed.dt

## visualize a small fraction data
head(letor)
```

```
##    r qid docid C1 C2 C3
## 1: 0   1     1  0  0  0
## 2: 1   1     2  1  0  0
## 3: 2   1     3  1  1  0
## 4: 0   1     4  0  0  0
```

```
## 5: 0   1     5  0  0  0
## 6: 1   1     6  1  0  0
```

```
length(unique(letor$qid)) ## number of queries
```

```
## [1] 3
```

```
length(unique(letor$docid)) ## unique docid
```

```
## [1] 60
```

Prepare data

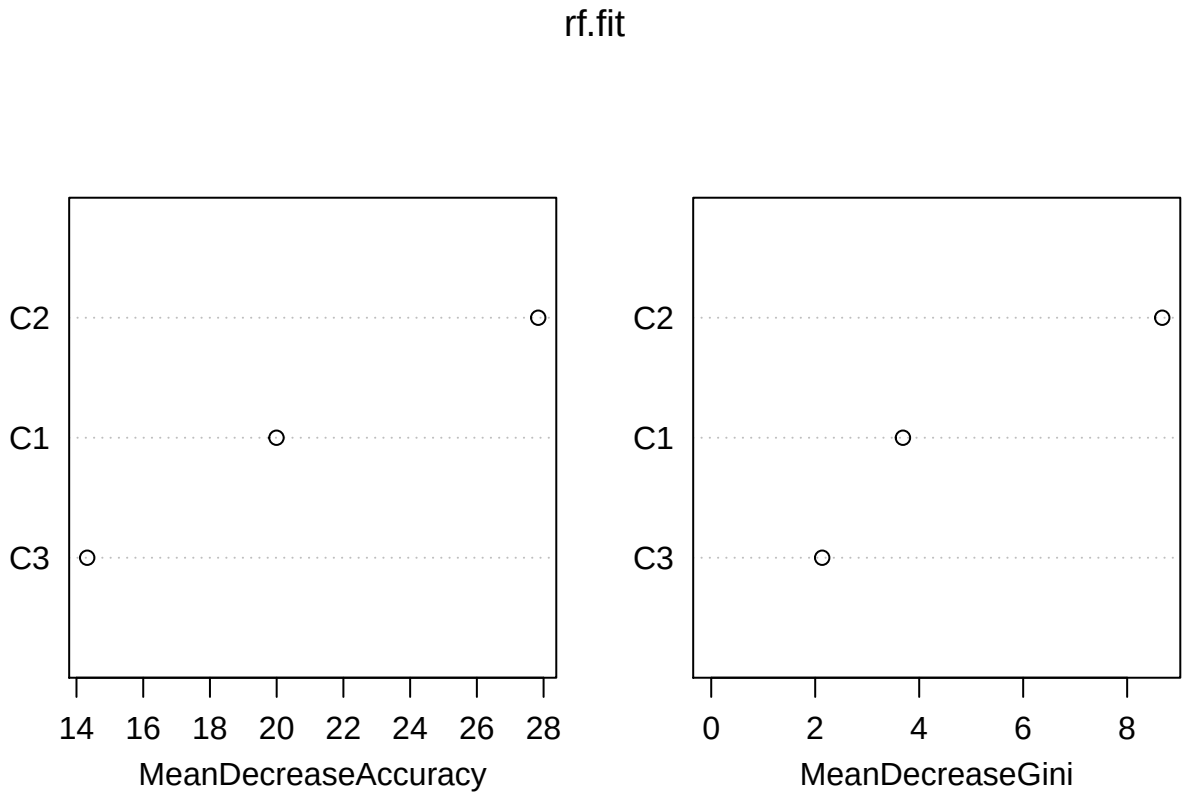
Column "r" = relevance: The larger value the relevance label has, the more relevant the query-docs pair. For this assignment I am going to consider 2,3 as relevant and 0,1 as irrelevant

Model

Here, we define our train control with 5 fold cv and use random forest to train. The features are C1,C2,C3; the lable is r.

```
# Define train control for k fold cross validation
train_control <- trainControl(method="cv", number=5)
##fit the model
formula <- as.factor(r) ~ C1+C2+C3

rf.fit <- randomForest(formula, train, importance=TRUE, trControl=train_control)
varImpPlot(rf.fit)
```



The predicted classification and confusion matrix. We could find that the accuracy is 100%.

```
prediction <- predict(rf.fit, test)
confusionMatrix(prediction, factor(test$r,levels = c(0,1,2,3)))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1 2 3
##          0 4 0 0 0
##          1 0 6 0 0
##          2 0 0 0 0
##          3 0 0 0 2
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.7354, 1)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 0.0002441
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3
## Sensitivity            1.0000      1.0       NA   1.0000
## Specificity            1.0000      1.0        1   1.0000
## Pos Pred Value         1.0000      1.0       NA   1.0000
## Neg Pred Value         1.0000      1.0       NA   1.0000
## Prevalence             0.3333      0.5        0   0.1667
## Detection Rate         0.3333      0.5        0   0.1667
## Detection Prevalence   0.3333      0.5        0   0.1667
## Balanced Accuracy      1.0000      1.0       NA   1.0000
```

Ranking

Finally, we rank the test set according to the predicted classification:

```
rankdf <- data.frame(qid = test$qid, docid = test$docid, r= test$r, pred = prediction)
rankdf <- rankdf[order(rankdf$pred, decreasing = TRUE),]
rankdf
```

```
##    qid docid r pred
## 7    3    41 3    3
## 10   3    50 3    3
## 2    1     2 1    1
## 5    2    35 1    1
## 8    3    45 1    1
## 9    3    46 1    1
## 11   3    53 1    1
## 12   3    57 1    1
## 1    1     1 0    0
## 3    1    13 0    0
```

```
## 4    2    24 0    0
## 6    2    38 0    0
```

Given a query, return the ranking. Here, we use query 1 (glucose in blood) as example.

```
rankdf[rankdf[, "qid"] == 1,]
```

```
##   qid docid r pred
## 2   1     2 1    1
## 1   1     1 0    0
## 3   1    13 0    0
```

Because the dataset provided is pretty small, we show the ranking for the whole dataset including training and testing in order to visualize better.

```
ranking <- predict(rf.fit, letor)
rankdf2 <- data.frame(qid = letor$qid, docid = letor$docid, r= letor$r, pred = ranking)
rankdf2 <- rankdf2[order(rankdf2$pred, decreasing = TRUE),]
rankdf2
```

```
##    qid docid r pred
## 41   3    41 3    3
## 50   3    50 3    3
## 54   3    54 3    3
## 58   3    58 3    3
## 2    1     2 1    1
## 3    1     3 2    1
## 6    1     6 1    1
## 8    1     8 1    1
## 9    1     9 1    1
## 11   1    11 1    1
## 12   1    12 1    1
## 16   1    16 1    1
## 17   1    17 1    1
## 19   1    19 1    1
## 22   2    22 1    1
## 23   2    23 1    1
## 25   2    25 1    1
## 27   2    27 1    1
## 29   2    29 2    1
## 30   2    30 2    1
## 33   2    33 1    1
## 34   2    34 1    1
## 35   2    35 1    1
## 36   2    36 1    1
## 37   2    37 1    1
## 39   2    39 1    1
## 40   2    40 1    1
## 42   3    42 1    1
## 43   3    43 1    1
## 44   3    44 1    1
## 45   3    45 1    1
## 46   3    46 1    1
## 47   3    47 1    1
## 49   3    49 1    1
## 51   3    51 1    1
## 52   3    52 1    1
```

```
## 53   3    53 1    1
## 56   3    56 2    1
## 57   3    57 1    1
## 60   3    60 1    1
## 1    1     1 0    0
## 4    1     4 0    0
## 5    1     5 0    0
## 7    1     7 0    0
## 10   1    10 0    0
## 13   1    13 0    0
## 14   1    14 0    0
## 15   1    15 0    0
## 18   1    18 0    0
## 20   1    20 0    0
## 21   2    21 0    0
## 24   2    24 0    0
## 26   2    26 0    0
## 28   2    28 0    0
## 31   2    31 0    0
## 32   2    32 0    0
## 38   2    38 0    0
## 48   3    48 0    0
## 55   3    55 0    0
## 59   3    59 0    0
```

```
rankdf2[rankdf2[, "qid"] == 1,]
```

```
##    qid docid r pred
## 2    1     2 1    1
## 3    1     3 2    1
## 6    1     6 1    1
## 8    1     8 1    1
## 9    1     9 1    1
## 11   1    11 1    1
## 12   1    12 1    1
## 16   1    16 1    1
## 17   1    17 1    1
## 19   1    19 1    1
## 1    1     1 0    0
## 4    1     4 0    0
## 5    1     5 0    0
## 7    1     7 0    0
## 10   1    10 0    0
## 13   1    13 0    0
## 14   1    14 0    0
## 15   1    15 0    0
## 18   1    18 0    0
## 20   1    20 0    0
```