User Authentication through Keystroke Dynamics

Report

CS725 Project

Indian Institute of Technology, Bombay
Department of Computer Science and Engineering

Devashish Singh (163059001)
Prateek Patidar (163059006)
Shubham Singh (163059008)
Hareesh Kumar (16305R013)

1 Project Description

Keystroke dynamics—the analysis of typing rhythms to discriminate among users—has been proposed for detecting impostors (i.e., both insiders and external attackers). Since many anomaly-detection algorithms have been proposed for this task, it is natural to ask which are the top performers (e.g., to identify promising research directions). Unfortunately, we cannot conduct a sound comparison of detectors using the results in the literature because evaluation conditions are inconsistent across studies.

2 Dataset Collection

As the subject types the password, it is checked for correctness. If the subject makes a typographical error, the application prompts the subject to retype the password. Whenever the subject presses or releases a key, the event (i.e., keydown or keyup), along the key involved, and a timestamp for the moment at which the keystroke event occurred. An external reference clock was used to generate highly accurate timestamps. The reference clock was demonstrated to be accurate to within 200 microseconds (by using a function generator to simulate key presses at fixed intervals). 51 subjects (typists) typed the same password, and each subject typed the password 400 times over 8 sessions (50 repetitions per session). They waited at least one day between sessions, to capture some of the day-to-day variation of each subject's typing. The password (.tie5Roanl) was chosen to be representative of a strong 10-character password.

3 Dataset Information

The data are arranged as a table with 34 columns. Each row of data corresponds to the timing information for a single repetition of the password by a single subject. The first column, subject, is a unique identifier for each subject (e.g., s002 or s057). The second column, sessionIndex, is the session in which the password was typed (ranging from 1 to 8). The third column, rep, is the repetition of the password within the session (ranging from 1 to 50). The remaining 31 columns present the timing information for the password. The name of the column encodes the type of timing information. Column names of the form H.key designate a hold time for the named key (i.e., the time from when key was pressed to when it was released). Column names of the form DD.key1.key2 designate a keydown-keydown time for the named digraph (i.e., the time from when key1 was pressed to when key2 was pressed). Column names of the form UD.key1.key2 designate a keyup-keydown time for the named digraph (i.e., the time from when key1 was released to when key2 was pressed). Note that UD times can be negative, and that H times and UD times add up to DD times.

4 Papers

- Comparing Anomaly Detectors for Keystroke Dynamics[1]
- ROCR: visualizing classifier performance in R [2]

5 Approach

- Feature engineering on the dataset i.e. added square and square root of features. We then used some classification methods available in scikit-learn library on the data like Logistic Regression, Support Vector Machines, Random Forests, K NN Classification,MLP and Gaussian Naive Bayes.
- Learnt why few algorithms performs better compared to others, and then tuned the data set with various constraints to get better accuracy.
- Trigraph was done on the given dataset, the given dataset had only dipgrah dataset, however there was no significatnt change in the accuracy of the model
- Gaussian Mixture Model [3] : In keystroke dynamics, false acceptance usually stems from the similarity in user's rhythm during typing the most common digraphs. In Gaussian-based identification, this statement corresponds to having similar mean and variance values.GMM is capable

of overcoming this issue by incrementing the number of components, if enough distinction is not provided.

6 Results

Results without Feature Engineering:
Logistic Regression : 0.702777777778
Support Vector Machines : 0.0970588235294
Random Forests : 0.873366013072
K NN Classification : 0.157679738562
Gaussian Naive Bayes : 0.57091503268

Results after Feature Engineering:
Logistic Regression : 0.776307189542
Support Vector Machines : 0.0197712418301
Random Forests : 0.854738562092
K NN Classification : 0.15637254902
Gaussian Naive Bayes : 0.37385620915

Results after Fine Tuning:

(xx/xx) means Training Data /Test Data

Logistic Regression : 0.686111111111
Logistic Regression + L1: 0.760784313725
Logistic Regression + l2 : 0.686111111111
.7/.3

Logistic Regression : 0.703676470588
Logistic Regression + L1: 0.759068627451
Logistic Regression + l2 : 0.70367647058
.8/.2

Logistic Regression : 0.722549019608
Logistic Regression + L1: 0.76862745098
Logistic Regression + l2 : 0.722549019608
.9/.1

Support Vector Machines : 0.752941176471
Support Vector Machines : 0.354901960784

.9/.1

Support Vector Machines : 0.689869281046
Support Vector Machines : 0.258660130719

.7/.3

Support Vector Machines : 0.716911764706
Support Vector Machines : 0.298284313725

.8/.2

Random Forests 100-gini: 0.875735294118
Random Forests 100-entropy: 0.871323529412
Random Forests 300-gini: 0.882843137255
Random Forests 51-gini: 0.872058823529
Random Forests 10-gini: 0.79387254902

.7/.3

Random Forests 100-gini: 0.899019607843
Random Forests 100-entropy: 0.895588235294
Random Forests 300-gini: 0.894607843137
Random Forests 51-gini: 0.886274509804
Random Forests 10-gini: 0.818137254902
Random Forests 1000-gini: 0.902450980392
Random Forests 3-entropy: 0.65637254902

.9/.1

K NN Classification : 0.170098039216
K NN Classification : 0.137745098039
K NN Classification : 0.0632352941176
K NN Classification : 0.0446078431373
Gaussian Naive Bayes : 0.405392156863
MLP (1200/50) : 0.0196078431373

The best achieved accuracy is about 0.89 by using Random Forest.

References

[1] Kevin S. Killourhy and Roy A. Maxion. Comparing anomaly detectors for keystroke dynamics. In *Proceedings of the 39th Annual International Conference on Dependable Systems and Networks*, DSN-2009, pages 125–134, New York, NY, USA, 2009. IEEE.

[2] Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. Rocr: visualizing classifier performance in r. *Bioinformatics*, 21(20):3940, 2005.

[3] Shambhu Upadhyaya Hayreddin Çeker. Enhanced recognition of keystroke dynamics using gaussian mixture models. *Military Communications Conference, MILCOM 2015 IEEE*, 2015.

Keystroke recording - https://github.com/goncalopp/keystroke$_d$*ynamics*

Viz

April 29, 2017

1 Analysis on the keystroke-timing dataset

Click here for Dataset

1.1 Information about the data

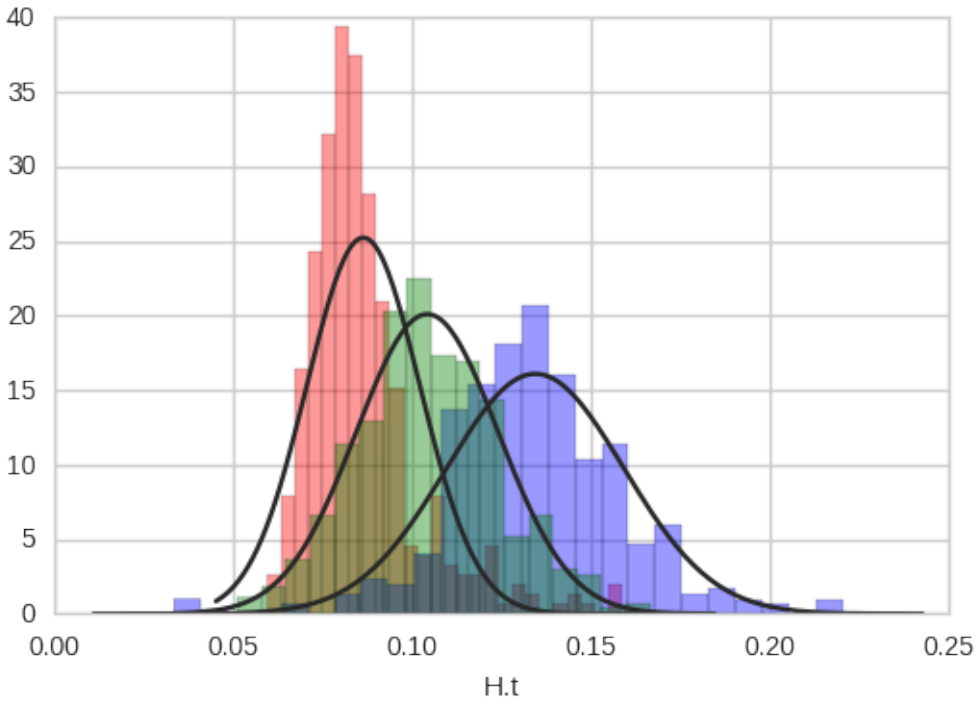
The data are arranged as a table with 34 columns. Each row of data corresponds to the timing information for a single repetition of the password by a single subject. The first column, subject, is a unique identifier for each subject (e.g., s002 or s057). Even though the data set contains 51 subjects, the identifiers do not range from s001 to s051; subjects have been assigned unique IDs across a range of keystroke experiments, and not every subject participated in every experiment. For instance, Subject 1 did not perform the password typing task and so s001 does not appear in the data set. The second column, sessionIndex, is the session in which the password was typed (ranging from 1 to 8). The third column, rep, is the repetition of the password within the session (ranging from 1 to 50).

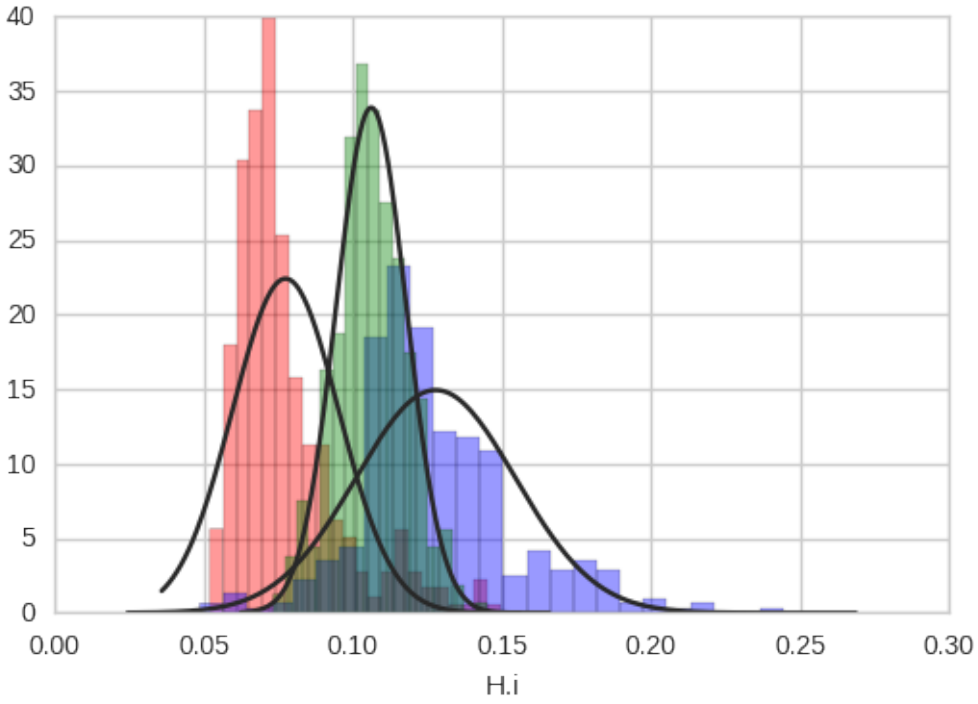
The remaining 31 columns present the timing information for the password. The name of the column encodes the type of timing information. Column names of the form H.key designate a hold time for the named key (i.e., the time from when key was pressed to when it was released). Column names of the form DD.key1.key2 designate a keydown-keydown time for the named digraph (i.e., the time from when key1 was pressed to when key2 was pressed). Column names of the form UD.key1.key2 designate a keyup-keydown time for the named digraph (i.e., the time from when key1 was released to when key2 was pressed). Note that UD times can be negative, and that H times and UD times add up to DD times.

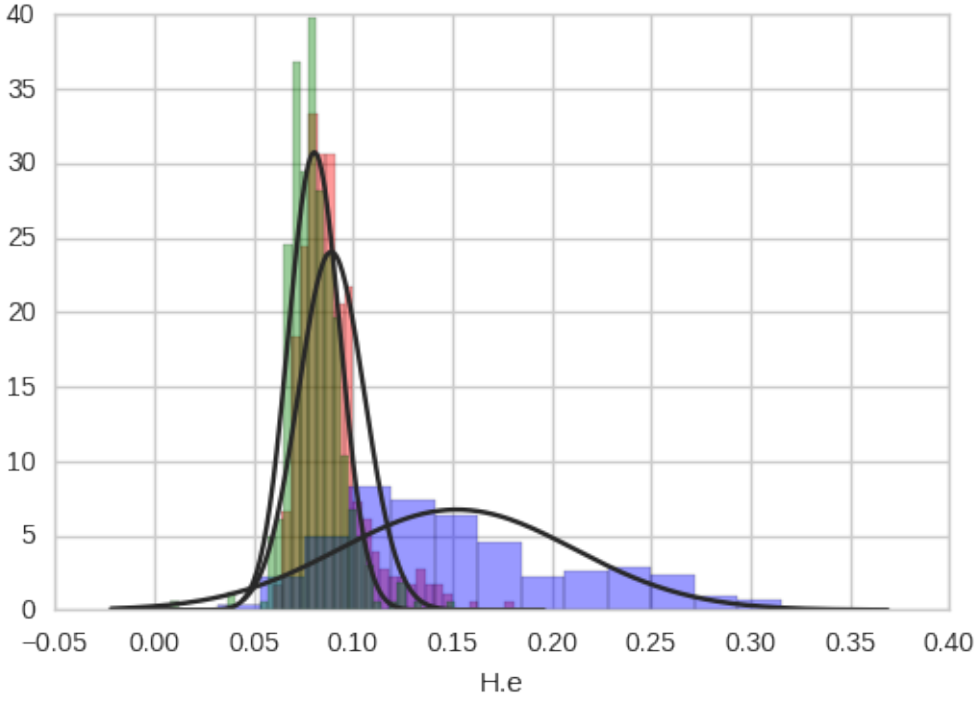
```
In [1]: import pandas as pd
        from pandas import Series,DataFrame
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        sns.set_style('whitegrid')
        %matplotlib inline
        # Reading the Password data into dataframes
        df = pd.read_csv("../data/PasswordData.csv")
        #getting to know about the data
        # df.info()
```

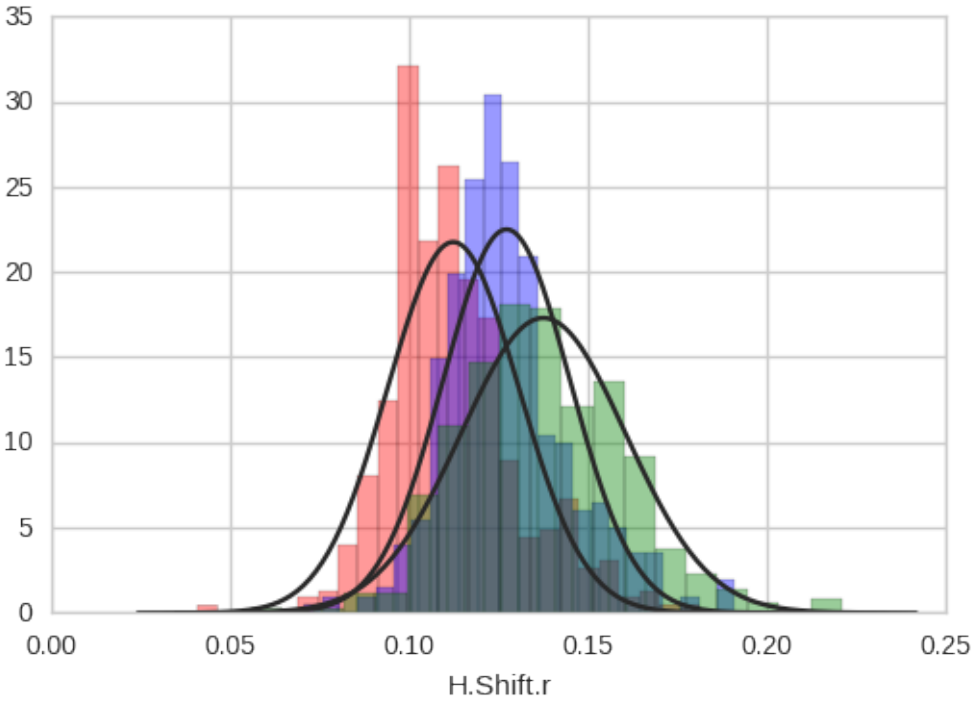
1.2 Gaussian Model for first 3 users

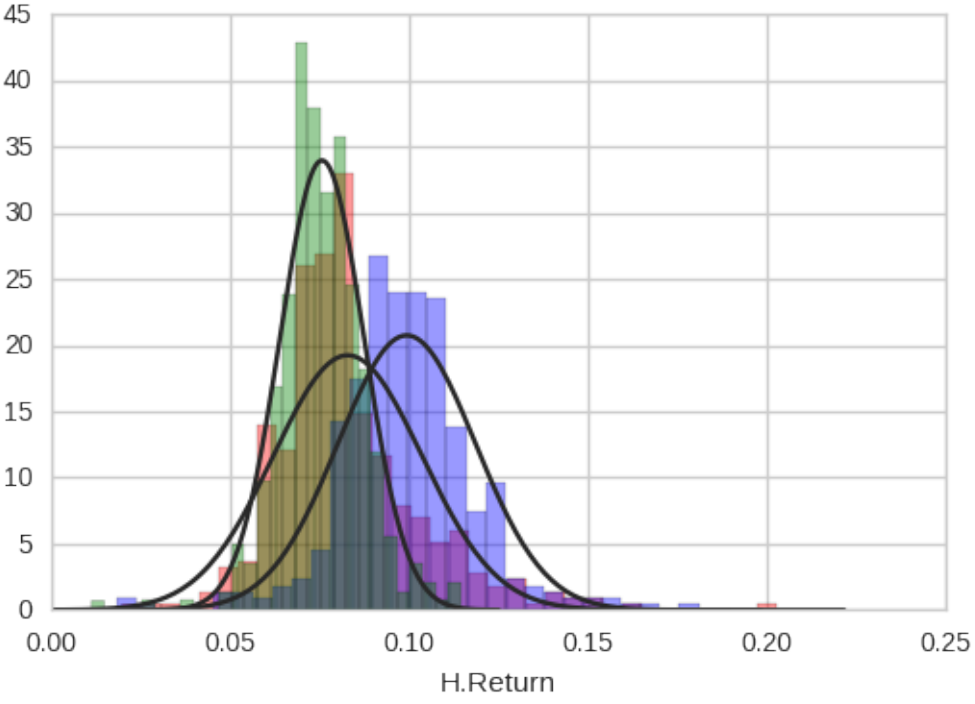
```
In [11]: from scipy.stats import norm
         for col in ["H.t","H.i","H.e","H.Shift.r","H.Return"]:
             sns.distplot(user_1[col], fit=norm, kde=False,color='red')
             sns.distplot(user_2[col], fit=norm, kde=False,color='blue')
             sns.distplot(user_3[col], fit=norm, kde=False,color='green')
             plt.figure()
```




40
35
30
25
20
15
10
5
0
0.00
0.05
0.10
0.15
0.20
0.25
0.30
H.i


40
35
30
25
20
15
10
5
0
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
H.e

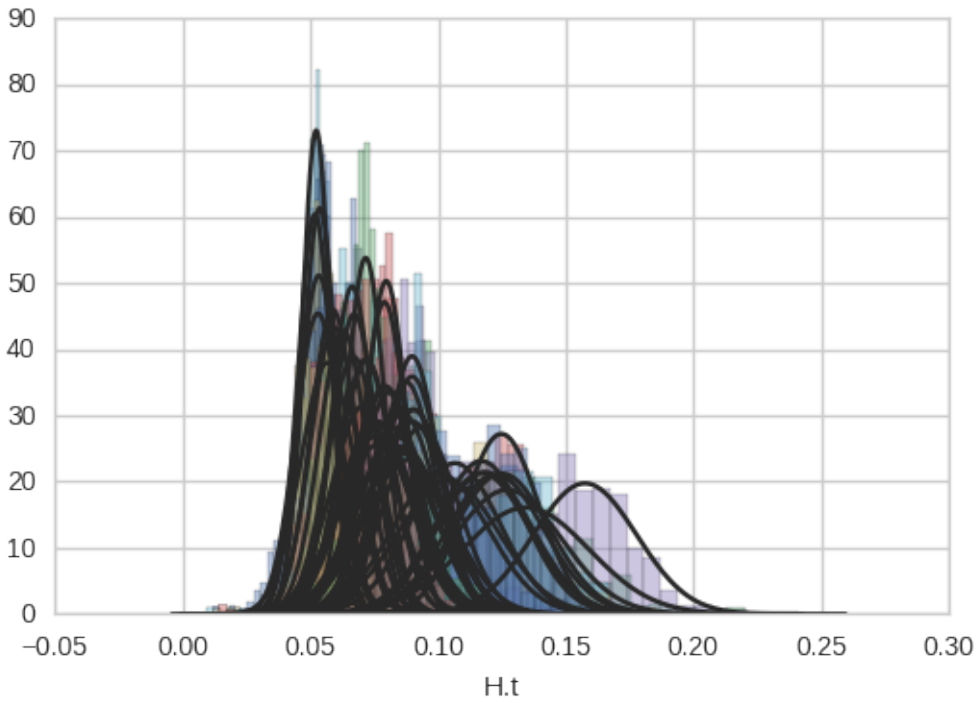

35
30
25
20
15
10
5
0
0.00
0.05
0.10
0.15
0.20
0.25
H.Shift.r

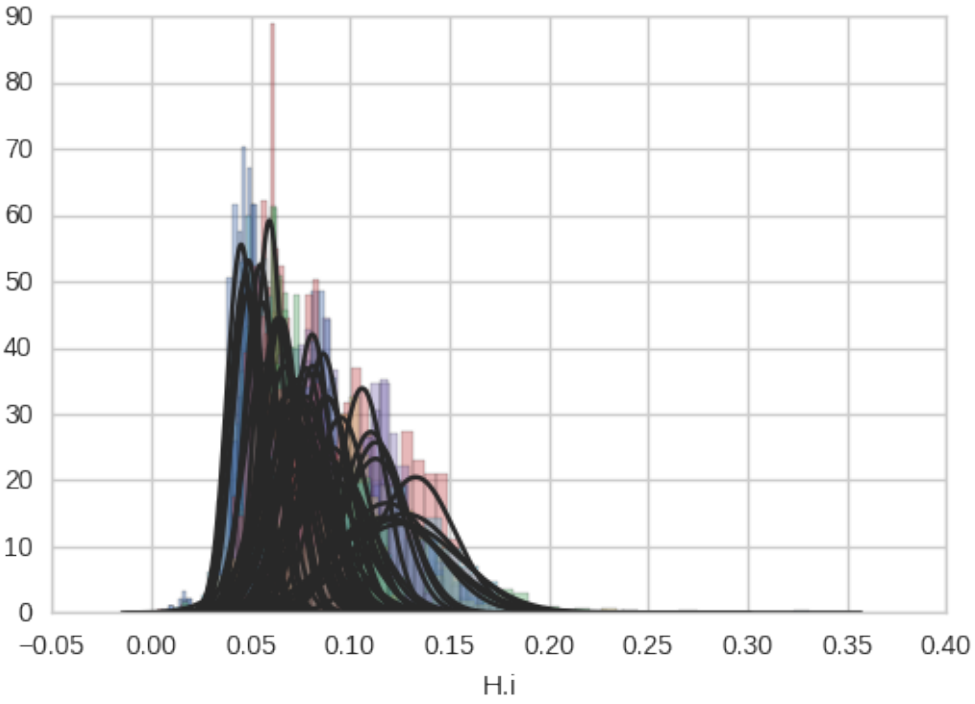

45
40
35
30
25
20
15
10
5
0
0.00
0.05
0.10
0.15
0.20
0.25
H.Return

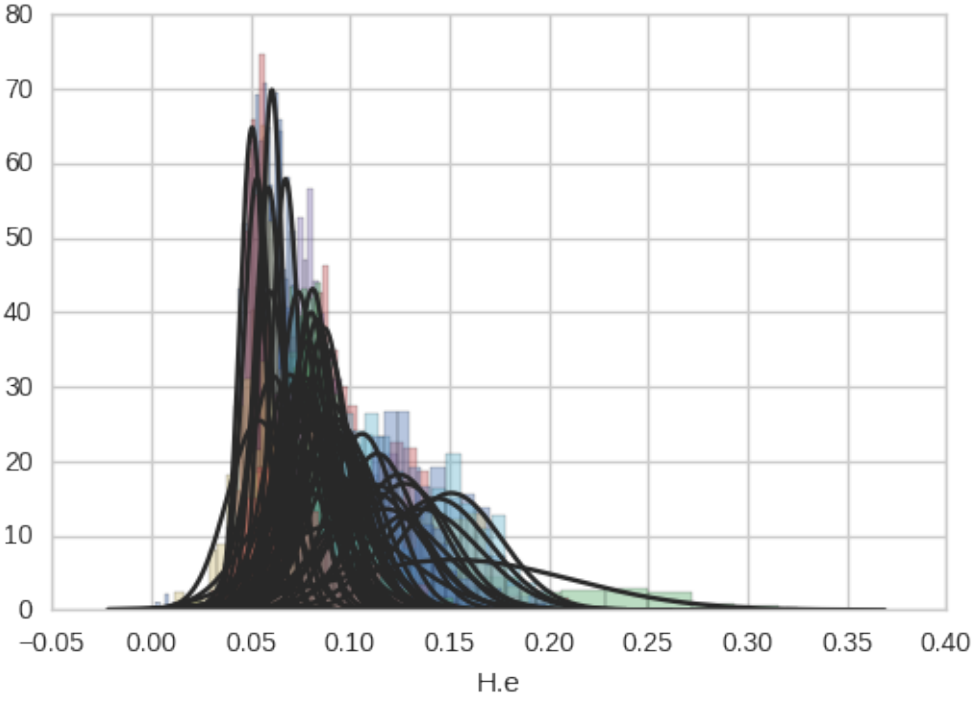
```
<matplotlib.figure.Figure at 0x7f55767d5b38>
```

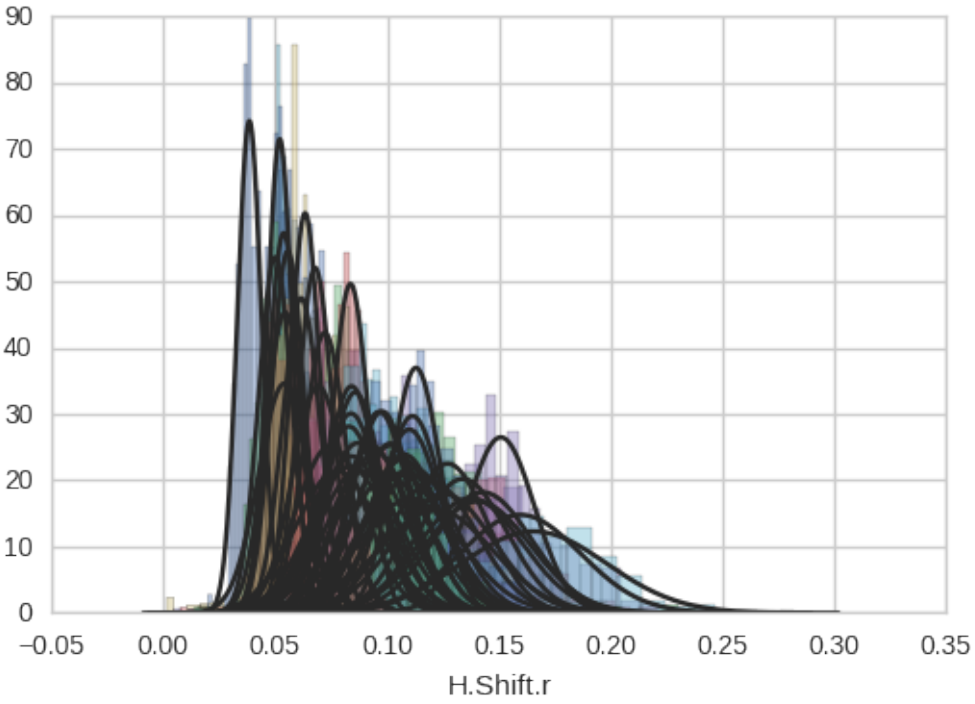
All the plots above show that there is a difference in the Gaussian models for the users. Some of keystrokes like “H.t”,“H.i”,“H.e”,“H.Shift.r”,“H.Return” show significant difference. To check if the differnce is only with the first three users or with all, plotting the Gaussian models for all the 50 users for “H.t”,“H.i”,“H.e”,“H.Shift.r”,“H.Return”

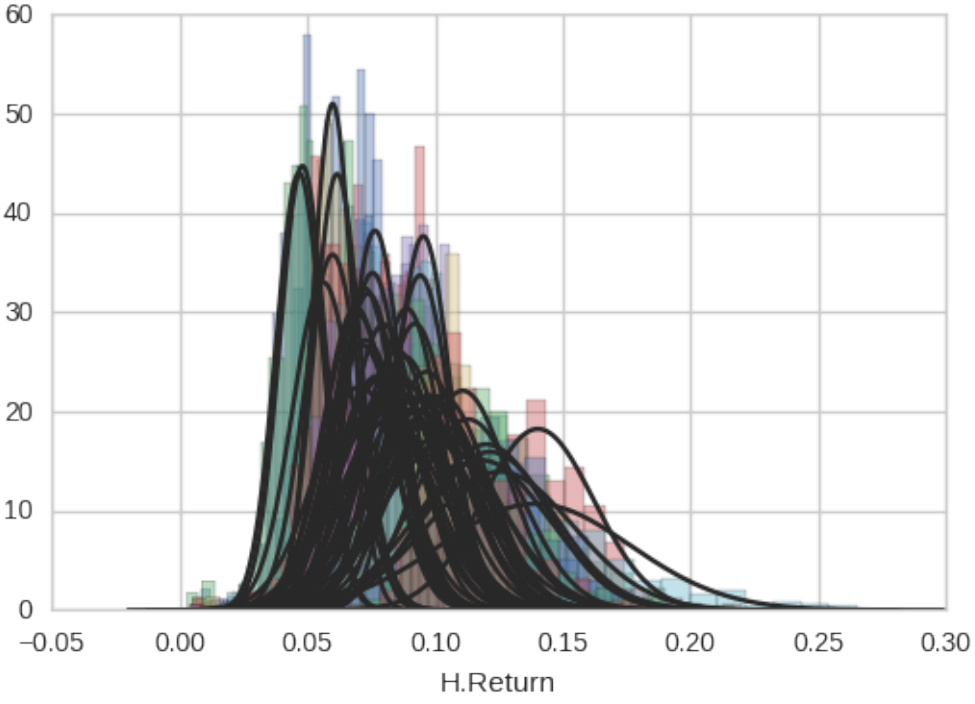
```
In [57]: from scipy.stats import norm
         for col in ["H.t","H.i","H.e","H.Shift.r","H.Return"]:
             for i in range (0,len(df),400):
                 tmp=df[i:i+400]
                 sns.distplot(tmp[col], fit=norm, kde=False)
             plt.figure()
```




90
80
70
60
50
40
30
20
10
0
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
H.i


80
70
60
50
40
30
20
10
0
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
H.e


90
80
70
60
50
40
30
20
10
0
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
H.Shift.r


60
50
40
30
20
10
0
−0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
H.Return

```
<matplotlib.figure.Figure at 0x7f97a230bb70>
```

1.2.1 Gaussian models can be used to differentiate the users based on the typing pattern.