Prediction Model for Human Activity Recognition Using Samsung Smartphone

Author: Chandra Srinivasan

Introduction

In the last decade, there has been an unprecedented growth in smartphone use, which offer a barrage of applications. New generations of smartphones include intuitive built-in features that may require the use of a built-in gyroscope or an accelerometer. A gyroscope allows you to change the orientation of the device by rotating its display. An accelerometer, on the other hand, tracks acceleration or senses device vibration.

It is possible to obtain a continuous record of quantitative measurements from these sensors, which in turn might be useful in predicting the activity of the individual carrying the smartphone.

Our goal was to build a prediction model for recognition of human activity using Samsung smartphone.

Methods

Data Collection

The experiments have been carried out with a group of 30 volunteers within an age bracket of 19-48 years. Observations from 21 of these volunteers have been provided for this task. Each person performed six activities (Walking, Walking upstairs, Walking downstairs, Sitting, Standing, Laying) wearing a smartphone (Samsung Galaxy S II) on the waist. The experiments have been video-recorded to label the data manually [2].

The data is made available in public domain at URL: http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones

The processed data for R used for this task is available at URL: https://spark-public.s3.amazonaws.com/dataanalysis/samsungData.rda

Attribute Information

For each record in the dataset it is provided:

- Triaxial acceleration from the accelerometer (total acceleration) and the estimated body acceleration
- Triaxial Angular velocity from the gyroscope
- A 561-feature vector with time and frequency domain variables
- Its activity label (Walking, Walking upstairs, Walking downstairs, Sitting, Standing, Laying)
- An identifier of the subject who carried out the experiment

A detailed description of the features (561) in the dataset is available at URL: UCI HAR Dataset.zip

The data was processed and analyzed using the R programming language [3] on Mac OS X Lion (10.7.5) platform with a 2.66 GHz Intel Core i7 processor (4 cores) and 8 GB of RAM. The R packages `'snowfall'` was used for enabling parallel processing as needed, utilizing 3 out of 4 cores on the local machine [4].

Exploratory Analysis

The activity variable was treated as a factor variable. All the possible predictors (561) were numeric variables. A matrix of pairwise correlations was computed. Principal component analysis (PCA) was performed by singular value decomposition of the centered and scaled training data to assess correlation among predictor variables and the variance among data.

Prediction Modeling

Data splitting

Based on the restrictions assigned for this task, the training set must include the data from subjects 1, 3, 5, and 6. Test set is the data from subjects 27, 28, 29 and 30. More subjects could be included in either set without overlapping.

Two subsets were initially created from the original data set (7352 observations): one with observations (obs) exclusively from subjects 1, 3, 5 and 6 and this formed the ‘training-only’ subset (1315 obs); another one with observations exclusively from subjects 27, 28, 29 and 30 and this formed the ‘testing-only’ subset (1485 obs). A balanced approximately 50/50% split of the remainder of the data (4552 obs) from all the other subjects was achieved by random sampling within each class preserving the overall class distribution. Each of these was added to the 'training-only' and 'testing-only' sets to form the final **'training'** (3593 obs) and **'testing'** (3759 obs) sets respectively. This **testing set** was excluded from all model-building steps.

Modeling Methods

The dimensionality of the data [5] and multi-collinearity [6] was considered in choosing the various modeling methods for this classification problem. The modeling methods chosen were Random Forest (RF), kernel based Support Vector Machine (SVM), k-Nearest Neighbor algorithm (k-NN), and Gradient Boosted Decision Trees (GBDT). 5-Fold cross-validation (5-fold CV) was performed to optimize the tuning parameters for individual models. The final models were used to predict responses on the test set ('hold-out set') and various performance measures were calculated for each of these models. The process is described in more detail below.

Pre-processing

The data was pre-processed by centering (mean subtraction) and scaling (to zero mean and unit variance) for SVM model and k-NN algorithm.

Model Training and Tuning

Random Forests (RF)

RF is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the classes, output by individual trees. The algorithm for inducing a RF was developed by Leo Breiman and Adele Cutler [7]. The method combines Breiman's "bagging" idea and the random selection of features, introduced independently by Ho and Amit and Geman in order to construct a collection of decision trees with controlled variation [8]. The package in R implementing RF needs tuning of parameter m_{try} , which is the number of variables randomly sampled as candidates at each split [9]. The number of trees to grow was left a default value of '500'. The parameters were tuned to minimize out-of-bag error estimate.

Support Vector Machine (SVM)

In machine learning, SVMs are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. SVMs can efficiently perform non-linear classification using what is called the **kernel trick**, implicitly mapping their inputs into high-dimensional feature spaces [10]. The kernel functions take two feature vectors as parameters and return the scalar dot product of the vectors.

`'kernlab'` is an extensible package for kernel-based machine learning methods in R [11]. We used Gaussian Radial Basis kernel function for this task with a multi-class classification formulation ('`spoc-svc` Crammer, Singer native multi-class') [12]. The parameter 'sigma' was tuned by a heuristics described in the package. The parameter 'C' was tuned to minimize 5-fold CV error (figure 1, panel d) at the optimal value of sigma.

k-Nearest Neighbor Algorithm (k-NN)

The k-nearest neighbor algorithm (k-NN) is a method for classifying objects based on closest training examples in the feature space [13].

The parameter 'k' was optimized by an in-built feature of the R package 'RWeka' [14]. RWeka is an R interface to Weka [15], a collection of machine learning algorithms for data mining tasks written in Java.

Gradient Boosted Decision Trees (GBDT)

Gradient boosting is a machine learning technique, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees [16].

This model was built using R packge `'gbm'` [17]. We used a 'multinomial' loss function for our model. The choice of values for parameters shrinkage and number of iterations was based on a balance between computational costs and performance. Shrinkage was left constant at '0.005' while tuning the number of iterations (up to 5000) and interaction depth (values studies: 1, 2 and 3) (figure 1, panel e)

Variable Importance

RF evaluates importance of variables based on a model-based method and ranks them based on mean decrease in accuracy and Gini index (figure 1, panel h). GBDT also allows estimation of relative influence of predictor variables (figure 1, panel g).

Results

The original data included obs from 21 different individuals. After data splitting, both training and test set included obs from 17 different individuals respectively. Out of 561 predictor variables, 431 had mean absolute correlation above 0.70. The first 98 out of 561 principal components accounted for 95% of cumulative variance in the predictor variables (figure 1, panel a, b and c).

Table 1 summarizes the prediction performance of all the models on the test data using a ‘one-against-all other classes’ approach. GBDT, SVM and RF performed exceptionally well with accuracies close to 97%. GBDT was the best model with an accuracy of 97.1%. The performance of k-NN was slightly inferior to others. The comparison of cross-validation estimate and prediction performance in test data did not indicate significant degree of ‘overfitting’.

The top twenty important predictor variables based on RF and GBDT is shown in figure 1, panels g & h.

Table 1: Modeling methods and prediction performance measures on test data

Method		**Random Forest**	**Support Vector Machine**	**k-Nearest Neighbor**	**Gradient Boosted Decision Trees**
R Packages used		`'randomForest'`	`'kernlab'`	`'RWeka'`	`'gbm'`
Pre-processing		None	Centering and Scaling	Centering and Scaling	None
Parameters tuned (Values used in final model)		m_{try} (23)[#] ntree (500) (OOB error: 2.1%)	C (11), Sigma (0.0019)	'k' (1)	Shrinkage (0.005), Interaction depth (3), Number of iterations (4985)
5-Fold CV Error (%)		2.5 (561 predictors)	1.6	4.3	0.04
Accuracy %(Acc) (95% CI)		96.9 (96.3 - 97.4)	97.0 (96.4 - 97.5)	93.2 (92.3 - 94.0)	97.1 (96.5 – 97.6)
P-Value [Acc > NIR]*		< 2.2e-16	< 2.2e-16	< 2.2e-16	< 2.2e-16
Kappa		0.9628	0.9635	0.9179	0.9651
Sensitivity (%)	Laying	99.7	98.4	97.4	99.7
	Sitting	91.6	90.8	82.3	91.9
	Standing	95.1	95.1	89.0	93.7
	Walk	98.7	99.8	97.8	99.5
	Walkdown	98.4	99.6	95.8	99.6
	Walkup	98.9	99.6	99.3	99.6
Specificity (%)	Laying	100.0	100.0	99.9	100.0
	Sitting	98.8	98.9	97.3	98.5
	Standing	98.1	97.6	95.7	98.2
	Walk	99.9	99.9	99.8	100.0
	Walkdown	99.8	100.0	99.8	99.8
	Walkup	99.7	100.0	99.2	99.9
PPV[@] (%)	Laying	100.0	99.9	99.7	100.0
	Sitting	94.4	94.6	86.9	93.2
	Standing	92.3	90.3	82.8	92.4
	Walk	99.3	99.5	99.0	100.0
	Walkdown	98.6	99.8	98.6	98.8
	Walkup	98.0	99.8	95.5	99.4
NPV[$] (%)	Laying	99.9	99.6	99.4	99.9
	Sitting	98.2	98.0	96.2	98.2
	Standing	98.8	98.8	97.4	98.5
	Walk	99.8	100.0	99.6	99.9
	Walkdown	99.8	99.9	99.4	99.9
	Walkup	99.8	99.9	99.9	99.9
Detection Rate[&] (%)	Laying	19.6	19.3	19.2	19.6
	Sitting	16.5	16.3	14.8	16.5
	Standing	18.1	18.1	16.9	17.8
	Walk	15.6	15.8	15.5	15.7
	Walkdown	13.0	13.1	12.6	13.1
	Walkup	14.2	14.3	14.2	14.3

*No-information rate (NIR) 19.7% is the largest proportion of the observed classes. A hypothesis test is computed to evaluate whether the overall accuracy rate is greater than the rate of the largest class.
[@]PPV: Positive Predictive Value; [$]NPV: Negative Predictive Value; [&]Detection Rate: The rate of true events also predicted to be events. [#]Parameter for random forest m_{try} tuned using OOB (out-of-bag) error estimate.

Conclusions

GBDT, SVM and RF models performed exceptionally well for this multi-class classification problem. Despite evidence for significant correlation among predictor variables and hence multicollinearity, there was no evidence for significant overfitting.

A balanced approach preserving the overall class distribution has to be adopted while splitting the data for training. CV error estimates helped minimize overfitting. Tuning of model-specific parameters is important. The long processing time for tuning could be a significant limitation depending on the computer hardware available. This could be abbreviated by use of parallel processing methods enabled by certain packages in R.

References

1. Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine. *International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012.*

2. R Core Team (2013). R: A language and environment for statistical computing. *R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.*

3. Jochen Knaus (2013). snowfall: Easier cluster computing (based on snow).. R package version 1.84-4. http://CRAN.R-project.org/package=snowfall

4. Caruana, R., Karampatziakis, N., & Yessenalina, A. (2008). An Empirical Evaluation of Supervised Learning in High Dimensions. *Proceedings of the 25 th International Conference on Machine Learning, Helsinki, Finland 2008.*

5. Dormann, C. F., et al. (2012). Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography 36(1) pp. 027-046.*

6. Breiman, Leo (2001). "Random Forests". *Machine Learning 45 (1): 5–32. doi:10.1023/A:1010933404324*

7. Wikipedia "Random Forest" Page. URL: http://en.wikipedia.org/wiki/Random_forest

8. Liaw and M. Wiener (2002). Classification and Regression by randomForest. *R News 2(3), 18--22.*

9. Wikipedia "Support Vector Machine. URL: http://en.wikipedia.org/wiki/Support_vector_machine
10. Alexandros Karatzoglou, Alex Smola, Kurt Hornik, Achim Zeileis (2004). kernlab - An S4 Package for Kernel Methods in R. *Journal of Statistical Software 11(9), 1-20. URL http://www.jstatsoft.org/v11/i09/*

11. Wikipedia "k-Nearest Neighbor Algorithm". URL: http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm

12. K. Crammer, Y. Singer
On the learnability and design of output codes for multiclass prolems
Computational Learning Theory, 35-46, 2000.
http://www.cs.huji.ac.il/~kobics/publications/mlj01.ps.gz

13. Wikipedia "K-Nearest Neighbor Algorithm". URL: http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm

14. Kurt Hornik, Christian Buchta, Achim Zeileis (2009) Open-Source Machine Learning: R Meets Weka. *Computational Statistics, 24(2), 225-232. doi:10.1007/s00180-008-0119-7*

15. Ian H. Witten and Eibe Frank (2005). Data Mining: Practical Machine Learning Tools and Techniques. 2nd Edition, Morgan Kaufmann, San Francisco.

16. Wikipedia "Gradient Boosting". URL: http://en.wikipedia.org/wiki/Gradient_boosting

17. Greg Ridgeway with contributions from others (2013). gbm: Generalized Boosted Regression Models. R package version 2.0-8. http://CRAN.R-project.org/package=gbm