Using machine-learning model techniques to predict categories of activities from large numbers of input variables

8th December 2013

1 Introduction :

The rapid evolution of accelerometer and gyroscope technology, coupled with lower prices due to massive economies of scale since the 1990s means that these devices are now a commonplace on most categories of smart phone[**?**]. This enables many new possibilities (e.g.[2]), for the smart phone including data-logging, new functions (pedometer), new forms of human computer interaction (e.g. wave the phone to perform some function). One field where new possibilities are enabled is the study of Activities of Daily Living [3], a discipline, which informs casework and theoretical development in fields of medicine (e.g. physiology), and design (e.g. architecture). Practitioners can use mobile phones as non-intrusive data-logging devices to measure the Activities of Daily Living in subjects. If data logged is converted accurately to activity being performed, this provides valuable data at low unit collection cost. However data produced from the smart phone devices is of considerable scale (100s of variables per observation, 1000s of observations per study) and there are numerous possibilities for error. [1] In this study I set out to address the question : Is it feasible to construct a function which can predict activity solely using quantitative accelerometer/gyroscope data from a smart-phone device? This question was extended to include the error-rate required and achievable, and the implementation characteristics. In this study, results from functions created using SVM, decision tree and nearest neighbour techniques are reported.

The SVM-based function is the recommended choice based on error-rate and favourable model characteristics. Limitations of the study and suggestions for further investigative work are set out.

2 Methods:

2.1 Data Collection

For this analysis I used data from a sample of 7,352 records, based on work published [4] and re-issued by Coursera Data Analysis Class edition 1[5]. The data was provided as part of a class individual assignment, and was loaded and processed using the R programming language [6, 7].

[1] (Consider : there are 2 basic sensors accelerometer and gyroscope measuring on 3 axes each. 6 independent measurements each at 5% error rate means an overall error rate of (1-(1-.95)^6) ~ 26%)

2.2 Exploratory Analysis

Exploratory analysis was performed by examining tables and plots of the downloaded data. This was used to :

1. partition the data into training, validation and hold-out sets (Table 1)
2. identify missing values, verify the quality of the data
3. determine likely approaches to modelling, which might best yield a predictive function

2.3 Statistical Modelling

An important part of predictive modelling is the careful partitioning of available data. A prerequisite of the study is that data from 4 (of the 21) subjects is used for training and data from a different set of 4 subjects is used for testing. Data from additional 13 subjects was randomly allocated resulting in partitioned data as shown in Table 1.

	Name of data-set	Subjects included	No. of observations
1	Training set 1 - mandated (trainset1)	1, 3, 5, 6	1315
2	Training set 2 - random (trainset2)	21, 16, 15, 14, 17	1793
3	Training set 3 - random (trainset3)	25, 7, 11, 22	1354
4	Quiz set - random (quizset)	8, 19, 23, 26	1405
5	Test set - mandated (test)	27, 28, 29, 30	1485

Table 1: Data set partitions used in the model-building and testing

Pairwise combination of the training subsets was used in model ubilding and testing - e.g. models were trained on trainset12 then tested on trainset 3, a next round might be trained on trainset23 and tested on trainset1 and son.

Following background reading [9, 11] and exploratory analysis models from the general class of 'machine learning' were chosen for further investigation. Three model classes were investigated: State Vector machines (SVM)[15]decision trees [10]and Kth Nearest Neighbour[12].

Three distinct modelling phases :

1. Naive modelling where only simple techniques were used to reduce the variables from the full set
2. A round of reduction where ANOVA analysis was applied to chosen subsets to allows selection of more important variables
3. A further round of modelling where these variables are used for model-building. From further results a selected model type (SVM) was chosen, fine-tuned (Cost parameter) and evaluated against the (previously held-out) test set.

During naive modelling the tree models were built with full sets of the highly-unique variables while for SVM and KNN, the same variables with highly-correlated (>0.9) variables were eliminated using caret::findCorrelation. see Figure 1(a) for a typical output.

Serendipitously, it was observed in this round of modelling that using the 'important variables' generated by the rpart::tree function resulted in KNN and SVM models both more efficient and more accurate than with the inital long set of variable.

Noting this, a round of variable pre-selection was carried out using a method derived from discussions with fellow students on Coursera discussion forums[5]. Subsets of the training set data were evaluated using the ANOVA function (stats::aov) and the F-statistic calculated. Using R code, each variable was ranked (F-statistic from aov(var~activity)) for subsets of the data including Sedentary vs Active activity classes, Sitting vs Standing and Walkup vs Walkdown (these latter 2 because initial investigations showed these activities to be the most difficult to predict).The variables with F-Statistics in the top 1%, 2%, 5% and 10% were identified, de-duplicated and used for model building. Investigation of the results showed that variables selected from the top 2% F-Statistic were the most useful and these were used for later modelling. This meant that just 40 variables from the original 561 were used for the later stages of model-building a 14:1 reduction. see Figure 1(b) for a graphical demonstration of the variable reduction achieved by this method.

Models were developed, tuned and cross-validated using 4-fold split of the data training set (see Table 1) to minimise overfitting risk. A bespoke R function was built to measure performance of each model. Models were evaluated for performance using classification correctness, cross-validation measures, and implementation characteristics (speed to compute, implementability, interpretability). For decision-tree modelling the function rpart::rpart was used, KNN modelling class::knn and for SVM modelling kernlab::ksvm. Decision trees were progressively grown and pruned [10], to avoid the risk of overfitting with a value of complexity parameter of 0.02. KNN models were built with a range of K values - values between 10 and 30 were found to yield the most effective model. For the SVM models the default rbfdot kernel was used with type=C-SVC. The Cost parameter was fine-tuned through iterative steps (see Tables 7-8) - in the final model a value of Cost =20 was used.

A simple scaffold function was built in R. This ran selected models initially on the training set sub-sets, iteratively to tune the SVM model using the full training set and finally on the (previously held-out) test data-set. Execution time, model size and complexity and the classification success rate and cross-validation error (K = 3 or 4 depending on the training set size) were reported via this function.

3 Results

The source data contained a total of 563 columns and 7,352 rows; a full codebook is available via [5]. From information provided and validated through basic checks, each observation consists of previously scaled (-1:+1) accelerometer and gyroscope data from a phone for a single activity (one of laying, sitting, standing, walking, walkup,walkdown), from a single subject (21 subjects, identified by numbers 1:30). Column names were re-mapped to eliminate duplicates - a simple scheme appending the column number (e.g. 1,2...561) to the start of the string name. This has the desirable side-effect of easily locating each variable from it's name.

According to information supplied the data had previously been filtered to remove noise and time/frequency transformed. The scale of the data-set meant no attempt was made to examine each row or column in detail. Rather summary statistics were calculated by row and column for the array of numeric variables. Additionally mass plots – where 50 columns at a time had a boxplot of all data printed to a single page were generated. These techniques confirmed that the data had been scaled so that all numerics lay in the range -1 ... +1. No missing data were observed though 13 columns were found to have very low (fewer than 100) unique values over the 7,352 observations (for most other columns the equivalent statistic was over 5,000 unique values). No difference was observed in modelling with or without these columns, and they were not used for model generation.

The numerical data (all columns excluding activity, subject) was pairwise tested for cross-correlation and one of each pair where correlation exceeded [0.9] was marked. Whether these highly-correlated columns (which accounted for 343 of the 561 variables) can be classed as 'confounders' in the conventional modelling sense is a moot point. However it was observed during modelling that better trees were built when including all columns, except for low-unique valued - i.e. 548 variables while better SVM and KNN models were built when also excluding the highly-correlated columns i.e. 205 variables. It also seems highly likely that some form of dimension reduction technique (SVD, PCA for instance) will pay dividends for future model development. See [8] for a discussion on the effects of highly-correlated variables in model selection, tuning and interpretability.

Exploratory data analysis focused on identifying and testing the effect of the correlated and uncorrelated variables. This yielded some promising results on the training data (see Table 4). Background reading ([8, 11]) suggested to develop this approach in several directions : decision-trees State Vector Machine (SVM) and K Nearest Neighbour (KNN) modelling.

SVM transforms regression and classification problems into simple linear form but in a high-dimension space, and utilises a feature commonly known as the "kernel trick" to make computation even with large data-sets straightforward. According to [15] it is

> "among the most popular and efficient classification and regression methods currently available"

With KNN a set of variables provide a 'black box' model, based on assessing similarities in vectors of data – an extension of the K-means clustering approach – which is long-established in the machine learning domain, and has a range of efficient implementation techniques[12]. With decision trees a small number of variables from the problem space provide an interpretable model by turning the classification problem into a series of binary decisions[10].

Background reading also informed the selection of model error measurement. Simple measures (correct classification rate and accuracy) were implemented, though accuracy was not found to be useful (since the high preponderance of 'true negatives', many of which can be ascribed to pure chance makes the metric not useful in practice). Instead correct classification rate was chosen, along with performance characteristics of the function. Considering that a device will be of little practical value if it is not predicting correctly most of the time lead to an a priori benchmark of 75-90% (5 – 6 times better than pure chance). Later in the study the results from [4, 17] were made available and showed that previous workers using an SVM model had achieved 89-95% correct classification rate, with a focus very much on a highly efficient model.

Accordingly for later rounds of function development the SVM model approach was used exclusively, and the criteria for evaluating the function evolved to include a correct classification rate of better than 90% with efficient model-build and predict-time characteristics. Tables 5 and later show representative output from later rounds of modelling.

Modelling results for the final function (tuned SVM model) are summarised in Tables 2 & 3,and the output of the final tuned function is shown graphically in Figure 1(c).

Model built with :

```
library(kernlab)
filter<-ksvm(activity~.,data=trainDataSVM,cross=4,C=20,type="C-svc",kpar="automatic")
predo1SVM<-predict(filter,testDataSVM)
```

	Descriptive Model Name	Class %	CV %	Model Time (secs)	Predict Time (secs)	No. of Vars	Model size (Bytes)
1	SVM with default parameter and Cost =20	94.8	97.0	11.0	0.6	40	701028

Table 2: Final model, SVM trained on full trainsets (incl quizset), Cost =20, evaluated on Test set,

It is noticeable through inspection of the model output (also compare Figures 1(a) and 1(c)) that reported cross-validation error rate (as a success- rather than error rate) is in every case more optimistic than the achieved classification rate. Observing also the 'per subject' variations in error rate seen in the training sets and later in the test set, it seems possible that the reported CV error rates are lower because they are mixing data from several subjects in both the training and the test sets, whereas the models themselves are always built on training and test sets which are disjoint in terms of subject.

	laying	sitting	standing	walk	walkdown	walkup
laying	293	0	0	0	0	0
sitting	0	222	42	0	0	0
standing	0	14	269	0	0	0
walk	0	0	0	220	2	7
walkdown	0	0	0	0	200	0
walkup	0	0	0	6	6	204

Table 3: Confusion Matrix from final SVM model, trained on full trainsets (incl quizset), Cost =20 evaluated on Test Set

4 Conclusions

My analysis has shown it is feasible to develop activity prediction functions from accelerometer and gyroscope data, based on known and accepted machine learning techniques – SVM models – which have

1. excellent classification rate of 95% on the held-out test data set
2. acceptable performance in the range : 84% to 95% correct classification rate across a range of sub-samples
3. levels of cross-validation error which seem to indicate the models will be robust to future data sample from similar studies, provided future data is from a statistically similar population
4. practical implementable characteristics(speed, complexity and size of model)
5. A rational method for variable selection (ANOVA analysis), which may be refined and extended through further research and testing.

These results must be considered bounded in scope to the data-set used, since there is no information available as to it's provenance or the sampling used from a larger population (if any).

Examining the graphs of activity in the training and test sets, it is likely that 'per-subject' variations may play a key role in determining accuracy – subject may be considered a 'confounding variable' - this needs further study.

Time and other constraints meant that many promising techniques for solving classification problems (e.g. SVD, PCA, Random Forest, Neural Networks and so on) could not be explored.

There is little domain-specific information on benchmarks for predictive models of this class (such as the classification error, the speed of performance) apart from[4, 17], making it difficult to tell whether the results obtained are either significantly better than other researchers or of practical benefit.

Other researchers are encouraged to build on these results, addressing in particular the weaknesses noted and extending the ANOVA analysis method for variable selection. An archive containing material required to reproduce the results presented is available at [14] –the data must be sourced separately because of the size of the data-set. Such results will be of interest to practitioners and researchers whose work is either directly or indirectly concerned with studying the Activities of Daily Living. Further studies should be generalised to similar classes of problems in other domains – where categorical information must be obtained quickly and reliably under possibly noisy conditions from large numbers of inputs.

References

[1] Accelerometer and gyroscope price evolution in smartphone. URL : http://www.sensorsportal.com/HTML/MEMS_and_Sensors_for_smartphones.htm Accessed 2013-12-7

[2] For instance this paper describes the use of the phone sensors to detect falls in vulnerable adults URL : http://sensorlab.cs.dartmouth.edu/phonesense/papers/Yavuz-fall.pdf Accessed 2013-12-7

[3] Journal of Gerontology Volume 45, Issue 6 Pp. S229-S237. Measuring the Activities of Daily Living: Comparisons Across National Survey .Joshua M. Wiener et al URL : http://geronj.oxfordjournals.org/content/45/6/S229.short. Accessed 2013-12-7 (abstract only consulted)

[4] D.Anguita, A.Ghio, L.Oneto, X.Parra, J.L.Reyes-Ortiz Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine 4th International Workshop on Ambient Assisted Living (IWAAL), Vitoria-Gasteiz, Spain, 3-5 Dec. 2012.

[5] 5. Coursera Data Analysis Oct – Dec 2013, Prof J Leek : https://class.coursera.org/dataanalysis-002/class/index Note : URL last accessed on 2013-12-8, and may not be available on completion of the course. The course wiki : https://share.coursera.org/wiki/index.php/dataanalysis:Main may continue to be available and requires creation of a separate login.

[6] About R page : http://www.r-project.org/about.html Accessed on 2013-12-7

[7] Project Template provides a framework structure for data analysis in R with separation of the data loading, munging, analysis and reporting steps : http://projecttemplate.net/getting_started.html Accessed on 2013-12-7

[8] Describes various exploratory and tuning techniques for predictive modelling : http://www.jstatsoft.org/v28/i05/paper Accessed on 2013-12-7

[9] Notes on data-mining covering many techniques including clustering, decision-trees and KNN : http://www.stat.cmu.edu/~cshalizi/350/ Accessed on 2013-12-7

[10] This paper describes decision-tree modelling using the rpart package : http://onepager.togaware.com/DTreesR.pdf Accessed on 2013-12-8

[11] This paper describes classification techniques generally including a good explanation of K-Nearest Neighbour : http://statistics.berkeley.edu/classes/s133/Class1a.html Accessed on 2013-3-9

[12] This paper describes some recent advances in making KNN work more efficiently at classification problems : http://link.springer.com/article/10.1007/s10115-004-0191-4 Accessed (abstract only) on 2013-12-8.

[13] 1An informal discussion on pros and cons of decision-tree modelling and various other techniques, including KNN : url : http://www.researchgate.net/post/What_are_pros_and_cons_of_decision_tree_versus_other_classifier_as_KNN_SVM_NN accessed on 2013-3-9

[14] Link to public folder containing zips with reproducible code. NB : includes the original data-file, so the total archive size is ~29MB. URL : http://sdrv.ms/15DM5RW Accessed 2013-12-8

[15] Alexandros Karatzoglou;David Meyer T &Kurt Hornik Support Vector Machines in R ,Journal of Statistical Software April 2006, Volume 15, Issue 9. Accessed through internet 2013-12-7.

[16] David Meyer :Support Vector Machines The Interface to libsvm in package e1071.Accessed at http://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf on 2013-12-7

[17] Davide Anguita, Alessandro Ghia, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz .ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0. Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010. Accessed on 2013-12-7.

	Descriptive Model Name	Class %	CV %	Model Time (secs)	Predict Time (secs)	No. of Vars	Model size (Bytes)
1	KNN with k=12	85.2	93.7	11.2	5.1	205	7354816
2	SVM with default parameter	85.1	97.1	27.1	2.4	205	5614284
3	Tree pruned with CP=0.02	69.4	77.2	4.7	0.3	548	16879568
4	KNN 2 with k=12 and tree-tuned variables	84.2	96.5	1.2	0.5	205	7354816
5	SVM 2 with default parameter and tree-tuned variables	89.4	94.0	2.9	0.4	205	368056

Table 4: Results from naive modelling, trained on trainset12, evaluated on trainset 3

	laying	sitting	standing	walk	walkdown	walkup
laying	254	0	0	0	0	0
sitting	0	228	56	0	0	0
standing	0	0	181	0	0	1
walk	0	0	0	224	22	35
walkdown	0	0	0	3	150	3
walkup	0	0	0	9	15	173

Table 5: Confusion table for naive modelling, trained on trainset12, evaluated on trainset 3

	Descriptive Model Name	Class %	CV %	Model Time (secs)	Predict Time (secs)	No. of Vars	Model size (Bytes)
1	KNN 8 with k=12	92.3	94.4	5.0	2.1	40	7354816
2	SVM 8 with default parameter	94.4	95.6	4.6	0.5	40	678256
3	Tree 5 pruned with CP=0.02	77.1	76.6	0.4	0.0	40	1257352

Table 6: Results from modelling with preselected variable (Top 2 p.c. FStat), trained on trainset23, evaluated on trainset 1

	Descriptive Model Name	Class %	CV %	Model Time (secs)	Predict Time (secs)	No. of Vars	Model size (Bytes)
1	SVM with default parameter and Cost =1	96.5	95.4	11.3	2.9	40	944292
2	SVM with default parameter and Cost =10	98.3	96.5	9.4	2.2	40	731260
3	SVM with default parameter and Cost =100	99.3	96.4	9.3	2.0	40	669452
4	SVM with default parameter and Cost =1000	99.8	96.0	10.5	2.0	40	670172

Table 7: Results from SVM Tuning, trained on full trainsets (incl quizset), evaluated on same set, C varying from 1 to 1000

	Descriptive Model Name	Class %	CV %	Model Time (secs)	Predict Time (secs)	No. of Vars	Model size (Bytes)
1	SVM with default parameter and Cost =1	96.5	95.3	11.2	2.9	40	944668
2	SVM with default parameter and Cost =5	97.9	96.6	9.6	2.4	40	778684
3	SVM with default parameter and Cost =10	98.3	96.6	9.4	2.3	40	741484
4	SVM with default parameter and Cost =15	98.6	96.6	9.3	2.2	40	717292
5	SVM with default parameter and Cost =20	98.7	96.9	9.5	2.3	40	737564
6	SVM with default parameter and Cost =30	98.8	96.6	9.5	2.2	40	728380
7	SVM with default parameter and Cost =40	98.9	96.8	9.3	2.1	40	697092
8	SVM with default parameter and Cost =50	98.9	96.6	9.4	2.1	40	698564

Table 8: Results from SVM Tuning, trained on full trainsets (incl quizset), evaluated on same set, C varying from 1 to 1000