Coursara Practical Machine Learning Project - v5

Chudnow

June 14, 2015

1. Executive Summary

Data sets relating to human activity recognition were examined to develop algorithms to recognize if a weight lifting exercises is being done correctly (classe="A") or identify how it is being done improperly (class="B").

After partioning the training data set into 60% training, 20% testing/probe, and 20% quiz/validation data sets, I examined the training data to see if any patterns could be found to assist in training. No clear patterns where found in any particular single column. Using a PCA analysis it two dimensions revealed clear clustering by `user_name` but no clear `classe` clusters. A 3 dimensional PCA analysis did show clustering by `classe`

I trained several different machine learning algorithms, with the following results. Accuracy meausrements are based upon `confusionMatrix` Overall Statistics / Accuracy output. These accuracies reflect a preliminary out-of-sample results because they are taken from data outside the training set. Comparable out-of-sample results are shown in Section 3 using the Quiz set.

```
Algorithm                     Accuracy  Comments
---------------------------    -------  ---------------
 K-Nearest Neighbors (KNN-N0)   0.97    Scale to N(0,1) K = 3
 KNN_Min Max (KNN-MM)           0.94    Scale min to 0, max to 1, k=1
 Naive Bayes                    0.49
 Log.Regression                 0.74
 Linear Discriminant Analysis   0.69
 Partial Least Squared Dis.     0.39
```

To determine the best value for K, the KNN algorithm was looped a number of times. In the data set provided, the best K values were small. I am concerned that this approach may not work with other data sets because it may be very much over-fitting the data. I do not expect that this will be a problem with the unknown TEST data set because it is taken from the same set of users as the training data.

For cross validation, KNN-N0 results were trained on one data set, probed using a test data set and verified on a quiz data set. These results were repeated with comparable results using different seed values to repartition the test and training sets. Validation against the quiz was performed only once.

2. Approach

This secion provides a quick overview of the approach taken to clean the data. To save word count, not all R code is provided in the HTML file. But it is the RMD file.

2a. Read in Data

Raw data was captured on disk and read into the file.

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

2b. Preliminary Exploration and Data Cleaning

An examination of the data shows that the training set had a number of factors that I didn't need to carry around for testing and training. Examples include:

- Col 1 Index (not relevant)
- Col 2 user_name (not relevant)
- Col 3:7 time and window number (not relevant)
- Col 12 kurosis (Almost all blank)
- Col 18 max_roll (almost all NA)
- etc

We removed the less useful columns to simplify and speed analysis.

The following column numbers remain: classe, roll_belt, pitch_belt, yaw_belt, total_accel_belt, gyros_belt_x, gyros_belt_y, gyros_belt_z, accel_belt_x, accel_belt_y, accel_belt_z, magnet_belt_x, magnet_belt_y, magnet_belt_z, roll_arm, pitch_arm, yaw_arm, total_accel_arm, gyros_arm_x, gyros_arm_y, gyros_arm_z, accel_arm_x, accel_arm_y, accel_arm_z, magnet_arm_x, magnet_arm_y, magnet_arm_z, roll_dumbbell, pitch_dumbbell, yaw_dumbbell, total_accel_dumbbell, gyros_dumbbell_x, gyros_dumbbell_y, gyros_dumbbell_z, accel_dumbbell_x, accel_dumbbell_y, accel_dumbbell_z, magnet_dumbbell_x, magnet_dumbbell_y, magnet_dumbbell_z, roll_forearm, pitch_forearm, yaw_forearm, total_accel_forearm, gyros_forearm_x, gyros_forearm_y, gyros_forearm_z, accel_forearm_x, accel_forearm_y, accel_forearm_z, magnet_forearm_x, magnet_forearm_y, magnet_forearm_z

2c. Create Training/Test and Quiz Set for Cross-Validation

Training data is split into three sets

1. 60% Training - Data for machine learning (ML) algorithms
2. 20% Test set - Set to probe / test specific ML algorithms and settings
3. 20% Quiz set - Set to validate algorithms after final selection (out of sample)

This worksheet is set up so that training and test sets can be randomly reassigned by changing a global `SeedVersion` value in the first block. The experiment has been run

several time, shuffeling rows into test and training sets, but leaving the Quiz set as is. Results are not shown but can be reproduced by changing data set.

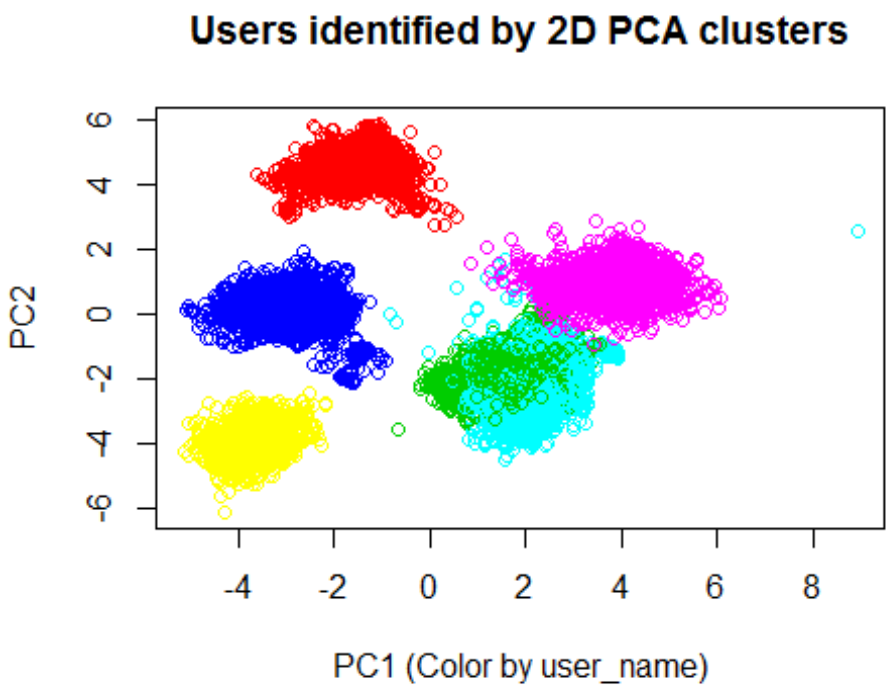
2d. Plot the data for each dimension vs. column and color by class

Initially, I wrote scripts to examine data to see if any clear features popped out. There are no obvious single columns that correlate well with states. However, this approach did identify several rows that have clear outliers: gyros_dumbbell_x

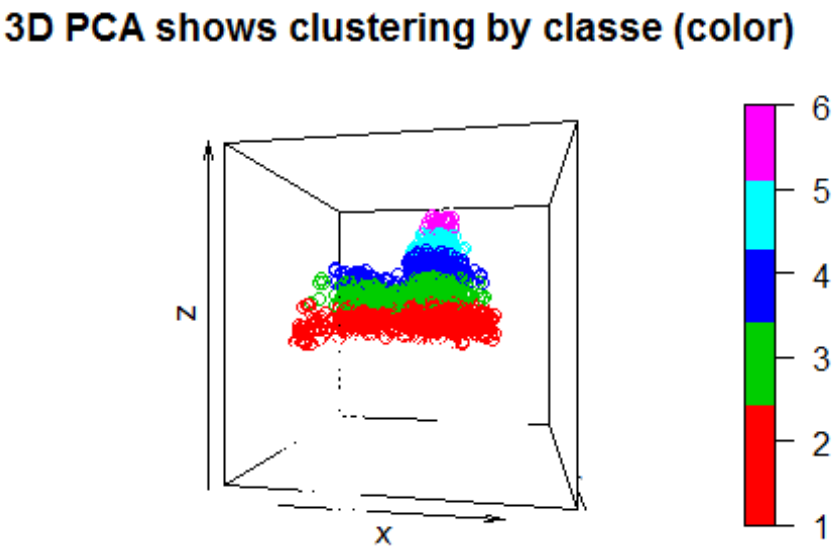
It turns out that using an 2-D PCA analysis will form users into clusters quite readily but the relationship between different classes is not so obvious.

```
set.seed(SeedVersion)
preProc_pca2 <- preProcess(training[,-1], method="pca", pcaComp=2)
    #preProc_pca2
    #preProc_pca2$rotation

trainP_pca2 <- predict(preProc_pca2, training[,-1])
#    plot(trainP_pca2$PC1,trainP_pca2$PC2, col=(as.numeric(train_c)+1),
#         main="classe vs 2D PCA shows significant overlap",
#         xlab="PC1", ylab="PC2" )
    plot(trainP_pca2$PC1,trainP_pca2$PC2, col=(as.numeric(train_u)+1),
         main="Users identified by 2D PCA clusters",
         xlab="PC1 (Color by user_name)", ylab="PC2" )
```



With a 3D PCA, there does appear to finally be a clustering of classe that we can take exploit. Unfortunately, I did not attempt to develop a ML algorithm to take advantage of these clusters



2e. Train classifiers on "Training Data", Evaluate on "Probe Data"

As described above, we evaluated a number of different approaches:

- K-Nearest Neighbors (KNN-N0)
- KNN_Min Max (KNN-MM)
- Naive Bayes
- Log.Regression
- Linear Discriminant Analysis
- Partial Least Squared Dis.

K-Nearest Neighbors (KNN-N0) Results

The approach for KNN is shown below. KNN needs a K to determine how many neighbors to evalue. To determine the best K, we looped over a number of different values and shown in the graphics below. Overall, we had very good results as shown.

```
library (class)
znorm <- function(x) { return ((x - mean(x)) / sd(x))}
```

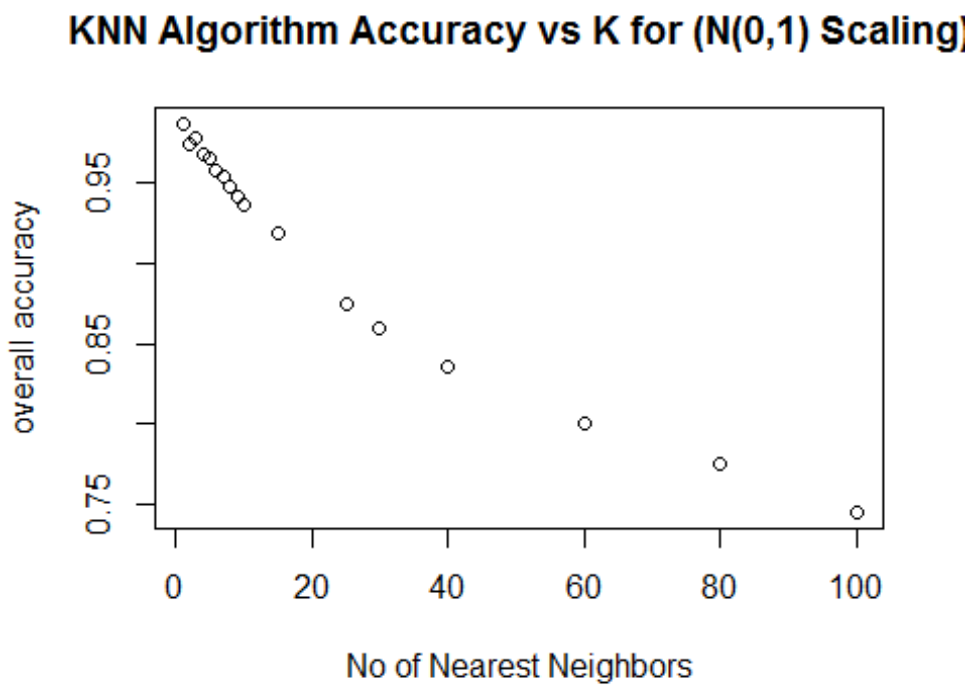
```r
#x<- znorm(rnorm(10,mean=30,sd=5)); mean(x); sd(x)  #Check Code

train_knnZ <- as.data.frame(lapply(training[2:colmax],znorm))
test_knnZ <-  as.data.frame(lapply(testing[2:colmax],znorm))

#Loop over different values of K to see which works best

x <- c(1,2,3,4,5,6,7,8,9,10,15,25,30,40,60,80,100) #values of K
y <- x * 0

set.seed(SeedVersion)
for(nNN in 1:length(x)){
    knn_test_pred <- knn(train = train_knnZ, #Training Set
                         test = test_knnZ,  #Test Data
                         cl = train_c, #Truth Labels for Training Data
                         k=x[nNN]) #Number of valued to compare
    cl_knn<- confusionMatrix(test_c,knn_test_pred)
    y[nNN] <- cl_knn$overall[1]
}
plot(x,y,xlab="No of Nearest Neighbors",  ylab="overall accuracy",
     main="KNN Algorithm Accuracy vs K for (N(0,1) Scaling)")
```




```r
#y
kBest = x[which(y==max(y))]
#set kBest<-3 Anyway because 1 seems overtraining
kBest <-3
```

```
knn_test_pred <- knn(train = train_knnZ, test = test_knnZ,
                     cl = train_c, k=kBest)

cl_knn<- confusionMatrix(test_c,knn_test_pred)
cl_knn
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1100   11    4    0    1
##          B   11  735   12    1    0
##          C    3    8  669    4    0
##          D    0    0   15  627    1
##          E    0    6    3    5  707
##
## Overall Statistics
##
##                Accuracy : 0.9783
##                  95% CI : (0.9733, 0.9827)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9726
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9874   0.9671   0.9516   0.9843   0.9972
## Specificity            0.9943   0.9924   0.9953   0.9951   0.9956
## Pos Pred Value         0.9857   0.9684   0.9781   0.9751   0.9806
## Neg Pred Value         0.9950   0.9921   0.9895   0.9970   0.9994
## Prevalence             0.2840   0.1937   0.1792   0.1624   0.1807
## Detection Rate         0.2804   0.1874   0.1705   0.1598   0.1802
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9909   0.9798   0.9735   0.9897   0.9964
```

```
#cl_knn$overall[1]
#kBest

#try it on the QUIZ SET - But Don't Look Yet
Quiz_knnZ <- as.data.frame(lapply(quizing[2:colmax],znorm))
knnN0_Quiz_pred <- knn(train = train_knnZ, #Training Set
                     test = Quiz_knnZ,  #Test Data
                     cl = train_c, #Truth Labels for Training Data
                     k=kBest) #Number of valued to compare
knn_cl_quiz <- confusionMatrix(quiz_c, knnN0_Quiz_pred)
```

```
#Try it on the FINAL TEST SET - And Save Answers for later output
TEST0_knnZ <- as.data.frame(lapply(df_TEST0[2:colmax],znorm))
knnN0_TEST0_pred <- knn(train = train_knnZ, #Training Set
                      test = TEST0_knnZ,  #Test Data
                      cl = train_c, #Truth Labels for Training Data
                      k=kBest) #Number of valued to compare
```

KNN_Min Max (KNN-MM) Results

```
##  Accuracy
## 0.9459597
```

Naive Bayes Results

```
## Loading required package: e1071

##  Accuracy
## 0.4980882
```

Log.Regression Results

```
## Loading required package: stats4
## Loading required package: splines
##
## Attaching package: 'VGAM'
##
## The following object is masked from 'package:caret':
##
##     predictors

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 26 elements replaced by 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 45 elements replaced by 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 74 elements replaced by 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 79 elements replaced by 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 79 elements replaced by 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control
## $wzepsilon): 79 elements replaced by 1.819e-12

##  Accuracy
## 0.7473872
```

Linear Discriminant Analysis Results

```
##  Accuracy
## 0.7137395
```

Partial Least Squared Dis.

```
library(caret)

train_plsda <- training[,2:colmax]
test_plsda <- testing[,2:colmax]

set.seed(SeedVersion)
fit_plsda <- plsda(train_plsda,train_c, probMethod="Bayes")
pred_plsda <- predict(fit_plsda, test_plsda)
cl_plsda<- confusionMatrix(test_c,pred_plsda)
#cl_plsda
cl_plsda$overall[1]
```

```
##  Accuracy
## 0.4073413
```

3. Results, Cross-Validation, and Out of Sample Error

3a. Out-of-Sample Error / Validate the best algorithm on quiz data

The best algorithm was the KNN algorithm using N(0,1) scaling. We evaluate the results on the quiz data set.

```
knn_cl_quiz <- confusionMatrix(quiz_c, knnN0_Quiz_pred)
knn_cl_quiz
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1105   10    1    0    0
##          B   10  738   10    1    0
##          C    0    5  666   13    0
##          D    0    0   23  618    2
##          E    4    4    3    3  707
##
## Overall Statistics
##
##                Accuracy : 0.9773
##                  95% CI : (0.9722, 0.9817)
##     No Information Rate : 0.2852
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9713
##  Mcnemar's Test P-Value : NA
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9875   0.9749   0.9474   0.9732   0.9972
## Specificity            0.9961   0.9934   0.9944   0.9924   0.9956
## Pos Pred Value         0.9901   0.9723   0.9737   0.9611   0.9806
## Neg Pred Value         0.9950   0.9940   0.9886   0.9948   0.9994
## Prevalence             0.2852   0.1930   0.1792   0.1619   0.1807
## Detection Rate         0.2817   0.1881   0.1698   0.1575   0.1802
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9918   0.9841   0.9709   0.9828   0.9964
```

3b. Stack the Test Results from each model

For curiosity, we compare the results from each of the models.

```
##    knnN0 knnMM bayes vgam lda
## 1      B     E     C    C   B
## 2      A     A     C    A   A
## 3      A     A     C    B   B
## 4      A     A     C    C   C
## 5      A     A     B    A   C
## 6      C     A     A    E   E
## 7      D     D     C    D   D
## 8      D     B     D    E   D
## 9      A     A     A    A   A
## 10     A     A     C    A   A
## 11     D     B     C    C   D
## 12     C     A     C    A   A
## 13     B     B     B    B   B
## 14     A     A     A    A   A
## 15     E     E     E    E   E
## 16     E     E     B    E   A
## 17     A     E     C    A   A
## 18     B     B     B    B   B
## 19     B     B     D    B   B
## 20     B     B     B    B   B
```

3c. Write answers to file

And we write the answers to the files for submission.

```
## [1] "2015-06-14 16:23:22 MST"
```

4. References and Links

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. See http://groupware.les.inf.puc-rio.br/har#ixzz3d4K34xch