Practical machine learning project

realsvik

Sunday, November 22, 2015

Summary

This document contains the steps to develop a machine learning algorythm to predict activity quality from activity monitors for Coursera Parctical machine learning class.

As a result of investigation, randomForest algorytm was chosen to be applied on data without PCA preprocessing.

Choice was based on model accuracy as the measure of out of sample error. Acceptable accuracy is chosen to be greater than 0.9

The selected algorythm showed 0.995 on cross validation data set.

More details on the dataset can be found at the researchers website: http://groupware.les.inf.puc-rio.br/har

Investigation approach

Test dataset will be split to 60% training and 40% testing parts.

I am going to build Random Forest model 2 times: * with PCA preprocessing * without PCA preprocessing and use the approach with higher accuracy.

Before building the models, the data will be cleaned.

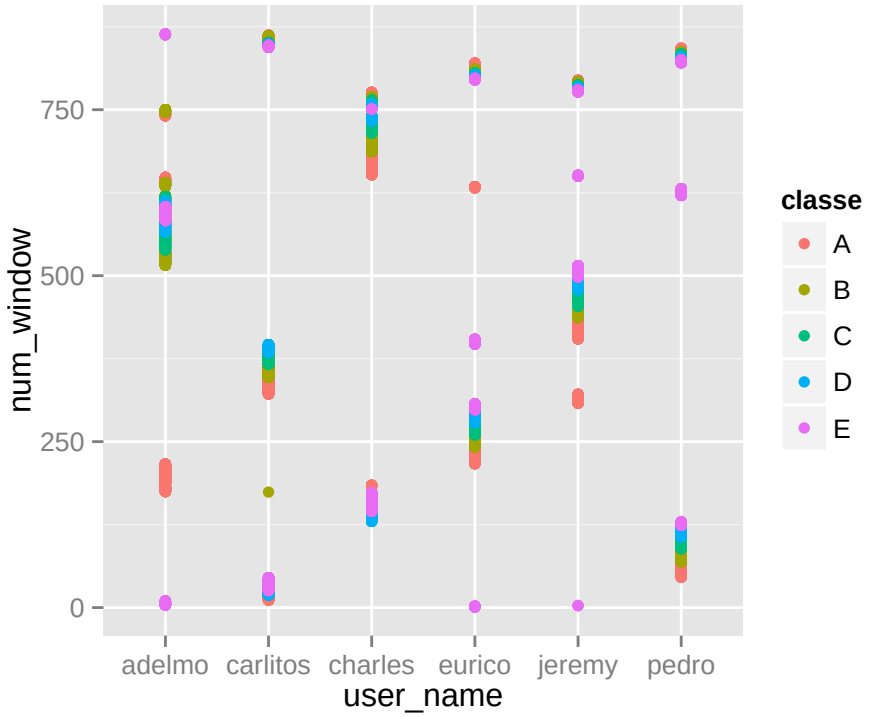
Building the algorythm

Start with loading data for traning and testing.

```
## Loading required package: lattice
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

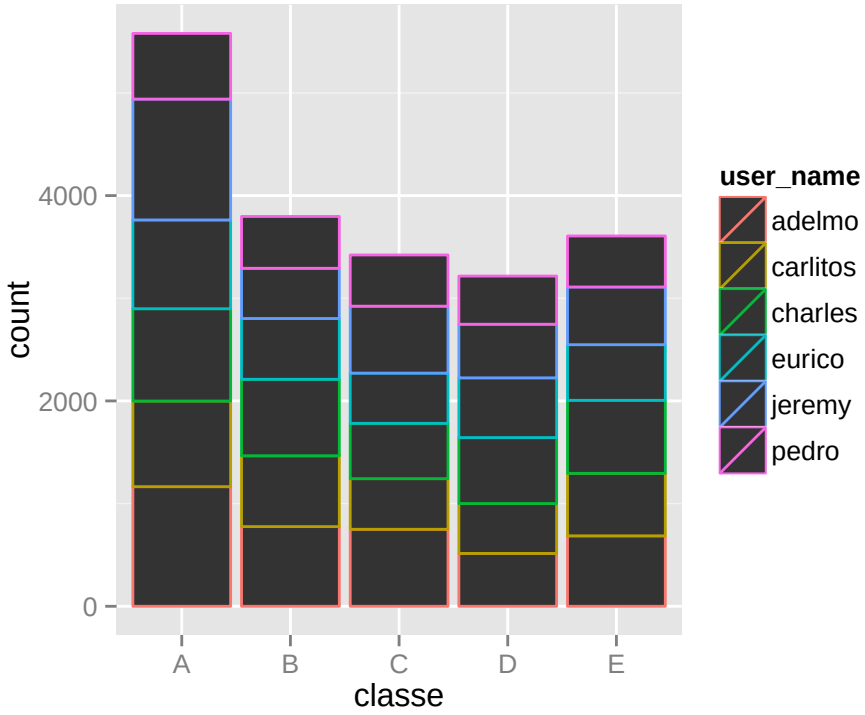
Exploratory analysis

The following plot shows activities, performed by athletes in different time windows. Activities colored by quality marks. Acitivities are performed in not overlapped time windows. Number of excercises per athlete does



not seem to be skewed.

Second plot shows, that in training set there are more "A" classified exerisces, however the diffrence is not dramatical, so I do not need to work on log scale. Also, activity quality classes seem to be evenly distributed between atgletes, so no one is inclined, for example, to do more "B" or "E" classified attempts.



Above said allows me to randomly split training into 60% training and 40% cross validation sets.

Data cleansing

Function cleanSet cleans data from near zero variance variables and the variables, which are over 95% NAs. It also removes acitivity numbers and character dates, because caret does not predict with dates and there is are date variables available.

Function createDums turns variable User_name into dummy variables.

```
#remove NAs, NZV
training<-cleanSet(training)
training<-createDums(training)

#cross validation preparation
testing<-cleanSet(testing)
testing<-createDums(testing)
```

Caret work

Firts I prepare training and cross validation objects with PCA and without PCA. Basing on above objects I build models, using Random forest algorithm.

```
#preprocessing with pca
preProc<-preProcess(training[,-61], method=c("pca", "center", "scale", "knnImpute"), thresh = 0.95)
trainPC<-predict(preProc,training[,-61])

#preprocessing without PCA
preProcNpca<-preProcess(training[,-61], method=c("knnImpute"))
trainPCNpca<-predict(preProcNpca,training[,-61])

#cross validation with and w\o PCA
testPC<-predict(preProc,testing[,-61])
testPCNpca<-predict(preProcNpca,testing[,-61])

#randomForest
modelFit<-randomForest(training$classe ~ ., data = trainPC)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values.  Are you sure you want to do regression?
```

```
roundPred<-round(predict(modelFit, newdata=testPC))
cm <- confusionMatrix(testing$classe,roundPred)

#random forest without pca
modelFitNpca<-randomForest(training$classe ~ ., data = trainPCNpca)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values.  Are you sure you want to do regression?
```

```
roundPredNpca<-round(predict(modelFitNpca, newdata=testPCNpca))
cmNpca <-confusionMatrix(testing$classe,roundPredNpca)
```

Accoracy of non PCA-based model 0.9956666 is higher, than for PSA-based one, 0.7732603 This is why non PCA model will be used to predict acitivity quality on test set.

```
testfitset<-cleanSet(testfitset)
testfitset<-createDums(testfitset)
testfitPCNpca<-predict(preProcNpca,testfitset)
roundTestClasse<-round(predict(modelFitNpca, newdata=testfitPCNpca))
#convert number predictions into letters
answers<-convertToLetters(roundTestClasse)
```

Finally, I create files with answers to submit for the 2nd part of the task

```
#create files with answers
pml_write_files(answers)
```

As a conclusion, the chosen algorithm was right in 19 of 20, which gives 95% of right answers. This meets the expectations to develop an algoritm with accuracy greater than 0.9

Appendix

Functions, used in the project

```
#function to remove NAs and non-zero vars
cleanSet <- function(dataset){
  length<-ncol(dataset)
  NAcols<-0
  thresh<-0.95
  for (i in 1:length){
    if(mean(is.na(dataset[,i]>thresh))){
      NAcols[i]<-"FALSE"
    }else NAcols[i]<-"TRUE"
  }
  NAcols<-as.logical(NAcols)
  dataset<-dataset[,NAcols]
  NonNzv<-nearZeroVar(dataset, saveMetrics=TRUE)
  dataset<-dataset[,!(NonNzv$nzv)]
#remove row number and date, because caret does not predict with dates
  dataset<-dataset[,-c(1,5),drop=FALSE]
  dataset$user_name<-as.factor(dataset$user_name)
  if("classe" %in% colnames(dataset))
  {

  dataset$classe<-as.factor(dataset$classe)
  dataset$classe<-as.numeric(dataset$classe)
  }
  if("problem_id" %in% colnames(dataset))
  {

    dataset$problem_id<-NULL
  }
  return(dataset)
}

createDums <- function(dataset){
  dummies<-dummyVars(~user_name, data=dataset, fullRank = TRUE)
```

```
  dumds<-predict(dummies, newdata=dataset)
  dataset<-cbind(dumds, dataset)
  dataset$user_name<-NULL
  return(dataset)
}
#convert numerical preeiction back to letters
convertToLetters<-function(classes){
  answer<-0
  convertArr<-c("A", "B", "C", "D", "E")
  for (i in 1:length(classes)){
    answer[i]<-convertArr[classes[i]]
  }
  return(answer)
}
#create files with answers
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
```