Keyboard Dynamics

Mohamed Al-Rasbi, Pranav Gummaraj Srinvas, Srinivasu Seeram

Problem Space

Keystroke dynamics is a detailed timing information of a user's typing pattern. When a user types on a keyboard, he follows a unique stroke pattern. The timing data can be collected to classify and identify if the person typing is the original user with a certain confidence interval. Even if the user uses a different type of keyboard, we should be able to identify the pattern, however, the accuracy could decrease. The raw collected data contain key press time, release time and hold time. Various features are extracted from the raw data for classification.

Approach

The first step was collecting data on our own, then extracting features from the collected raw data. Then, we tried different classification algorithms, such as KNN, Logistic Regression and Neural Networks. However, we realized that classification models, in general, would not work on our problem space because we are trying to classify if the current user is the original user, the one we have built our model on. The best way is use anomaly detection methods, which basically will detect outliers. The general idea is to fit the model on one of the training data, then predict from the test data if the user is the original user or not. We tried three different models from scikit-learn: OneClassSVM, LocalOutlierFactor, and EllipticEnvelope. After testing on different parameters, OneClassSVm gave us the best results, which are shown on the results section.

Data

We collected data by ourselves. We wrote a script that uses pynput package in python to collect raw data, which consist of key press time, release time and hold time.

	key	press_time	release_time	hold_time
0	Key.shift	1.556496e+09	1.556496e+09	0.249838
1	A	1.556496e+09	1.556496e+09	0.057448
2	Key.space	1.556496e+09	1.556496e+09	0.079591
3	f	1.556496e+09	1.556496e+09	0.064061
4	r	1.556496e+09	1.556496e+09	0.065949

Then we extracted features from raw data using another script. In general, we are capturing the hold time of a key and the transition time from one key to another. The number of features is 39, however here are some examples:

- ri_left: transition time from left hand to right hand index finger
- rm_ht: hold time of a key pressed by right hand middle finger
- key_sb: transition time from spacebar to any other key

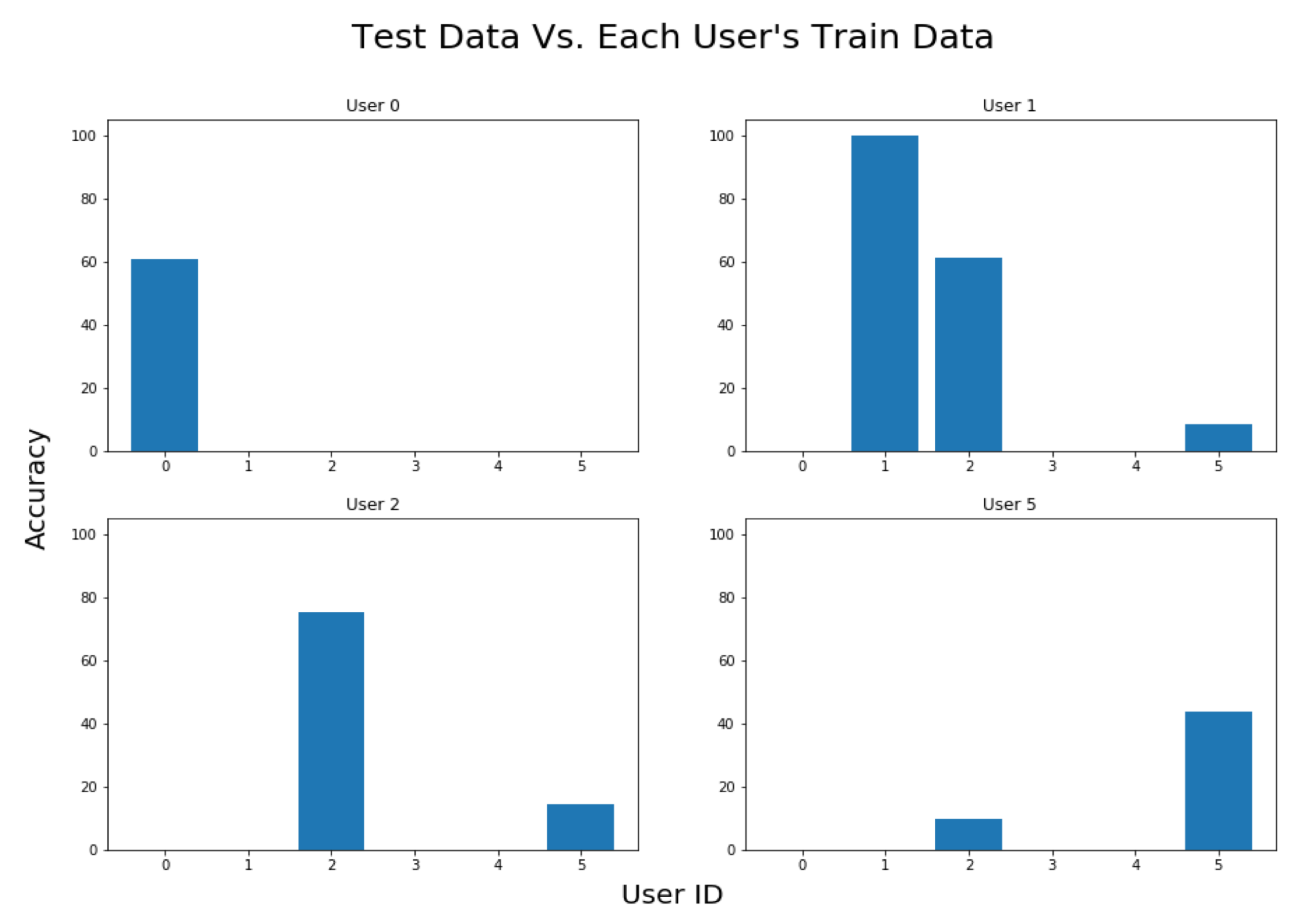
The way we created those features is that we mapped every key to a finger. For example, the key 'a' generally, is pressed by left hand little finger, hence these features are created: ll_ht, ll_same, ll_left, ll_right. We do this for every finger.
Note: Mapping keys to fingers was based on assumption.

	ri_left	ri_right	ri_same	ri_ht	rm_left	rm_right	rm_same	rm_ht	rr_left	rr_
0	0.139271	0.237643	0.164051	0.156092	0.235258	0.186615	0.0	0.152273	0.112993	0.26
1	0.141277	0.225690	0.249318	0.170579	0.098025	0.221116	0.0	0.128798	0.093151	0.23
2	0.299525	0.138293	0.253424	0.262318	0.080132	0.222003	0.0	0.123496	0.180391	0.62
3	0.116764	0.153303	0.253424	0.123494	0.106061	0.190925	0.0	0.130140	0.104458	0.19
4	0.136112	0.168064	0.497378	0.137211	0.180212	0.170476	0.0	0.176967	0.100907	0.07

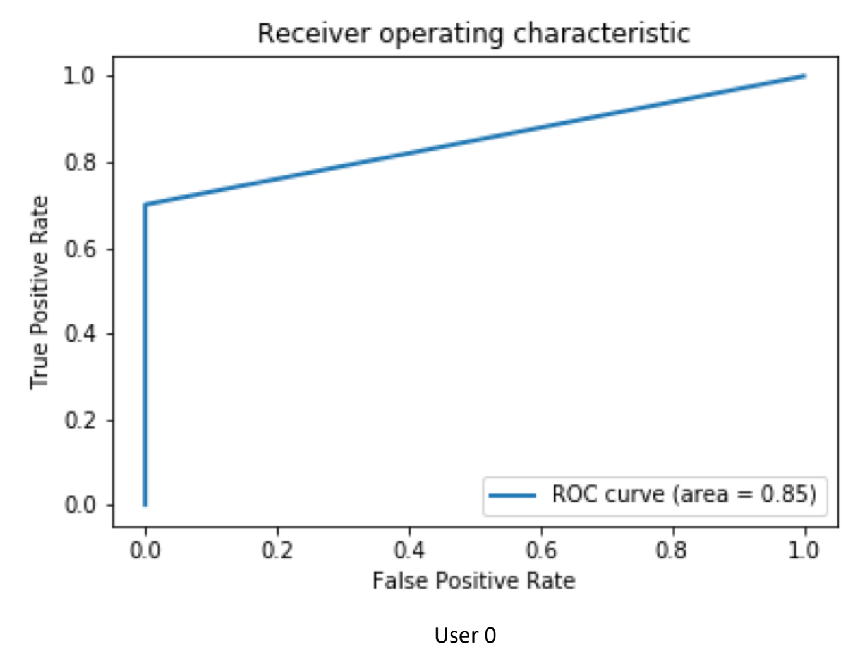
5 rows × 40 columns

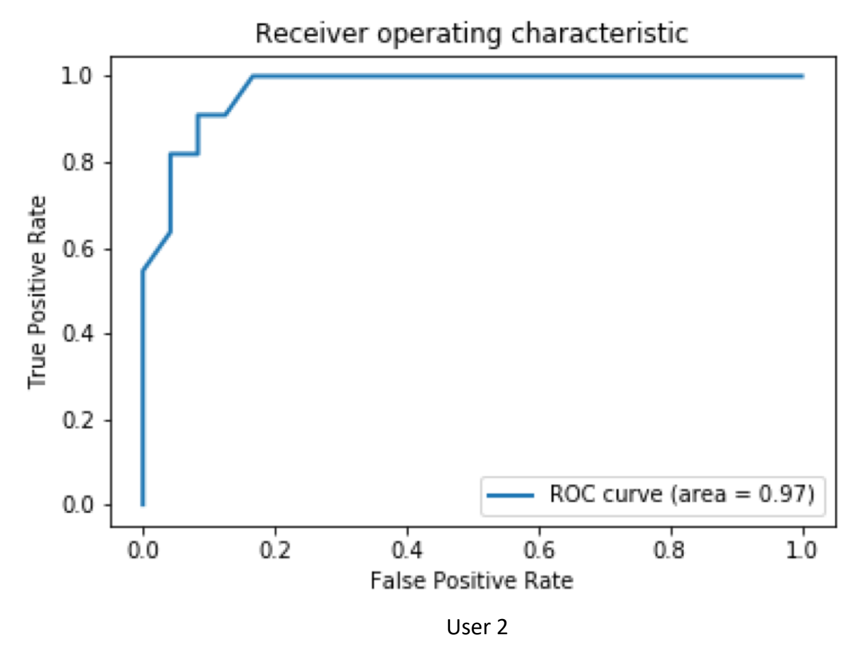
Results

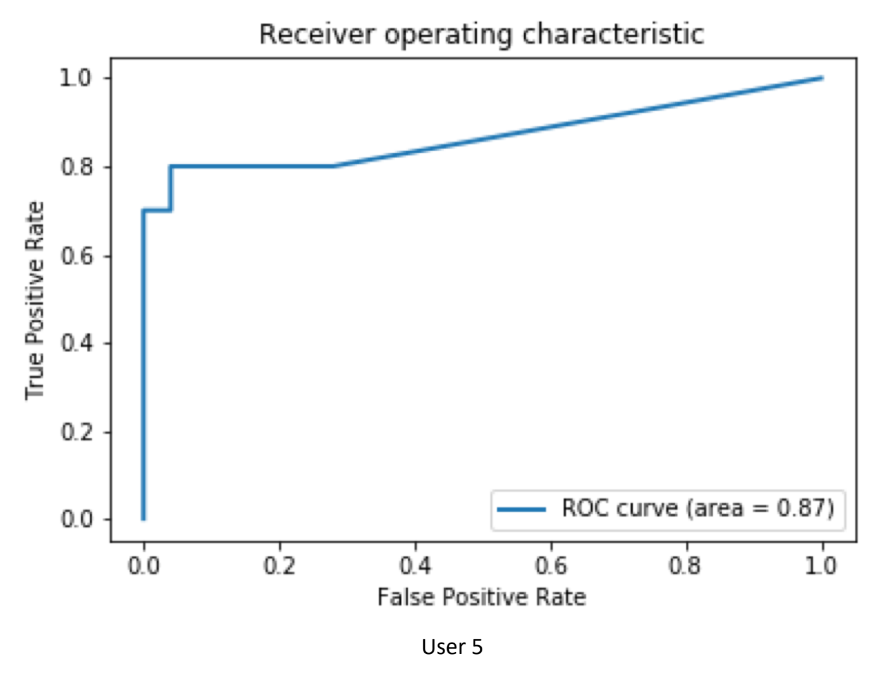
To measure the model's performance, we created the following chart and ROC curves which compares test data of a user with the training data of all the users.



To plot ROC curves, we compared three of the users' hit rate with the false alarm rate. To generate imposter data we used data from other user to test against a particular user.







Discussion

The project showed a promising start using a small set of data and we could better analyse if if we had more data. However, the results were far from perfect. One affecting factor to accuracy is the difficulty in the text that is being typed. Each user's pattern can change depending on the text they are typing; it could contain complicated words that are hard to type. We can investigate further is using touch keyboards instead of normal ones. Also, currently we are not considering the correctness of the data that the user is typing with the one presented to the user. This could be included as one of the features, which could potentially improve our accuracy.

Our future plans for this work include collecting more data, optimize our classifier and explore more anomaly detection models. If we could consider other ways user interacts with the device i.e., mouse pointer, it could extract some useful characteristics to better identify the user correctly.

References

1. K. Killourhy and R. Maxion. Comparing Anomaly-Detection Algorithms for Keystroke Dynamics. In Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on, pages 125–134. IEEE, 2009.
2. Kim, Junhong, Haedong Kim, and Pilsung Kang. "Keystroke dynamics-based user authentication using freely typed text based on user-adaptive feature extraction and novelty detection."
3. Kochegurova, Elena, Elena Luneva, and Ekaterina Gorokhova. "On continuous user authentication via hidden free-text based monitoring." International Conference on Intelligent Information Technologies for Industry. Springer, Cham, 2018.
4. Chatterjee, Kakali. "Continuous User Authentication System: A Risk Analysis Based Approach." Wireless Personal Communications: 1-15.