В наш час з високими темпами розвинення технологій потрібні відповідні способи забезпечити безпеку користувача від втрати данних, або видавання користувача за когось іншого. Будь який статичний пароль може бути зламаний в майбутньому, або навіть зараз. Саме тому потрібно використовувати більш надійні способи ідентифікації, а точніше біометричні: відбиток пальця, силует обличчя, тощо.

Біометрія натискання клавіш (Keystroke authentication), спосіб визначити чи є користувач тим кого ми очікуємо побачити. Вхідними даними в даному випадку є тимчасові інтервали між натисканнями клавіш і час утримання клавіші, утримуючи. У разі набору тексту людиною так само грає роль використання лівого / правого shift або caps lock.

Для того що б навчитися визначати чи є людина тим за кого ми його приймаємо потрібно виконати наступні пункти:

• Зібрати дані про користувача, визначити час середні натискання між усіма клавішами.

• Знайти моменти властиві лише цьому користувачеві.

Далі коли ми зібрали достатньо даних і згрупували в безліч точок в просторі, виділили в область, за такими вхідними даними можемо визначити чи лежить наша точка в області. Для більш точного результату використовуються довірчі інтервали і середньоквадратичне відхилення. На виході отримуємо ступінь впевненості що точка належить до області, ймовірність того що користувач той за кого себе видає.

Звіт про виконану роботу

Робота складалася з двох частин. Перша підготувати данні. Одні данні нам вже дали готовими, другі потрібно було зібрати самостійно. Пароль на базі якого ми мали будувати данні: **.tie5Roanl** У папці **console\_keylogger > data** знаходяться данні, які нам надіслали. У папці self\_data знаходяться три датасети, які я підготував сам за допомогою файлу **console\_fixed\_pas.py** . Функція press фіксує коли клавіша була натиснута, функція release фіксує коли клавіша була відпущена. Далі ми в циклі проганяємо це 15 разів, для того щоб зробити 3 датасети з вводом пароля. Далі у файлі **stat\_analyzer.py** ми рахуємо деякі вибіркові статистики (Дисперсію, середнє арифметичне, стандартне відхилення, тощо) та перевіряемо наші данні на коректність. І в останньому файлі в **classifier.py**: функції uni\_parse(), bio\_parse(), trio\_parse() готують наші данні. FAR, FRR рахують кількість помилок False-Positive та False-Nagative. arr\_pred i arr\_val – значення, яке спрогнозував алгоритм та значення яке є насправді. У нас в програмі будемо вважати, що якщо значення 0, то користувач вводив пароль сам, а якщо 1, то його намагаються взломати. 30% своїх даних відберемо під тестову вибірку. apply\_tree(), apply\_knn(), apply\_MLP() – алгоритми машинного навчання, за якими навчається наша программа.

В папці **Original** знаходиться вже Android проэкт. Якщо запустити через Android Studio, то можна побачити поле вводу, клавіатуру та кнопку, які в перспективі будуть використовувати класифікатор, та зможуть відповідати на питання, який саме користувач вводить пароль.