Homework #3: Scoring

PPOL 670

The introduction of wearable technologiest such as Fitbit, Jawbone, iPhones, and Androids has opened new possibilities in better understanding physical activity. At the core of these devices is a bundle of sensors, including an accelerometer and gyroscope that measures gravity, user acceleration, among other things. These measures in turn enable the capability to track physical activity and provide users with insight into their daily activities. The implications of these devices are far reaching: it allows for one to understand their lifestyle better and perhaps the physical stresses [or lack] in an occupation.

The key value proposition of activity trackers is insight at scale. From a relatively small training records, it's possible to build a model to *score* acceleration data as it streams. As data is converted from $\frac{m}{s^2}$ to discrete types of activity, the data can be used to understand energy burned among other things.

The dataset. You've been provided with two datasets: A training set labeled "training" ($n = 35486$) and a "test" ($n = 311174$) . Both datasets contain accelerometer measurements from an Apple iPhone 5 and collected for one person. Data was collected using the PowerSense app at a sampling rate of 20 Hertz (e.g. 20 measurements per second). Thus, these accelerometer readings have internal validity with limited external validity. The data provided contains the following fields:

- `id`. Record ID.
- `time`. Time index in 0.05-second increments.
- `user_acc_x.G.`. X-axis user acceleration in $\frac{m}{s^2}$
- `user_acc_y.G.`. Y-axis user acceleration in $\frac{m}{s^2}$
- `user_acc_z.G.`. Z-axis user acceleration in $\frac{m}{s^2}$
- `accel`. The total acceleration as calculated as $\sqrt{\text{accel x}^2 + \text{accel y}^2 + \text{accel z}^2}$.
- `avg50`, `max50`, `min50`, `sd50`. Lagging moving window statistics for last 50 observations relative to time t. For example, `avg50` is the average `accel` value for the past 50 records
- `avg100`, `max100`, `min100`, `sd100`. Lagging moving window statistics for last 100 measures relative to time t. For example, `avg100` is the average `accel` value for the past 100 records.
- `diff100`, `diff50`. The difference between the max and min value. For example, `diff100` is calculated as max100 − min100.
- `activity`. Label for type of activity divided into four categories: `walk`, `run`, `idle` (e.g. sit or stand), and `stairs`.

Acceleration on the X, Y, and Z-axes are illustrated in the diagram below. It is worth noting that total acceleration only preserves magnitude, but not physical direction. This may be appropriate as the position at which the phone is held may change depending on activity (e.g. hand held, in pocket, etc.).

The specific task. Train any classification model (e.g. logistic regression, KNN, Random Forest, Decision Tree) of your choosing on the `training.csv`, then use that model to predict the activity in `test.csv`.

Tips.

- Divide your training set into either a 70-15-15 partition or k-folds.
- When predicting labels using the `predict()` method, be sure to specify `type = "class"`. To get the probabilities, use `type = "prob"`.
- Libraries you may consider include `class` for kNNs, `rpart` for decision trees, `randomForest` for Random Forests, and regular R for `glm` regression.

Submission.

- You will be given a `.R` script template and your submission will be a functioning R script. This script will be run by class instructors and the output of which will be graded automatically using a grading script. Save this `.R` file following this naming convention "firstname-lastname-hmwk2.R". For example: a file for Bill Clinton would be "bill-clinton-hmwk2.R".

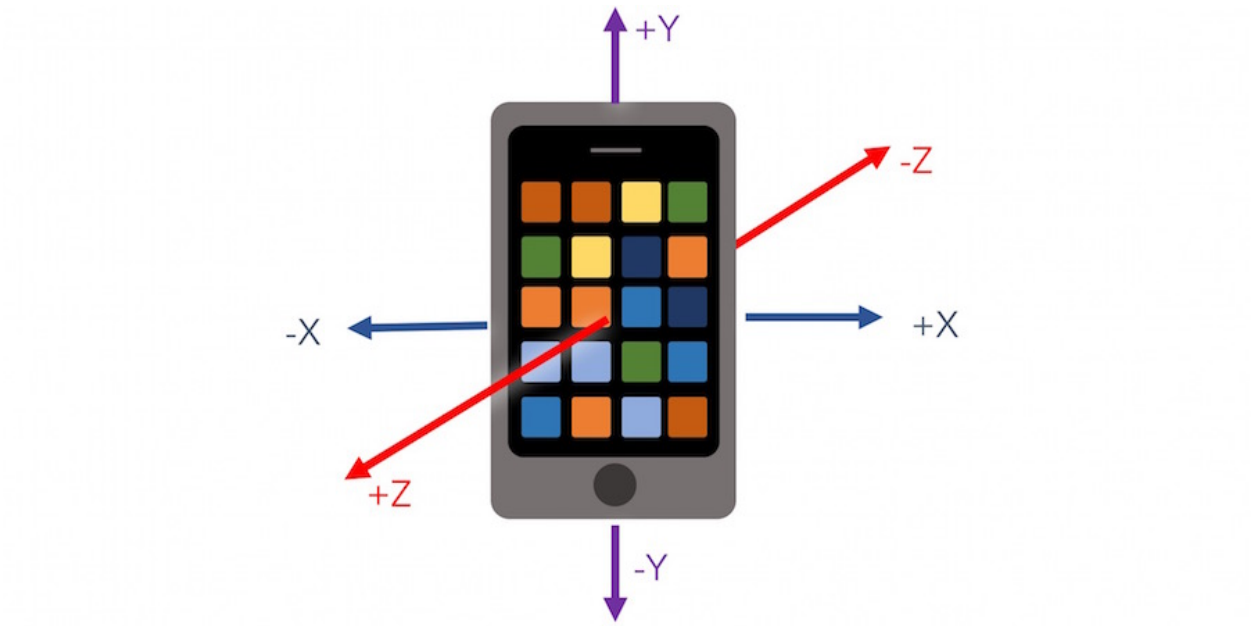

Figure 1: Axes of accelerometer

- Your code should generate a data frame of predictions with $n = 311174$ records and labeled `myPredictions`. The data frame should have the following fields labeled in this manner: `id` and `activity` (your prediction). Your grade will be based on the out-of-sample accuracy of `test`. Again, you should label your the field with your predictions as `yhat` in the `myPredictions` dataframe.
- Accuracy metric. Your accuracy score will be based on the *Mean F1-Score* of your out-of-sample predictions. A F_1 in for a binary problem is formulated as:

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

  where precision $= \frac{TP}{TP+FP}$ and recall $= \frac{TP}{TP+FN} = \frac{TP}{Actual(+)}$. The score is out of 1, where a value of 1 is a very well-balanced prediction.
- The True Positive (TP) is the count of cases where the actual positive ($Y = 1$) is accurately predicted.
- The True Negative (TN) is the count of cases where the actual positive ($Y = 0$) is accurately predicted.
- The False Positive (FP) is count of cases where the actual label was $Y = 0$, but the model classified a record as $\hat{Y} = 1$. This is also known as Type I error.
- The False Negative (FN) is count of cases where the actual label was $Y = 1$, but the model classified a record as $\hat{Y} = 0$. This is also known as Type II error.

The Mean F1-Score is the weighted by each activity class k (e.g. run, walk):

$$\text{Mean } F_1 = \sum_{\text{i}}^{\text{k}} (\frac{n_k}{n} F_{1k})$$

Scoring. 8 points for accuracy (see below). 2 points if your code runs without error.

- Mean F1 > 0.85: 8 points
- Mean F1 > 0.8: 7 points
- Mean F1 > 0.7: 6 points
- Mean F1 > 0.6: 5 points