# Effective Touch Dynamics Features

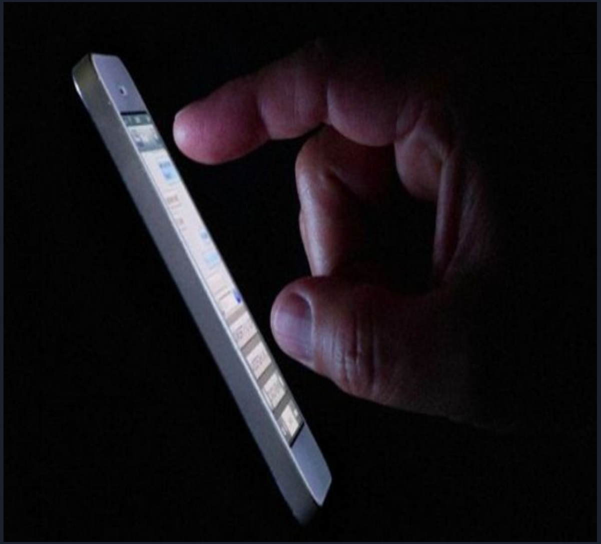
Carlos Leon, Subhrajyoti Pradhan, Karshan Arjun

## Outline

- Introduction to Touch dynamics
- Team Goal
- Team Setup
- Data Acquisition
- Our Research
- Our Results
- Conclusion

## Introduction of Touch Dynamics

- The process of measuring and assessing human touch rhythm on a touch screen device
- A unique signature is created with the devices.
- First works were on the keystrokes from keyboards in the 1980s
- This biometric has useful feature, but has challenging issues such as:
    - *Minimizing computation and communication costs,* so the time complexity of the algorithm must be optimal when gathering data.
    - *Minimizing energy consumption,* on mobile devices the energy from the battery must be used for other resources and gathering data from touch dynamics can lead to high energy consumption
    - *Maximizing Accuracy*, a person's mood, fatigue or distraction can lead to inaccurate scores
    - *Adaptation capability*, this means as the user gets used to the device, the touch pattern might change so there must be adaptation installed in to the touch dynamic system



## Team Goal

- Our goal was to find the most efficient features of touch dynamics
- The reason:
  - Because there was not much study done on finding the most effective features, according to the researchers of  “A survey on touch dynamics authentication in mobile devices”
  - The need for this should improve the study of this biometric
  - As the study improves we should see more mobile devices integrate the features for more optimal security for mobile devices.
- If our goal was reached, we felt that we could have helped out in the challenges that were currently faced in this biometrics.
- We assume this would lead to a lesser time to get highly accurate results, which should take up less energy consumed. We also assume this would lead to easier methods of adapting the biometric to the user habits.

## Data Acquisition

- We used  datasets containing touch dynamic features, which was provided to us by Dr. Tempestt.
- Each subject was instructed to type '.tie5Roanl' as their keystroke/touch information was collected.
- The measured features (attributes) are Hold (H), Up-Down (UD), Down-Down (DD), Pressure (P), Finger-Area (A), Averages of Hold (AH), Pressure (AP) and Finger Area (AFA).
- There are 71 features because each feature has a set of feature elements corresponding to the typed characters.

## Team Setup

We put up various feature selection algorithms which we employed in our final processes of validation. The algorithms involved were -

- Chi square -measures the frequency of expected and observed data, was used to filter the amount of features of touch dynamics.
- Information gain - using entropy in this algorithm we were able to also filter out the best features which had a high information gain.
- Recursive - select features by recursively considering smaller and smaller sets of features. The estimator was defined as a support vector machine Then, the least important features were removed from current set of features.That procedure was recursively repeated on the removed set until the desired number of features to select is eventually reached.
- Tree Selection - It employs the process of  tree selection to determine feature importance. It then uses the average of the importance to remove unneeded features
- Variance Threshold -  remove all features whose variance doesn't meet the declared threshold.

## Our Research - Initial Approach

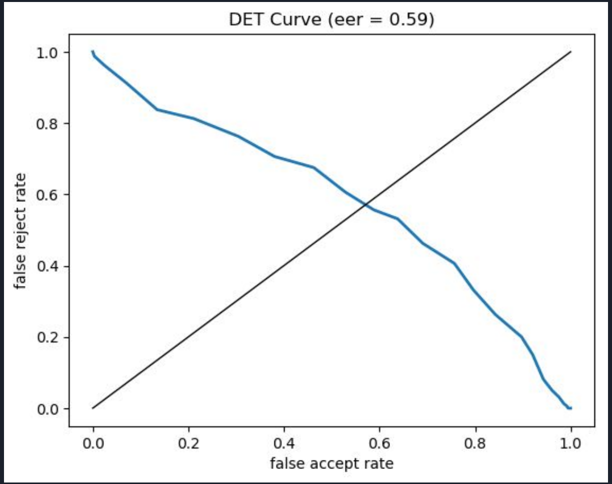
- The script breaks up the dataset using 5 fold cross validation.
- The chunk meant to be used for testing was left alone, and the rest had outliers removed.
- Outlier detection was done using isolation forest in combination with leave-one-out-validation. Thus every row was tested as being an outlier against every other row. In practice we found that only a small percentage of rows were dropped; about 5%.
- The remaining training rows post outlier detection were used for the KNN training. We chose k=31 since that yielded possible results during debugging with a partial database, but later tested other values for k.
- The main script was adapted into another script called a validation script.
- The purpose of this was to test our method so far on another database - the second database. This would serve to confirm or reject out method.
- The script initially used all the same code with only minor changes to account for different location of the dataset and type of file to read.

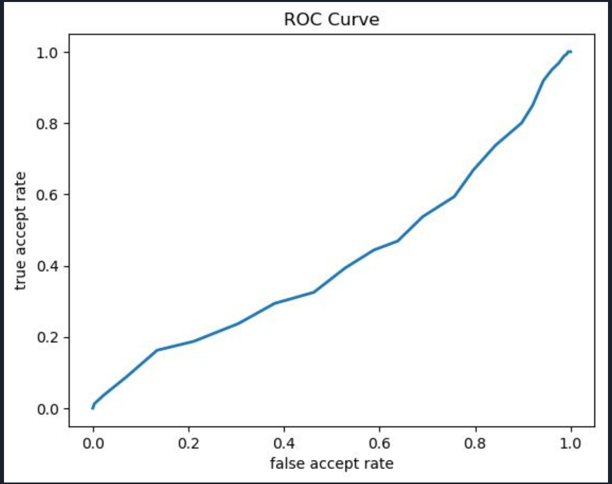
## Our Research - Observations during testing

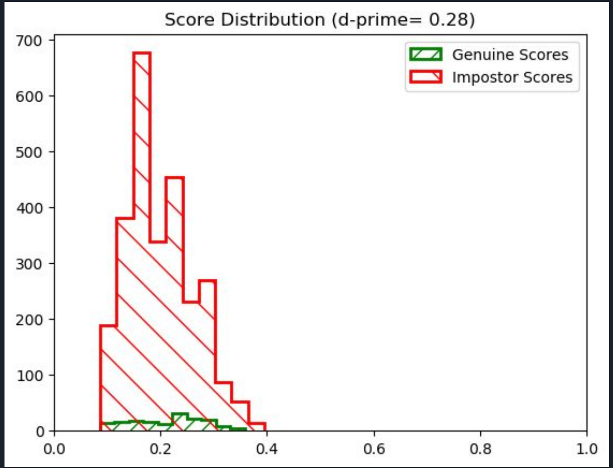
- The method used for the first dataset did not work when applied to the second dataset. It rejected our method. In order to make it work, we started analyzing it.
- First we noticed that there were simply too much data, for the first dataset. Out script never finished even when given several hours.
- To remedy this we switched to taking a random sample of 2,000 rows.
- Secondly, we noticed the data had text data and our script was meant to work with numerical. To remedy this we applied a label encoder.
- Thirdly, we noticed that the recursive feature elimination took too long.
- To remedy this we changed the SVM estimator for a Logistic Estimator that would perform much faster.
- An SGD Estimator was used. It worked considerably faster and did converge.
- The feature selection algorithms did not agree on a set of features to keep after the intersection of them were taken. We could not solve this.
- For the machine learning algorithm to work more than two or more feature columns were needed, but we saw that after 5 attempts to get like features at best 1 feature was chosen by the 5 algorithms.

# Our Results

- We used ~350 rows of the first dataset when building our script. When the script was done we observed that with k for KNN being 31 for 12 runs we averaged 65.7% accuracy. The number of features chosen were between 3 - 6, collectively they were Hold i, Hold Shift, Hold Caps, Hold a, Hold n, Hold l.
- On testing our code on the entire dataset, we got strangely low accuracy for these results,for integrity they were 5.6% to 7%. With the 7% being reached with k=75 and the lowest coming from k=81.







## Conclusion

- Our approach to feature selection yielded fairly high results when used on small samples.
- Even though our results were, far from ideal when it came to testing on the entire dataset, we determined the major problem existed in the speed of performing all the necessary calculation.
- Our approach of using k Nearest Neighbor, seemed to be the issue, as our k value was not ideal for our dataset. We were unable to determine a good k value as the runtime of our entire code exceeded thirty minutes, and hence multiple tests on sample datasets were not feasible.
- We determined some sort of tree approach like the random forest would have been better for future works.
- This would ensure us to achieve our goal of achieving a system, which would seamlessly implement the touch gesture as a feasible mode of biometric authentication.

## Sources

- Niu, Yuan, and Hao Chen. "Gesture authentication with touch input for mobile devices." *International Conference on Security and Privacy in Mobile Information and Communication Systems.* Springer, Berlin, Heidelberg, 2011.
- Teh, Pin Shen, et al. "A survey on touch dynamics authentication in mobile devices." *Computers & Security* 59 (2016): 210-235.

- Jain, Anil K., Arun A. Ross, and Karthik Nandakumar. *Introduction to biometrics.* Springer Science & Business Media, 2011.

# Q&A