Project II: Accelerometer Biometric Competition

PH 244 Big Data: A Public Health Perspective

Rachael Phillips

March 14, 2018

Summary

From showing your current speed in a driving app, to switching apps from portrait to landscape, the accelerometer is one of your phones most important sensors. Accelerometers are responsible for axis-based motion sensing and, in addition to having an application in consumer electronics; medical devices, weather sensors, and collision detection systems also use accelerometers. Since everyone moves differently and accelerometers are fast becoming ubiquitous, it has become of recent interest to investigate the feasibility of using accelerometer data as a biometric for identifying users of mobile devices. In this work, we explore the utility of motion-based user identification with data collected from 387 different devices. This unique dataset is made available the Kaggle Competition Question, *Accelerometer Biometric Competition* [1]. We utilize the predictive power of cross-validated ensemble machine learning to shed light on the ability to use this motion-based data for user identification. Additionally, we apply variable importance measures to seek the variables that are most predictive of device identification.

1 Questions

In this report, data from a Kaggle Competition is used to show that accelerometer smartphone data can be used to identify smartphone devices. The company hosting the Kaggle Question, Seal Mobile ID Inc., collected the accelerometer data over a period of several months during normal device usage via an app on Googless Android PlayStore. Seal's *Accelerometer Biometric Competition* Kaggle Contest [1] asks competitors to recognize users of mobile devices from the accelerometer data. We proceed to answer this Kaggle question using the training dataset and we propose an additional question. Which accelerometer-related variables are the most predictive of device identification?

2 Data Overview

The training dataset contains just 5 variables (See Table 1 to view the first two rows of the raw data). Four of these variables (`X`, `Y`, `Z`, and `T`) are response variables and the explanatory variable is `Device` (the unique Id of the device that generated the samples). `X`, `Y`, and `Z`, represent the tri-axial data (3-axis coordinate system) and the values are the acceleration measured in G namely, the acceleration due to gravity along each axis. The other response variable, `T`, is the Unix time (a common system for encoding a date and time) in milliseconds since 1/1/1970. There are roughly 30 million entries of these 5 variables! That means, there are about 75 thousand accelerometer readings on average for each of the 387 devices!

T	X	Y	Z	Device
1336645068311	0.3405087	8.308413	4.1405845	7
1336645068531	0.38136974	8.390134	4.2495475	7

Table 1: The first two rows of the raw training data

We use the rich information within the `X`, `Y`, and `Z` variables, to 1) predict the explanatory variable, `Device`, and 2) explore the most valuable information in predicting `Device`.

Note: Because the test dataset is designed to test Kaggle competitors predictive models, it does not have the corresponding `Device` IDs. So, we did not use the test dataset in our analysis. We felt justified in making this decision because the sample size in the training dataset is so large.

3 Data Processing

According to the article, *Person Recognition using Smartphones Accelerometer Data* [2], it is common to extract both time and frequency domain features from tri-axial accelerometer data. The Fast Fourier Transform (FFT) was utilized to extract the frequency domain [3] for the axial data using the `fft` function in the `stats` R package on CRAN [4] (See variables `FFT_X`, `FFT_Y`, and `FFT_Z` in Table 2). Table 2 shows the processed tri-axial data and considers the same two rows as 1.

Table 2 shows us that, for $\delta \in \{$`X`, `Y`, `Z`$\}$, FFT_δ is a complex number. The complex numbers in the FFT result are simply 2 real numbers, which are both required to give the 2D coordinates of a result that has both a length and a direction angle (or a magnitude and a phase, See Figure 1). The a real number and b imaginary number of a complex number can be converted to magnitude by calculating $\sqrt{a^2 + b^2}$ and to phase by calculating $tan^{-1}(b/a)$. We will consider converting the FFT data to its magnitude and phase components after some dimensionality reduction.

Device	X	Y	Z	FFT_X	FFT_Y	FFT_Z
7	0.3405087	8.308413	4.1405845	21462327+0i	97558596+0i	140615682.1+0.0i
7	0.38136974	8.390134	4.2495475	-7411242+3617477i	7899050- 917063i	-491912.8+1471960.0i

Table 2: The first two rows of the processed training data only considering the tri-axial transformations

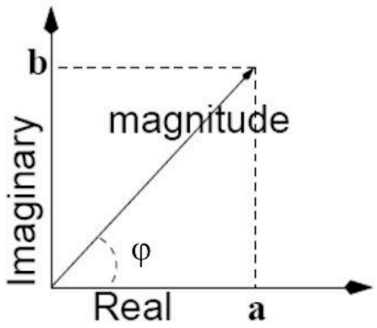

Figure 1: Magnitude-phase representation of complex numbers

3.1 Device Specific Summary Measures

For each device (i.e. user), several device specific summary measures were calculated based on [2]. Using `dplyr` functionality, the device specific mean and median values were calculated for the following variables: `X`, `Y`, `Z`, `FFT_X`, `FFT_Y`, and `FFT_Z`. This gave us a set of 6 mean values and a set of 6 median values for each device. Also, 4 device specific cross-correlations were computed. The z axis remains constant for almost all possible orientations of the smart phone so it is the frame of reference, and the ratios are taken with respect to z axis. Specifically, `Corr_XZ` = `mean_X` / `mean_Z`, `Corr_YZ` = `mean_Y` / `mean_Z`, `Corr_FFT_XZ` = `mean_FFT_X` / `mean_FFT_Z`, and `Corr_FFT_YZ` = `mean_FFT_Y` / `mean_FFT_Z` were calculated for each device separately.

To avoid running into issues with complex numbers, the 8 device specific measurements corresponding to the FFT results were converted to a magnitude and a phase with `Mod` and `Arg`, respectively (base R functionality). As an example, the `mean_FFT_X` $= 1.203096 - 18.486245i$ for `Device` $= 7$ is split into a magnitude component, `mean_M_X` $= 28.130278$, and a phase component, `mean_P_X` $= -0.7170519$. Lastly, all 24 of the device specific summary measures (`mean_X`, `median_X`, `mean_Y`, `median_Y`, `mean_Z`, `median_Z`, `Corr_XZ`, `Corr_YZ`, `mean_M_X`, `mean_P_X`, `median_M_X`, `median_P_X`, `mean_M_Y`, `mean_P_Y`, `median_M_Y`, `median_P_Y`, `mean_M_Z`, `mean_P_Z`, `median_M_Z`, `median_P_Z`, `Corr_M_XZ`, `Corr_P_XZ`, `Corr_M_YZ`, `Corr_P_YZ`) were merged together via `join` in the `plyr` R package to obtain a final, 24×387 data frame. Table 3 shows a preview of 9 of the measures for Devices 7 and 8.

Device	mean_X	median_Y	mean_Z	Corr_YZ	Corr_XZ	median_M_X	mean_P_Y	Corr_P_XZ
7	1.017466	6.851035	4.759725	1.165305	0.2137658	8423.444	-1.283341	0.7931315
8	0.1274531	4.249548	7.666879	0.4569251	0.016623862	13873.818430	-1.47986460	2.503847649

Table 3: Two of the device specific summary measures only considering 9 measures from the set of 24

4 Data Analysis

4.1 Winning Methods

The winner, Dmitry Baranov, describes his method in [5]. Baranov won by exploiting the design flaws in the data that yield useful information for evaluating the hypothesis. First, Baranov identified pairs of sequences (a, b) where the time interval t between the first timestamp of b and the last timestamp of a were between 0 and 1000 ms. Baranov then calculated several "key" features and probabilities based off of the (a, b) pairs of time sequences. Next, Barnov uses these features and a synthetic set of randomly chosen sequences and consecutive sequences to train a logistic regression model. Baranov refines the logistic regression model with some filtering (See Section 2.2 and 2.3) and ends up with chains of consecutive sequences (i.e. $D_1, D_1, D_2, D_1, D_3, D_1, D_1, D_4$). From these chains, guessing the real device is often simple. The probability of "real device of the chain referenced above is D_1" is calculated using Bayes rule. Baranov performs a few additional computations (differentiation between device specific discrete time readings and fitting a random forest classifier) to refine the model further and increase an AUC of 0.9975 to 0.999. The main idea of Baranov's approach is the construction of chains of consecutive sequences using timestamp leakage.

4.2 Novel Methods

We used 10-fold cross-validation with `SuperLearner` [6] to predict the device (i.e. user) with the device specific summary measures. SuperLearner is an ensemble machine learning algorithm that uses cross-validation to estimate the performance of multiple machine learning models. It creates an optimal weighted average of those models, using the test data performance (created internally by the folds) and a specified loss function. This approach has been proven to be asymptotically as accurate as the best possible prediction algorithm that is tested [7]. Our `SuperLearner` library contained a few robust algorithms that can handle multinomial outcomes - `SL.ranger`, a faster implementation of random forest [8], `SL.nnet`, single-hidden-layer neural network [9], and `SL.rpart`, classification and regression trees (CART) [10]. Also, both CART and Random Forest were utilized in [2]. We choose the negative log-likelihood as the loss function for the multinomial outcome prediction of the user.

4.2.1 Predictive Performance

In this section, we define the metric used to measure the performance of our SuperLearner classifier. The predictive performance is measured as the 10-fold cross-validated risk estimate for SuperLearner where the risk is based on the negative log likelihood loss. The results are shown in Table 4. The 10-fold cross-validated risk estimate for SuperLearner is quite low.

Algorithm	Ave	Min	Max
SuperLearner	0.001	0.001	0.001
Discrete SL	-5.268	-5.284	-5.245
SL.nnet	-5.268	-5.284	-5.245
SL.rpart	-5.116	-5.302	-4.948
SL.ranger	-5.166	-5.368	-5.039

Table 4: SuperLearner performance, 10-fold cross-validated risk estimates based on negative log-likelihood

4.2.2 Variable Importance Measures

We used the `randomForest` function and R package and the `varImpPlot` function to see which variables are most important in predicting the device. The Gini plot (See Figure 2) incorporates the mathematics behind decision trees, but essentially measures how pure the nodes are at the end of the tree, testing to see the result if each variable is taken out and a high score means the variable was important. Interestingly, the frequency domain data (or, magnitude and phase data) are consistently more predictive than the time domain data. The median values of the X and Y components in the frequency domain are the most important variables in predicting the user.

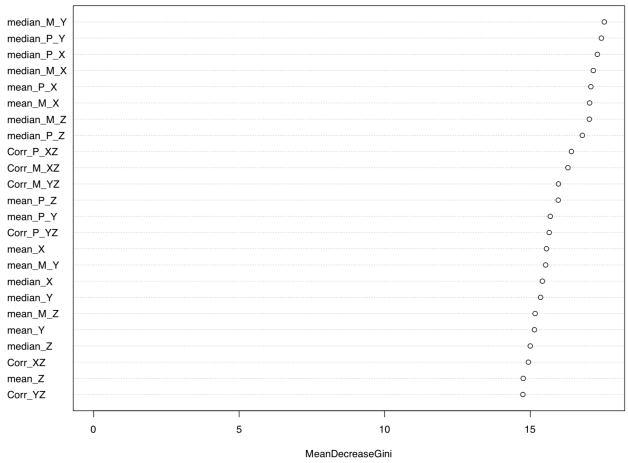

Figure 2: Gini importance measures plot

5 Discussion

In this work, we explore the feasibility of using accelerometer data as a biometric for identifying users of mobile devices. We show that, with low risk of predictive error, accelerometer data can indeed be used to identify smart phone users. We extrapolated about 20 measures from the accelerometer data and we found that the frequency domain data were consistently more predictive than the time domain data. Accompanying this report is complete repository of the R code, tables, and results [11].

References

[1] Accelerometer Biometric Competition. `https://www.kaggle.com/c/accelerometer-biometric-competition`, 2013.

[2] Thingom Bishal Singha, Rajsekhar Kumar Nath, and AV Narsimhadhan. Person Recognition using Smartphones' Accelerometer Data. *arXiv preprint arXiv:1711.04689*, 2017.

[3] Douglas F Elliott and K Ramamohan Rao. *Fast transforms algorithms, analyses, applications*. Elsevier, 1983.

[4] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[5] Dmitry Baranov. Accelerometer biometric competition: description of algorithm. `https://kaggle2.blob.core.windows.net/forum-message-attachments/9976/gaddawin-description.pdf`.

[6] Eric Polley, Erin LeDell, Chris Kennedy, and Mark van der Laan. *SuperLearner: Super Learner Prediction*, 2017. R package version 2.0-22.

[7] Eric C Polley and Mark J Van der Laan. Super learner in prediction. 2010.

[8] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[10] Terry M Therneau, Elizabeth J Atkinson, et al. An introduction to recursive partitioning using the RPART routines.

[11] Rachael Phillips. Big Data Project II: Accelerometer Biometric Competition. `https://github.com/rachaelvphillips/ph244-big_data`, 2018.