Steps to run the Project:

1). Run project on PyCharm (Recommended) - Version: [Python 3.7]
also, Anaconda Navigator (JupyterLab) – Version: 1.9.7

2). Install and import the libraries if not available

- Import Pandas
- Import Numpy
- Import Math
- From sklearn.metrics Import roc_curve
- From scipy.spatial.distance Import cityblock, euclidean
- From array Import array
- Import OS
- Import Random

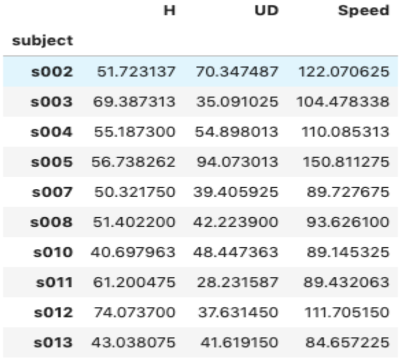
3). In PyCharm: list of file names to run

- DataInsights.py
- Keystroke detector.py
  - ➔ HybridManhattan.py
  - ➔ Ecludian.py
- UserCategorization.py: Categorization of users on the basis of sessions of each user.
- All_UserCategorization.py: Categorization of users on the basis of FAR, FRR and TPR.
- Genetic_keystroke.py
- Multi_Genetic_Keystroke.py
- DSL-StrongPasswordData.csv

4). In Anaconda: Same file names with “. ipynb” extension.

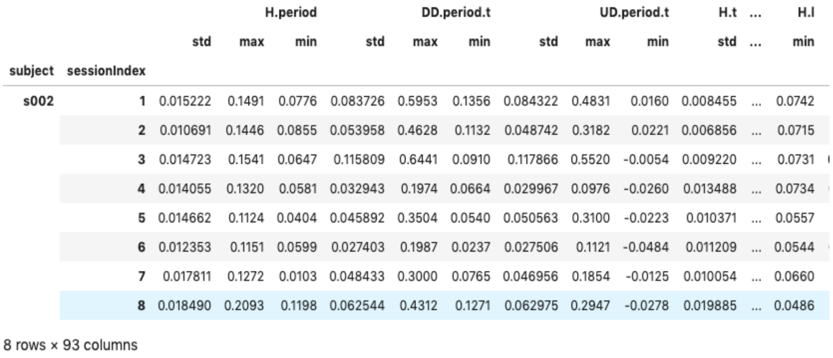
## **<u>Output Explanation:</u>**

### 1). Data Insights



| | H | UD | Speed |
|---|---|---|---|
| subject | | | |
| s002 | 51.723137 | 70.347487 | 122.070625 |
| s003 | 69.387313 | 35.091025 | 104.478338 |
| s004 | 55.187300 | 54.898013 | 110.085313 |
| s005 | 56.738262 | 94.073013 | 150.811275 |
| s007 | 50.321750 | 39.405925 | 89.727675 |
| s008 | 51.402200 | 42.223900 | 93.626100 |
| s010 | 40.697963 | 48.447363 | 89.145325 |
| s011 | 61.200475 | 28.231587 | 89.432063 |
| s012 | 74.073700 | 37.631450 | 111.705150 |
| s013 | 43.038075 | 41.619150 | 84.657225 |

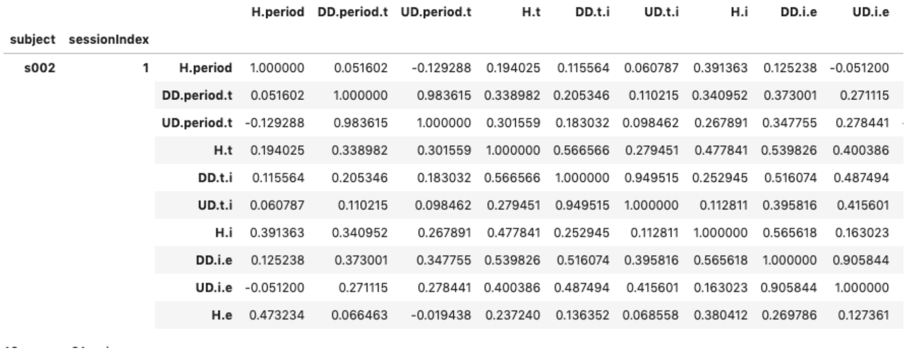
Figure 1: Speed of user on basis of H + UD key values



| | | H.period | | | DD.period.t | | | UD.period.t | | | H.t | ... | H.l |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | std | max | min | std | max | min | std | max | min | std | ... | min |
| subject | sessionIndex | | | | | | | | | | | | |
| s002 | 1 | 0.015222 | 0.1491 | 0.0776 | 0.083726 | 0.5953 | 0.1356 | 0.084322 | 0.4831 | 0.0160 | 0.008455 | ... | 0.0742 |
| | 2 | 0.010691 | 0.1446 | 0.0855 | 0.053958 | 0.4628 | 0.1132 | 0.048742 | 0.3182 | 0.0221 | 0.006856 | ... | 0.0715 |
| | 3 | 0.014723 | 0.1541 | 0.0647 | 0.115809 | 0.6441 | 0.0910 | 0.117866 | 0.5520 | -0.0054 | 0.009220 | ... | 0.0731 |
| | 4 | 0.014055 | 0.1320 | 0.0581 | 0.032943 | 0.1974 | 0.0664 | 0.029967 | 0.0976 | -0.0260 | 0.013488 | ... | 0.0734 |
| | 5 | 0.014662 | 0.1124 | 0.0404 | 0.045892 | 0.3504 | 0.0540 | 0.050563 | 0.3100 | -0.0223 | 0.010371 | ... | 0.0557 |
| | 6 | 0.012353 | 0.1151 | 0.0599 | 0.027403 | 0.1987 | 0.0237 | 0.027506 | 0.1121 | -0.0484 | 0.011209 | ... | 0.0544 |
| | 7 | 0.017811 | 0.1272 | 0.0103 | 0.048433 | 0.3000 | 0.0765 | 0.046956 | 0.1854 | -0.0125 | 0.010054 | ... | 0.0660 |
| | 8 | 0.018490 | 0.2093 | 0.1198 | 0.062544 | 0.4312 | 0.1271 | 0.062975 | 0.2947 | -0.0278 | 0.019885 | ... | 0.0486 |

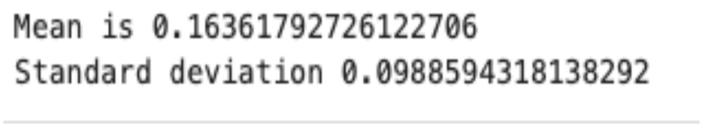
8 rows × 93 columns

Figure 2: Max, Min, Std deviation of each key



| | | | H.period | DD.period.t | UD.period.t | H.t | DD.t.i | UD.t.i | H.i | DD.i.e | UD.i.e |
|---|---|---|---|---|---|---|---|---|---|---|---|
| subject | sessionIndex | | | | | | | | | | |
| s002 | 1 | H.period | 1.000000 | 0.051602 | -0.129288 | 0.194025 | 0.115564 | 0.060787 | 0.391363 | 0.125238 | -0.051200 |
| | | DD.period.t | 0.051602 | 1.000000 | 0.983615 | 0.338982 | 0.205346 | 0.110215 | 0.340952 | 0.373001 | 0.271115 |
| | | UD.period.t | -0.129288 | 0.983615 | 1.000000 | 0.301559 | 0.183032 | 0.098462 | 0.267891 | 0.347755 | 0.278441 |
| | | H.t | 0.194025 | 0.338982 | 0.301559 | 1.000000 | 0.566566 | 0.279451 | 0.477841 | 0.539826 | 0.400386 |
| | | DD.t.i | 0.115564 | 0.205346 | 0.183032 | 0.566566 | 1.000000 | 0.949515 | 0.252945 | 0.516074 | 0.487494 |
| | | UD.t.i | 0.060787 | 0.110215 | 0.098462 | 0.279451 | 0.949515 | 1.000000 | 0.112811 | 0.395816 | 0.415601 |
| | | H.i | 0.391363 | 0.340952 | 0.267891 | 0.477841 | 0.252945 | 0.112811 | 1.000000 | 0.565618 | 0.163023 |
| | | DD.i.e | 0.125238 | 0.373001 | 0.347755 | 0.539826 | 0.516074 | 0.395816 | 0.565618 | 1.000000 | 0.905844 |
| | | UD.i.e | -0.051200 | 0.271115 | 0.278441 | 0.400386 | 0.487494 | 0.415601 | 0.163023 | 0.905844 | 1.000000 |
| | | H.e | 0.473234 | 0.066463 | -0.019438 | 0.237240 | 0.136352 | 0.068558 | 0.380412 | 0.269786 | 0.127361 |

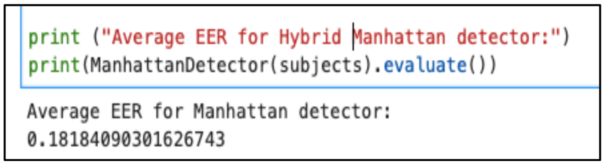
Figure 3: Correlation among the columns in the dataset



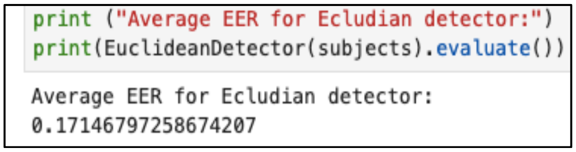
```
Mean is 0.16361792726122706
Standard deviation 0.0988594318138292
```

Figure 4: Overall mean and std deviation

## 2). Keystroke Detector Analysis



```
print ("Average EER for Hybrid Manhattan detector:")
print(ManhattanDetector(subjects).evaluate())

Average EER for Manhattan detector:
0.18184090301626743
```

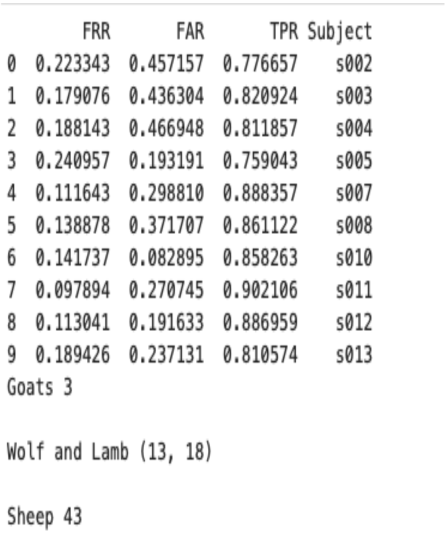


```
print ("Average EER for Ecludian detector:")
print(EuclideanDetector(subjects).evaluate())

Average EER for Ecludian detector:
0.17146797258674207
```

Figure 5: Hybrid Manhattan Equal Error rate.

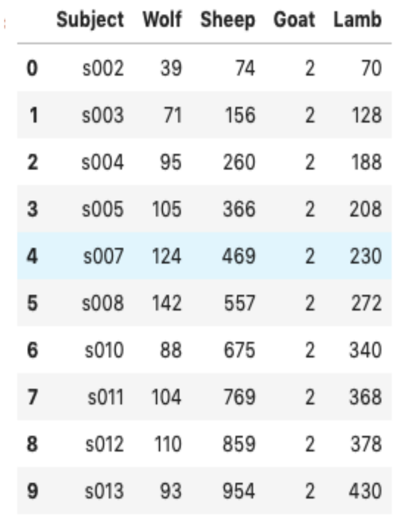
Figure 6: Euclidean Equal Error rate

## 3). User Categorization



```
        FRR       FAR       TPR Subject
0  0.223343  0.457157  0.776657    s002
1  0.179076  0.436304  0.820924    s003
2  0.188143  0.466948  0.811857    s004
3  0.240957  0.193191  0.759043    s005
4  0.111643  0.298810  0.888357    s007
5  0.138878  0.371707  0.861122    s008
6  0.141737  0.082895  0.858263    s010
7  0.097894  0.270745  0.902106    s011
8  0.113041  0.191633  0.886959    s012
9  0.189426  0.237131  0.810574    s013
Goats 3

Wolf and Lamb (13, 18)

Sheep 43
```

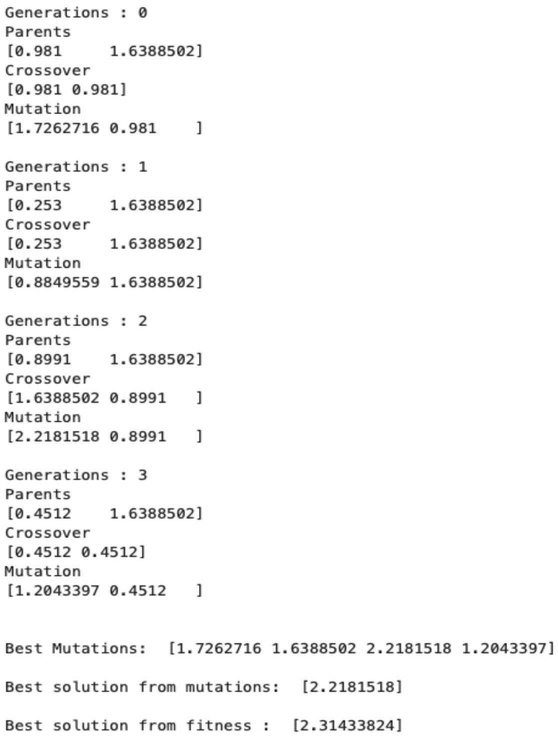
Figure 7: User Category Detection based on overall FAR, FRR and TPR values.



| | Subject | Wolf | Sheep | Goat | Lamb |
|---|---|---|---|---|---|
| **0** | s002 | 39 | 74 | 2 | 70 |
| **1** | s003 | 71 | 156 | 2 | 128 |
| **2** | s004 | 95 | 260 | 2 | 188 |
| **3** | s005 | 105 | 366 | 2 | 208 |
| **4** | s007 | 124 | 469 | 2 | 230 |
| **5** | s008 | 142 | 557 | 2 | 272 |
| **6** | s010 | 88 | 675 | 2 | 340 |
| **7** | s011 | 104 | 769 | 2 | 368 |
| **8** | s012 | 110 | 859 | 2 | 378 |
| **9** | s013 | 93 | 954 | 2 | 430 |

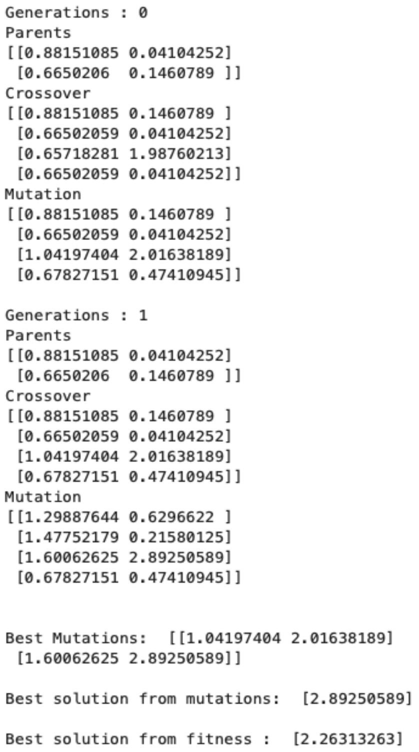
Figure 8: User Category Detection based on each session of a user FAR, FRR and TPR values.

## 4). Biometric Adversary Framework



```
Generations : 0
Parents
[0.981     1.6388502]
Crossover
[0.981 0.981]
Mutation
[1.7262716 0.981    ]

Generations : 1
Parents
[0.253     1.6388502]
Crossover
[0.253     1.6388502]
Mutation
[0.8849559 1.6388502]

Generations : 2
Parents
[0.8991    1.6388502]
Crossover
[1.6388502 0.8991   ]
Mutation
[2.2181518 0.8991   ]

Generations : 3
Parents
[0.4512    1.6388502]
Crossover
[0.4512 0.4512]
Mutation
[1.2043397 0.4512   ]


Best Mutations:  [1.7262716 1.6388502 2.2181518 1.2043397]

Best solution from mutations:  [2.2181518]

Best solution from fitness :  [2.31433824]
```

Figure 9: Genetic algorithm based on single chromosome length.



```
Generations : 0
Parents
[[0.88151085 0.04104252]
 [0.6650206  0.1460789 ]]
Crossover
[[0.88151085 0.1460789 ]
 [0.66502059 0.04104252]
 [0.65718281 1.98760213]
 [0.66502059 0.04104252]]
Mutation
[[0.88151085 0.1460789 ]
 [0.66502059 0.04104252]
 [1.04197404 2.01638189]
 [0.67827151 0.47410945]]

Generations : 1
Parents
[[0.88151085 0.04104252]
 [0.6650206  0.1460789 ]]
Crossover
[[0.88151085 0.1460789 ]
 [0.66502059 0.04104252]
 [1.04197404 2.01638189]
 [0.67827151 0.47410945]]
Mutation
[[1.29887644 0.6296622 ]
 [1.47752179 0.21580125]
 [1.60062625 2.89250589]
 [0.67827151 0.47410945]]


Best Mutations:  [[1.04197404 2.01638189]
 [1.60062625 2.89250589]]

Best solution from mutations:  [2.89250589]

Best solution from fitness :  [2.26313263]
```

Figure 10: Genetic algorithm based on multi chromosome length.