Title: Human activity prediction using smartphone sensor data

Introduction:

Many interesting applications have been made possible as more and more sensors are integrated into smartphones. One of them is applications that help people monitor their daily life activities, e.g. the quantified-self movement [1], in other to be more self-aware of him/herself, and thus be able to make more well justified decisions based on quantitative data.

In this study we look on how to perform prediction of typical human activity based on smartphone sensors measurements. Such prediction constitutes an essential element of any applications that benefit from data produced by sensors embedded in a smartphone.

Methods:

Data Collection

The data for our analysis consist of 7,352 accelerometer and gyroscope measurements from 21 volunteers. The sensors are embedded in a Samsung Galaxy SII smartphone, which the volunteers wear on their waists. The experiments with the volunteers have been video recorded in order to label the with activity labels, which consist of: LAYING, SITTING, STANDING, WALK, WALKDOWN, and WALKUP. The data were downloaded from the Coursera Data Analysis web page [2]. The original raw data and pointer to the original paper studying the problem can be found on [3].

Exploratory Analysis

Exploratory analysis was performed by examining tables and plots of the observed data. Exploratory analysis was used to (1) identify missing values, (2) verify the quality of the data. We discovered that the data are already in a good shape, with one observation per row, and each observation labeled by subject ID and activity type. No missing values have been found.

Statistical Modeling

To relate activity type to sensor measurements we fit the data to a combined Random Forest and Support Vector Machine (SVM) classifiers [4]. We use all raw features/variables from the data without any processing (i.e. no scaling, no logarithmic transformation, and no derived features were considered).

Reproducibility

All analyses performed in this manuscript are reproduced in Python programming language, developed on IPython notebook environment[1].

Results:

The data used in this analysis contain 561 variables corresponding to accelerator and gyroscope measurements, e.g. mean/max/standard deviation of acceleration on X-, Y-, and Z-axes.

We split the data into training and test data sets. The assignment asks that subjects 1, 3, 5, and 6 to be in the training dataset, and subjects 27, 28, 29, and 30 in the test dataset. Since no specific requirements on other subjects, we decide to include them in the training set, in order to have as much training data as possible.

In addition to fitting the data to the model, we need to select a set of parameters, the so-called *hyperparameters*, which control the performance of the model with respect to prediction accuracy and its ability to generalize beyond the training data (i.e. avoiding overfitting). We use a simple grid search algorithm to choose the optimum set of hyperparameters. In order to do this, we further split the training dataset into training and cross-validation datasets. We choose to use a k-fold (k=3) cross-validation strategy, where at each fold a random subset of the training dataset is selected as the training and cross-validation subsets.

At each point of the hyperparameter grid (i.e. for every combination of hyperparameters), the classifier model is trained using the training dataset, and its performance is evaluated on the cross-validation dataset. In k-fold cross-validation this step is repeated k times, where at each fold, a different subset of training and cross-validation subset are randomly selected. The model's parameters and its performance are then obtained as the average of the k-fold cross-validation steps. These are then repeated for each possible combination of hyperparameters.

The overall dataset splitting and the k-fold cross-validation steps are illustrated in Figure 1.

[1] The notebook will be made available on GitHub (https://github.com/herrfz/py-da) after the submission deadline, either for online viewing or to reproduce the analysis.

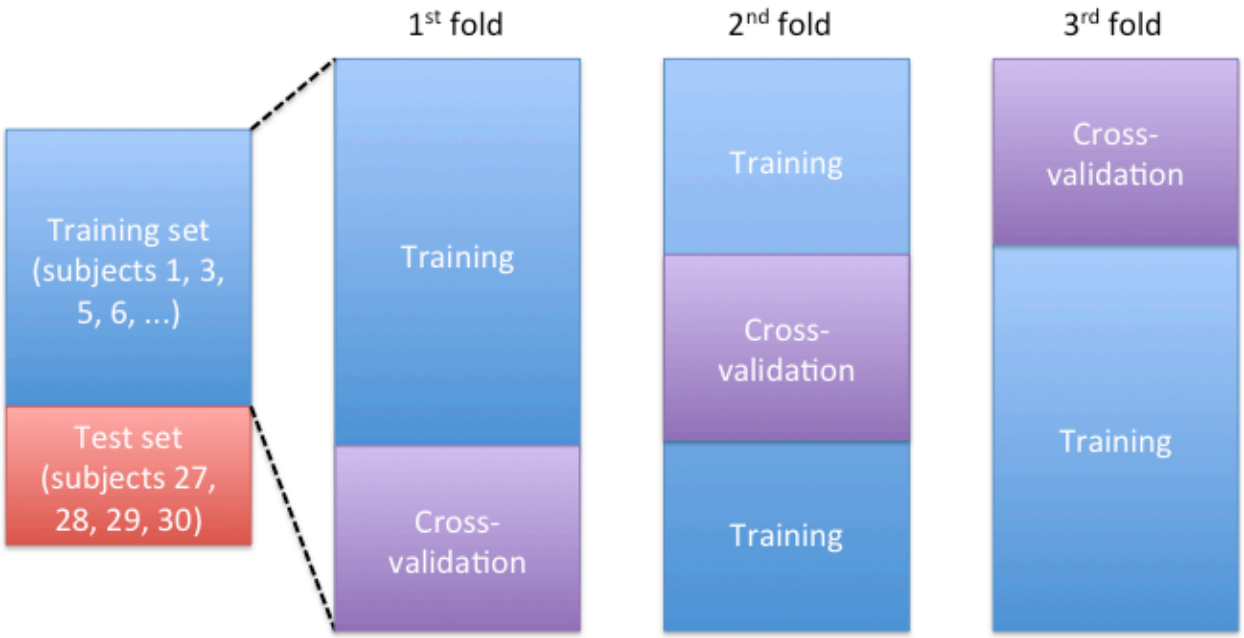

Figure 1: Dataset partition into training, cross-validation, and test datasets

We use the test dataset solely to report the prediction performance. We use the metric *prediction accuracy*, which is computed as:

$$A = \frac{1}{N_T} \sum_{i=1}^{N_T} 1(\hat{y}_i = y_i)$$

$1(\cdot)$ denotes an indicator variable that takes a value of 1 if the expression inside the parenthesis evaluates to True, and 0 if it's False. N_T indicates the number of samples in the training set. $\hat{y}_i$ and y_i indicate the predicted and actual activity value for each sample in the test set, respectively.

Combining predictions from several models generally produces a better prediction performance compared to prediction of one model only. This is the approach that we take in this assignment. Specifically we will use two Random Forests and five SVMs, and perform majority voting from all the seven models to get the prediction. The rationale of selecting seven models is because we have six different activities in the dataset (LAYING, SITTING, STANDING, WALK, WALKDOWN, WALKUP), and by the pigeonhole principle [5], one needs at least seven predictions in order to guarantee that there is always a majority, i.e. more than one vote for a class. The partition between the number of Random Forest and SVM models is currently a rather arbitrary choice, mainly due to the fact that an SVM is faster to train and thus cheaper to create. Whether different partitions could lead to different (i.e. better) prediction performance, is subject to future study.

To get the most out of the data, we do not partition them further to train the different models. Instead, we use the original data to train one of each Random Forest and SVM model, and use bootstrapping [6], i.e. randomly selecting N_{train} from N_{train} number of training data, with replacement, to get new data for training the rest. This is illustrated in the following figure.

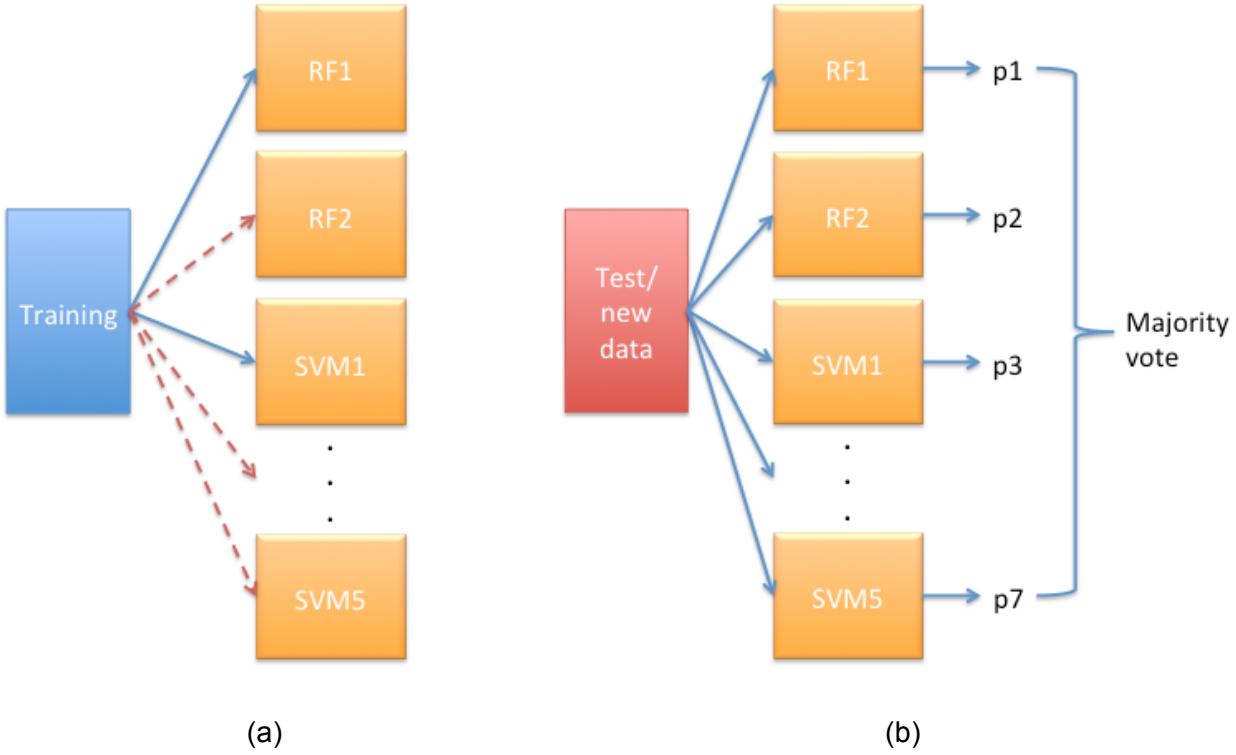

Figure 2: (a) Datasets used for training of the different classifiers, blue line: original dataset; red, dashed line: bootstrapped dataset, (b) Test / prediction of new data and majority vote between the classifiers

The resulting classifiers consist of two Random Forests, each of 500 trees, and five SVMs, with regularization parameter *C* between 10 and 1000, and Radial Basis Function (RBF) kernel with *gamma* parameter between 0.1 and 0.001. The ten most important features obtained from the Random Forest algorithm is shown in the following figure.

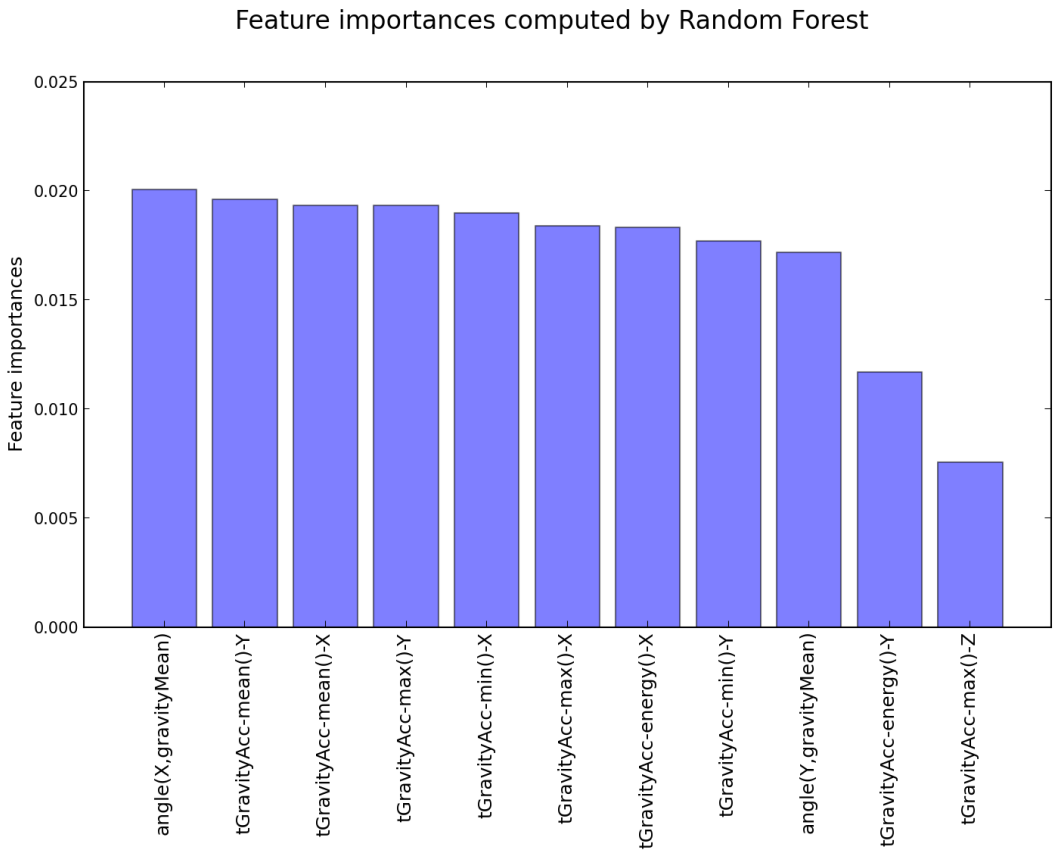

Figure 3: Feature importances computed by Random Forest

Note that training these different models is a computationally expensive, but embarrassingly parallel operation [7], because we have to perform hyperparameter grid search and k-fold cross-validation, as described in the previous section, for each of the models.

Using this technique, we get a **classification accuracy of 0.9791 on the test dataset**. Looking at the results in more detail, we found that all misclassified predictions are for the activities SITTING and STANDING. Separating these two classes is proven to be challenging, as shown in the figure below, where we perform a Principal Component Analysis (PCA) to get the first two principal components of the feature vectors of the training samples having SITTING and STANDING as labels.

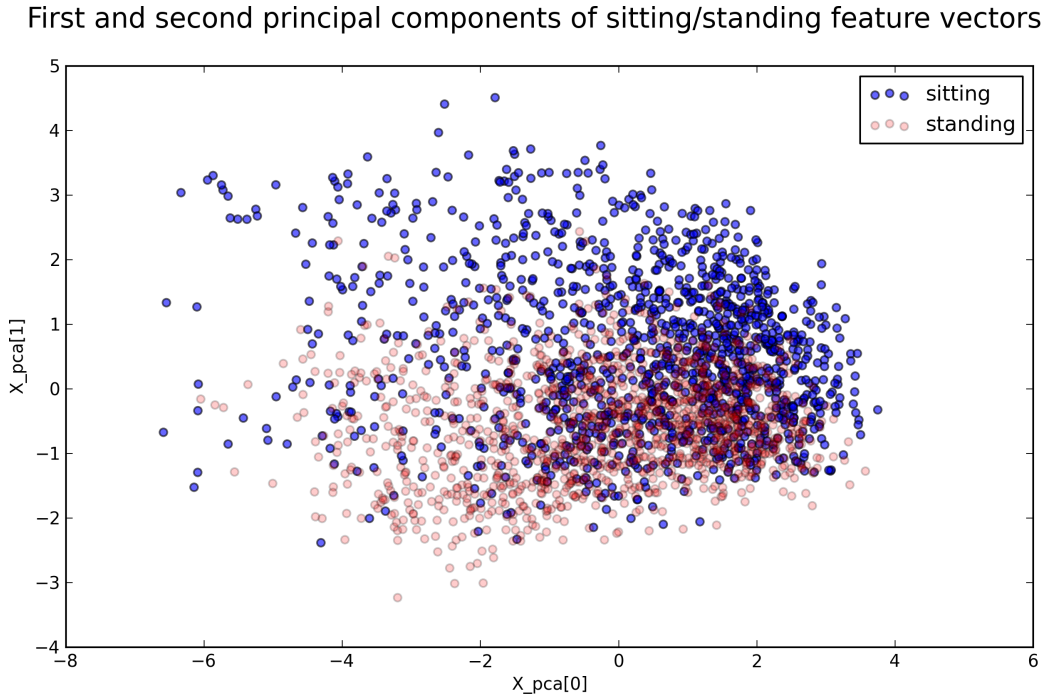

Figure 4: Scatter plot of first and second principal components of sitting/standing feature vectors

The figure shows that, using only two principal components, there is no clear grouping between SITTING and STANDING activities. We believe that more feature engineering is needed to separate these two activities more clearly, and thus improve the prediction performance even more.

Conclusions:

We have designed and implemented a classification model using a combination of Random Forest and SVM with majority voting. The resulting combined classifier achieves 0.9791 prediction accuracy on the test set. Improvement of the classifier performance might be achieved by further feature engineering, concentrating on devising features (or a derivation/transformation thereof) that separate better the SITTING and STANDING activities.

References

[1] Wikipedia page on Quantified Self. URL: http://en.wikipedia.org/wiki/Quantified_Self. Accessed: 10.03.2013.

[2] Coursera Data Analysis assignment 2 page. URL: https://class.coursera.org/dataanalysis-001/human_grading/view/courses/294/assessments/5/submissions. Accessed: 10.03.2013

[3] UCI Machine Learning Repository, Human Activity Recognition Using Smartphones Data Set page. URL: http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones. Accessed: 10.03.2013

[4] Trevor Hastie, Robert Tibshirani and Jerome Friedman (2011). *The Elements of Statistical Learning*, Springer. ISBN 978-0-387-84857-0.

[5] Wikipedia page on the Pigeonhole Principle. URL: http://en.wikipedia.org/wiki/Pigeonhole_principle. Accessed: 10.03.2013

[6] Wikipedia page on Bootstrapping (statistics). URL: http://en.wikipedia.org/wiki/Bootstrapping_(statistics). Accessed: 10.03.2013

[7] Wikipedia page on Embarrassingly Parallel computation. URL: http://en.wikipedia.org/wiki/Embarrassingly_parallel. Accessed: 10.03.2013