Capstone 2: Milestone Report 2

Problem: Can we detect someone's sobriety/drunkenness by their movement?

Client: Public health organizations, sociologists, universities, younger people. Potential to create app that warns users of heavy alcohol use.

Data: UCI Detecting Heavy Drinking Dataset
https://archive.ics.uci.edu/ml/datasets/Bar+Crawl%3A+Detecting+Heavy+Drinking

Data Description: Accelerometer data for 13 participants involved in a "bar crawl" event. CSV file includes position in 3 axes and time of measurement in milliseconds. Additionally, separate CSV files for each participant of TAC (transdermal alcohol content) readings, taken roughly every half hour. "Clean" readings are shifted back in time by 45 minutes to account for time it takes to release alcohol through the skin.

Data Wrangling: The following steps were taken:

1) The first two rows of the accelerometer data were dropped, as they contained all zero values and therefore appear to be a data entry error.

2) Time measurements for accelerometer data are taken in milliseconds since UNIX Epoch (01/01/1970). Values converted to correct datetime objects using the pandas .to_datetime function.

3) TAC readings were concatenated into a single dataframe by looping over the unique participant ID's. Here, time measurements are changed to datetime objects as well.

4) TAC readings for two participants are found to be identical, indicating a data entry error. One participant is dropped.

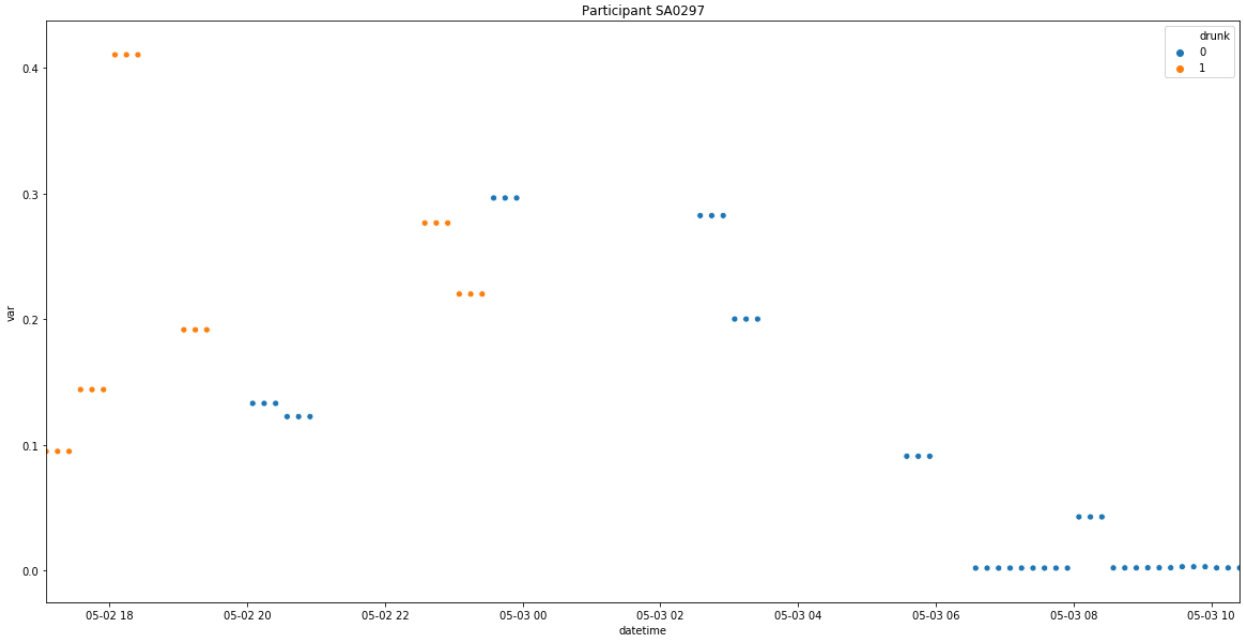
5) Variance is calculated from accelerometer data.
    a) Loop over readings for an individual participant.
    b) Take average of TAC readings within a thirty minute window. Because readings happen roughly every half hour, usually only one or two readings are aggregated.

c) For every one TAC window, there are 3 sub-windows, each representing the variance in movement over a ten minute period.
d) If the TAC reading is greater than or equal to 0.08 (legal limit), label row as “drunk” (1 in “drunk” column).
e) Loop over all participants.

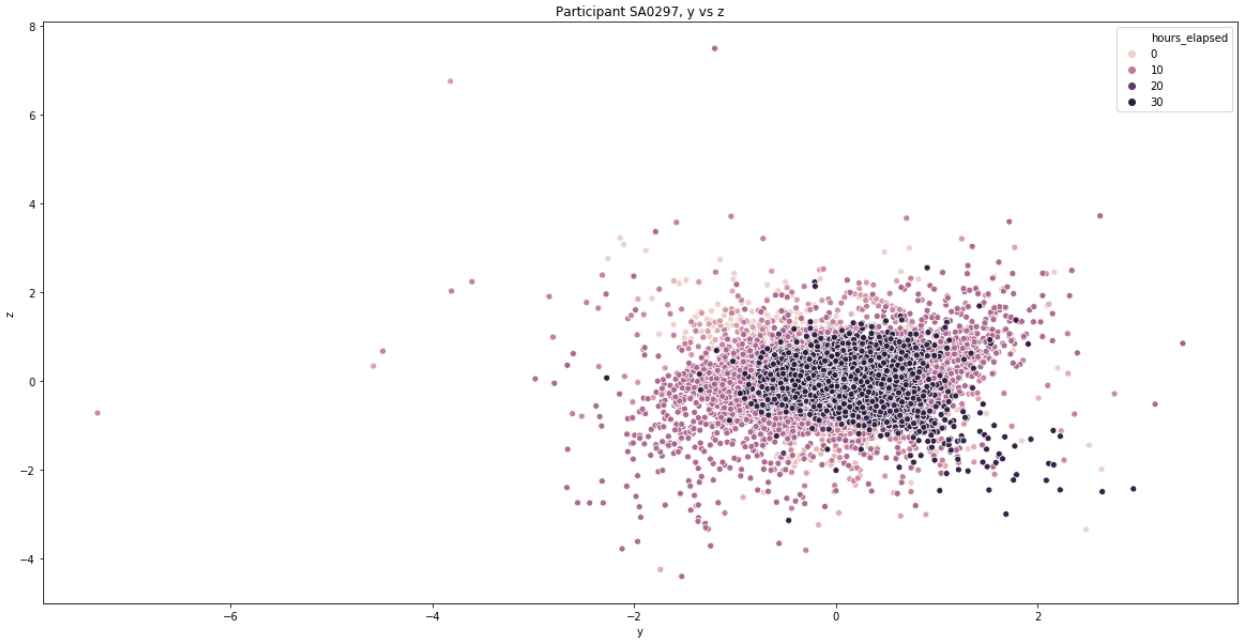
6) Resulting dataframe should have observations with datetime, variance in three axes, TAC reading, and “drunk” label.

Exploratory Data Analysis: Visual Analysis:

1) Visual examination of variance over time clearly shows higher average variance for a drunk participant than a sober one:



2) If the x-axis, which has lower variance than the other axes, is believed to be the vertical axis, then plotting the y- and z-axes against each other should show an “overhead” view of the participant’s movement:



We see the greatest variance for points around 10 hours into the event. This corresponds to midnight and later for this participant, the time when they would be most heavily drinking.

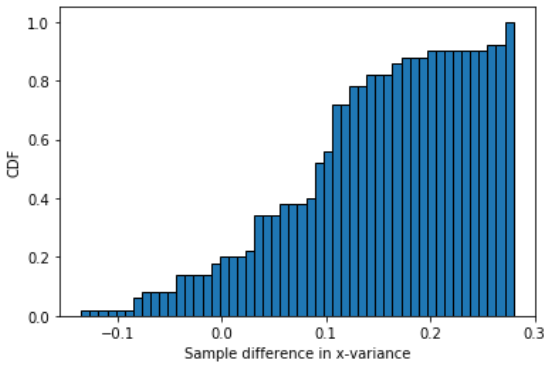
Inferential Statistics:

1) A z-test is used to determine if there is a statistically significant difference between drunk and sober movement. The formula for computing z is as follows:

$$z = \frac{\bar{x}_1 - \bar{x}_2 - \Delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

The standard deviation of both distributions can be computed from the observed data. To get sample means, I randomly sampled with replacement the observed variance with a sample size of 50. Comparing drunk and sober variance in the x-axis for one user, I found a z-score of 6.13, which equates to a two-sided p-value of 5.43e-5. With a significance level of 0.05, this disproves the null hypothesis, that there is no difference

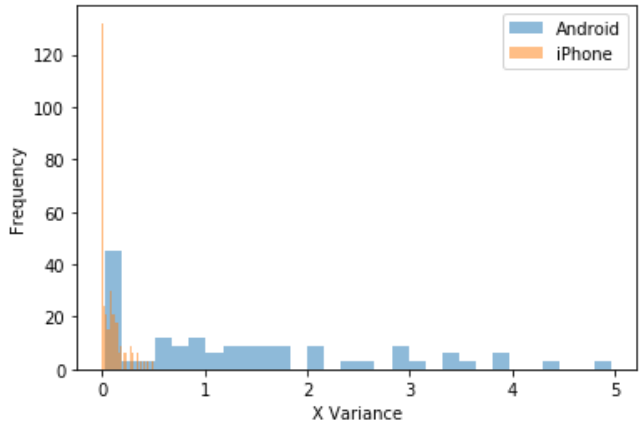
between the groups. The CDF for the sample distribution:



The p-values for the other axes were even smaller: 3e-12 for y-variance, and 7.61e-17 for z-variance.

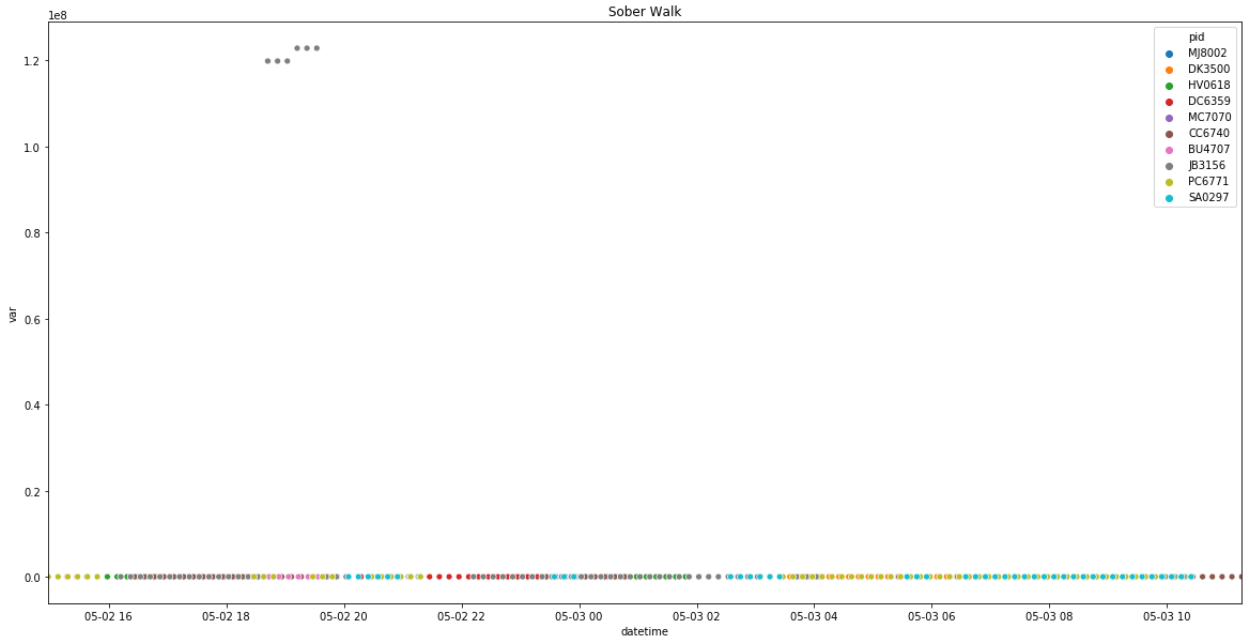
Visual exploratory analysis and inferential statistics indicate that there is a difference in variance of movement while drunk. In other words, there is a relationship between variance and whether a person is drunk. A machine learning model should be able to analyze this relationship and predict a “drunk” label for unseen data.

2) Later, another z-test is used to examine the difference between Android and iPhone users. A quick look at the histograms for both groups shows a clear difference:

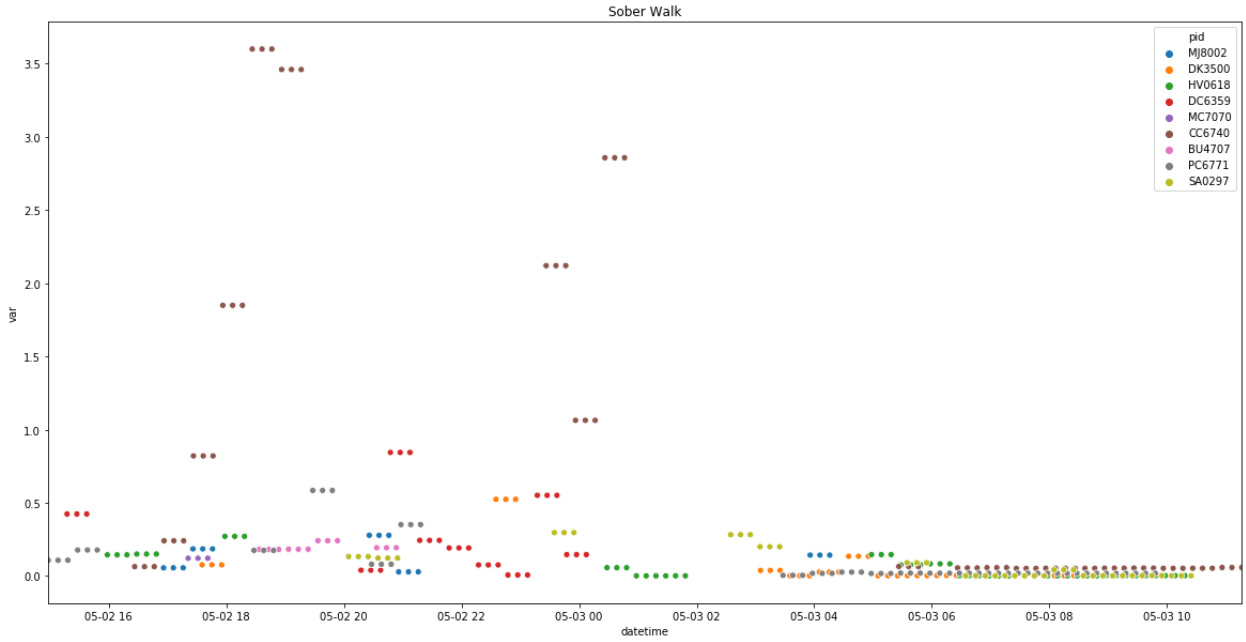


The p-value for the difference in means of x-variance between the groups is found to be 4.9e-10, so we reject the null hypothesis.

We see clear evidence of a difference between Android and iPhone users. This could be potentially explained by a technical fault in how the android application in the original experiment recorded its data. In fact, one Android user in particular is clearly at least an outlier. Look at this following plot of variance over time for sober users:



This one Android user, JB3156, has variance measurements so much higher than all the other participants, the other measurements can't even be shown on the same scale. Qualitatively, we can also observe the impact this one participant has on the dataset. Counterintuitively, with all the users considered, the variance of sober movement is actually significantly higher than for drunk movement; the standard deviations of drunk and sober movement respectively are 1.33 and ~2,000,000. However, when this one participant is excluded, we see more reasonable output:



(Note that even here, the user with the highest variance is the other Android user). The new drunk variance is .78, compared to .25 for sober variance, which corresponds to

our expectations. As a result of this investigation, one model will be trained without this outlier user, another will use phone type as a feature, and a final will also use “crazy_user” as a binary categorical feature.

Machine Learning:

One of the key aspects of building these machine learning models was how to properly split the training and test data. The first twelve models were trained using a naive stratified split - the first 70% of observations were for training data, the remaining 30% for test data - instead of the random sampling used in other machine learning problems. This is done because we are using time series data, meaning all data points for a given feature are correlated due to their relationship to time. If you randomly sample such data, the training data points will give you even more information about test points because of this time correlation, and so the model would have a deceptively high test accuracy but most likely perform poorly on totally new data.

The primary features of all models are the variances in three axes. In addition, some of the models use participant ID as a feature as well. Because Scikit-learn models cannot use strings as features, I instead implemented the pandas get_dummies function, which created a number of categorical label columns for the participant ID’s via One Hot Encoding. The target variable, the “drunk” label, was consistent for all models.

In terms of the algorithms I used, I moved generally from simple to complex. I started with logistic regression, then support vector machines, then random forest, and finally gradient boosting classifier. I also tried to scale the data in two ways: standard scaler, which subtracts the mean and divides by the standard deviation for each point, and minmax scaler, which scales values to between 0 and 1.

Three metrics were primarily used to evaluate models. Training accuracy and test accuracy provide a simple baseline score, and usually test accuracy is lower than training. However, we should note that there is a class imbalance; only about 15% of the data points were labeled drunk. In such cases of class imbalance, accuracy can be a bad metric. For example, if my model predicted sober 100% of the time, it would be 85% accurate here. Consequently, I use F1 score instead as my primary statistic. F1 measures accuracy with precision and recall, which are, respectively, how relevant the results are and how complete the relevant output is.

After 12 models with low scores, I noticed a mistake I had made in splitting the training data. I had naively split the first 70% of the data; instead, I should have split on 70% for

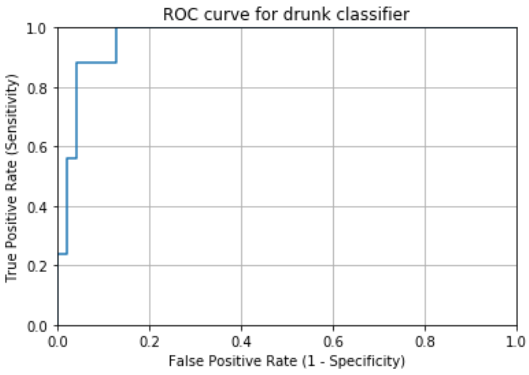
each participant. Splitting the data in this way led to an immediate improvement in both test accuracy and F1 score.

I attempted to use blood alcohol readings as a feature for one set of models. In theory, this should lead to a large increase in test accuracy, as this variable is heavily autocorrelated with the target variable and therefore helps the model "cheat". Unsurprisingly, this increase in accuracy did indeed come, with several models predicting at 100%. Note that these results are actually invalid because a) of the relationship between blood alcohol and the target variable, and b) it's practically meaningless in the real world, since it's relatively easy to find someone's position in space and relatively complex to take consistent blood alcohol readings.

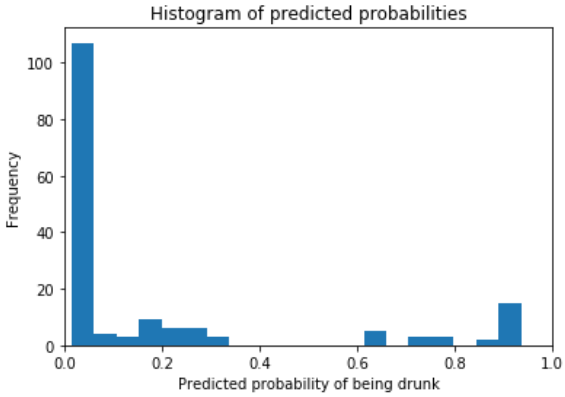
Through 37 models, the best performing was a gradient boosting classifier with participant ID's as a feature with an F1 score of .83 (max of 1). This model was tuned with GridSearchCV and scaled, to little effect. The classification report for this model:

	precision	recall	f1-score	support
0	0.98	0.96	0.97	141
1	0.79	0.88	0.83	25
accuracy			0.95	166
macro avg	0.88	0.92	0.90	166
weighted avg	0.95	0.95	0.95	166

And here we see the ROC curve:

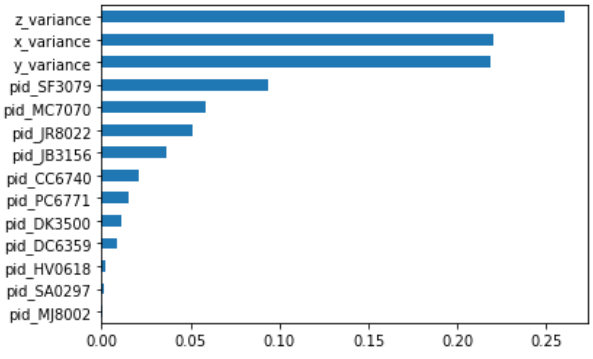


Because of the class imbalance between drunk and sober, I thought it might be good to investigate whether the label threshold should be adjusted. I made a histogram of predicted probabilities:



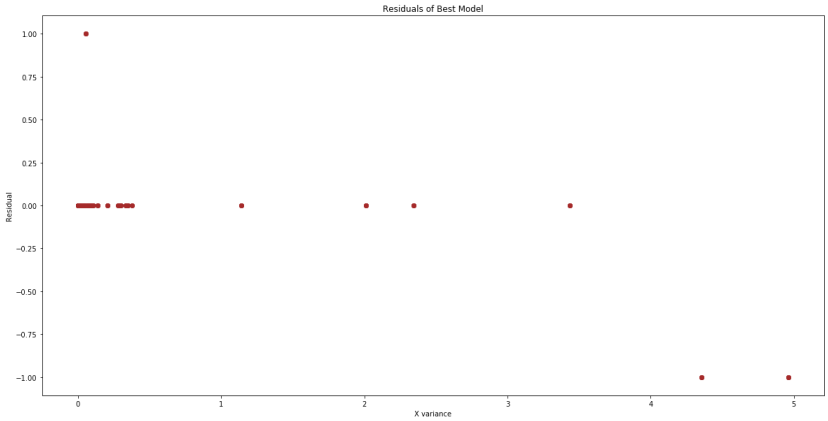
This suggested the threshold might be better at .3 as opposed to the usual .5 value. I extracted the true positive rate, false positive rate, and threshold from the roc_curve function used above. I found that in terms of sensitivity (true positive rate) and specificity (1 - false positive rate), models with a threshold between .3 and .5 are equal, but if the threshold is raised or lowered it becomes worse.

Next, I extracted the feature importances from the best-performing model and graphed them:



As expected, the variances were the most important features, although participant ID also helped the model.

A graph of residuals also indicated a high-performing model:



Three final models were trained. In the first, the “crazy user” JB3156 was excluded from the data. In the second, phone type was encoded via dummy variables. In the third, a label of “crazy_user” was added as a feature. The last two models saw no difference in performance from the previous best performer.

Excluding the outlier user, however, markedly improved the model. It went from a test accuracy of 95% and F1 score of .83 to 98% accuracy and F1 of .94.

model_name	features	training_accuracy	test_accuracy	f1_score	hyperparameters
Gradient Boosting Classifier, new split, dummies, exclude user	variance, pid dummies	1.0000	0.9793	0.9362	
Gradient Boost Classifier, new split, dummies	variance, pid dummies	1.0000	0.9458	0.8302	
Gradient Boosting Classifier, new split, dummies	variance, pid dummies, phone type	1.0000	0.9458	0.8302	
Gradient Boosting Classifier, new split, dummies	variance, pid dummies, phone type, crazy_user	1.0000	0.9458	0.8302	
Gradient Boosting Classifier, new split, Standard Scaler, dummies	variance, pid dummies	0.9674	0.9277	0.7857	
Gradient Boosting Classifier, new split, MinMax Scaler, dummies	variance, pid dummies	0.9674	0.9277	0.7857	
Random Forest Classifier, new split, dummies	variance, pid dummies	1.0000	0.9096	0.7458	
Gradient Boosting Classifier, new split, tuned with GridSearchCV, dummies	variance, pid dummies	0.9810	0.9157	0.7083	{'max_features': None, 'min_samples_leaf': 1,

					'min_samples_split': 8, 'n_estimators': 100, 'warm_start': True}
Logistic Regression, Standard Scaler, new split, dummies	variance, pid dummies	0.7717	0.8916	0.6897	
Random Forest Classifier, new split	variance	1.0000	0.8916	0.6786	
Logistic Regression, new split, dummies	variance, pid dummies	0.7663	0.8675	0.6452	
Logistic Regression, MinMax Scaler, new split, dummies	variance, pid dummies	0.7500	0.7590	0.5000	
Random Forest Classifier with dummies	variance, pid dummies	1.0000	0.7019	0.4286	
Gradient Boost Classifier, new split	variance	1.0000	0.7289	0.3662	
Gradient Boost Classifier with dummies	variance, pid dummies	0.9920	0.7019	0.3333	
Gradient Boost Classifier	variance	0.9920	0.6832	0.3200	
Random Forest Classifier	variance	1.0000	0.5714	0.3030	
Logistic Regression	variance	0.6783	0.6832	0.0000	
Logistic Regression with dummy variables	variance, pid dummies	0.7507	0.6832	0.0000	
SVM classifier	variance	0.6783	0.7019	0.0000	
SVM classifier with dummies	variance, pid dummies	0.6783	0.7019	0.0000	
Logistic Regression with Standard Scaler	variance	0.6702	0.6832	0.0000	
Logistic Regression with Standard Scaler, dummies	variance, pid dummies	0.7507	0.7019	0.0000	
Logistic Regression with MinMax Scaler	variance	0.6622	0.7019	0.0000	
Logistic Regression with MinMax Scaler, dummies	variance, pid dummies	0.7507	0.7019	0.0000	
Logistic Regression, new split	variance	0.6277	0.7952	0.0000	
SVM Classifier, new split	variance	0.6114	0.8494	0.0000	
SVM Classifier, new split, dummies	variance, pid dummies	0.6114	0.8494	0.0000	
Logistic Regression, Standard Scaler, new split	variance	0.6277	0.7590	0.0000	
Logistic Regression, MinMax Scaler, new split	variance	0.6277	0.7771	0.0000	

Conclusion:

We were able to train a gradient boosting classifier that predicts whether someone is drunk based on their movement with an F1 score of .94. Variances in three axes were the most important features, followed by participant ID. Negligible p-values indicated a definitive statistical difference between sober and drunk, and Android / iPhone users. By excluding one droid user who appeared to be registering faulty data, the accuracy of the model was improved even further, although scaling and tuning were less successful.