ITI105 Milestone Report

User Authentication Using Classical Machine Learning: Leveraging Key Typing Dynamics Behavior

Allen Lee and Jacob Abraham

8-Aug-2023

1. Problem statement and solution

Computing devices, including mobile phones, use various biometric authentication methods like fingerprints or facial features to identify users. However, these methods rely on specific hardware, which can increase the overall cost. An economical alternative is to authenticate users based on their behaviour, such as typing dynamics.

Keystroke dynamics, also known as keystroke biometrics, pertain to the comprehensive timing data that precisely records the moment each key is pressed and released as an individual types on a computer keyboard. These dynamics offer valuable insights that aid in user authentication. Through capturing the intervals between key presses, key hold durations, and the periods between key releases and next key presses, significant user insights can be derived. When the user logs in again, companies can compare their present typing pattern with their past patterns, allowing for authentication to distinguish legitimate users from potential fraudulent ones.

Conventional machine learning will be applied to generate models that can infer keystroke dynamics and then authenticate users.

2. Description of data

Critical information that is captured during typing a password are:

a. Hold duration (H): Time from when a key was pressed to when it was released.
b. Up-to-Down (UD) duration: Time from when key1 was released to when key2 was pressed.
c. Down-to-Down (DD) duration: Time from when *key1* was pressed to when *key2* was pressed.

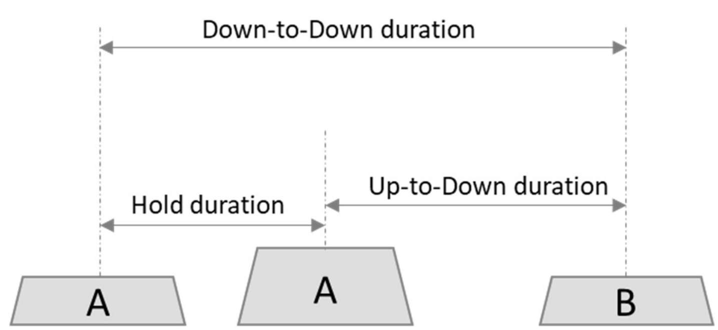


Figure 1. Critical keystroke timings

2.1. Benchmark dataset

- Benchmark dataset was downloaded from Carnegie Melon University http://www.cs.cmu.edu/~keystroke/.
- Data consist of keystroke-timing information from 51 subjects (typists), each typing same 10-character password ".tie5Roanl" 400 times (in 8 sessions, with 50 repetitions per session).
- Total number of samples is 20400.
- Features and their data types are:

```
Data columns (total 34 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   subject         20400 non-null  object
 1   sessionIndex    20400 non-null  float32
 2   rep             20400 non-null  float32
 3   H.period        20400 non-null  float32
 4   DD.period.t     20400 non-null  float32
 5   UD.period.t     20400 non-null  float32
 6   H.t             20400 non-null  float32
 7   DD.t.i          20400 non-null  float32
 8   UD.t.i          20400 non-null  float32
 9   H.i             20400 non-null  float32
 10  DD.i.e          20400 non-null  float32
 11  UD.i.e          20400 non-null  float32
 12  H.e             20400 non-null  float32
 13  DD.e.five       20400 non-null  float32
 14  UD.e.five       20400 non-null  float32
 15  H.five          20400 non-null  float32
 16  DD.five.Shift.r 20400 non-null  float32
 17  UD.five.Shift.r 20400 non-null  float32
 18  H.Shift.r       20400 non-null  float32
 19  DD.Shift.r.o    20400 non-null  float32
 20  UD.Shift.r.o    20400 non-null  float32
 21  H.o             20400 non-null  float32
 22  DD.o.a          20400 non-null  float32
 23  UD.o.a          20400 non-null  float32
 24  H.a             20400 non-null  float32
 25  DD.a.n          20400 non-null  float32
 26  UD.a.n          20400 non-null  float32
 27  H.n             20400 non-null  float32
 28  DD.n.l          20400 non-null  float32
 29  UD.n.l          20400 non-null  float32
 30  H.l             20400 non-null  float32
 31  DD.l.Return     20400 non-null  float32
 32  UD.l.Return     20400 non-null  float32
 33  H.Return        20400 non-null  float32
```

Note:
H.period: during the period "." Key was held.
DD.period.t: Time from when period "." was pressed to when "t" key was pressed.
UD.period.t: Time from when period "." was released to when "t" key was pressed.

	Features		
Keystroke	Hold time	Down-to-Down time	Up-to-Down Time
.	H.period	DD.period.t	UD.period.t
t	H.t	DD.t.i	UD.t.i
i	H.i	DD.i.e	UD.i.e
e	H.e	DD.e.five	UD.e.five
5	H.five	DD.five.Shift.r	UD.five.Shift.r
R	H.Shift.r	DD.Shift.r	UD.Shift.r
o	H.o	DD.o.a	UD.o.a
a	H.a	DD.a.n	UD.a.n
n	H.n	DD.n.l	UD.n.l
l	H.l	DD.l.Return	UD.l.Return
Enter	H.Return		

2.2. Own hybrid dataset

- Create own key-sniffer program to collect 400 samples each from two new subjects (Allen and Jacob).

- A hybrid dataset will be created by using first 20 subjects' data from the benchmark data set augmented with the additional data from the two new subjects.

3. Conventional Machine Learning with benchmark dataset

3.1. Exploratory Data Analysis (EDA) and Visualization

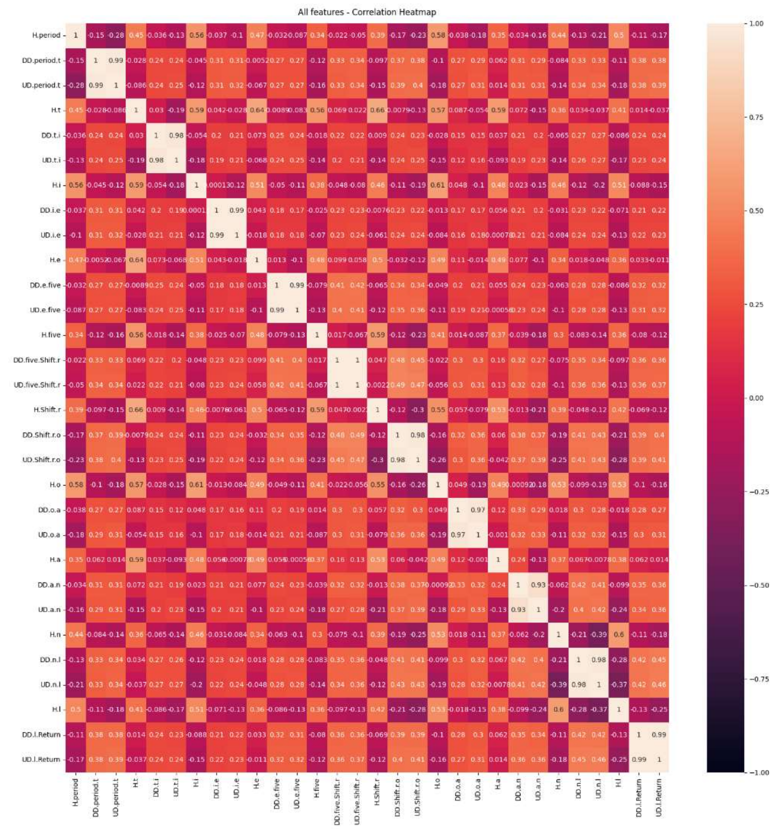
- Total number of samples is 20400.
- Total number of columns is 33.

```
1 df_bench.columns
```

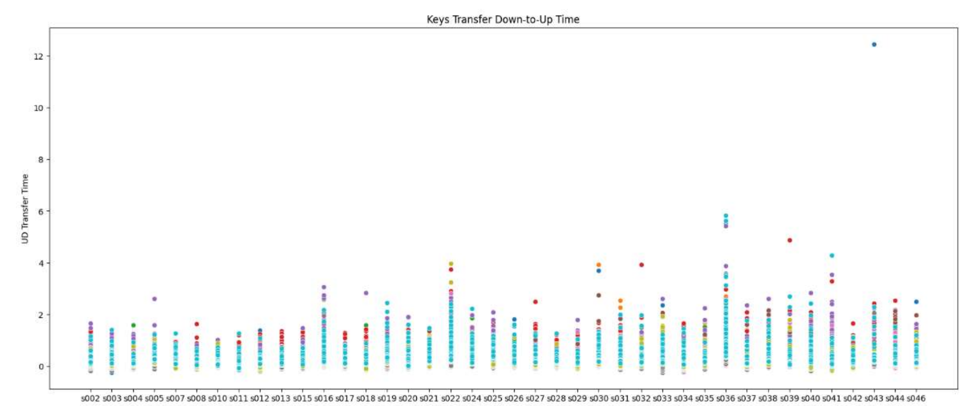
```
Index(['subject', 'sessionIndex', 'rep', 'H.period', 'DD.period.t',
       'UD.period.t', 'H.t', 'DD.t.i', 'UD.t.i', 'H.i', 'DD.i.e', 'UD.i.e',
       'H.e', 'DD.e.five', 'UD.e.five', 'H.five', 'DD.five.Shift.r',
       'UD.five.Shift.r', 'H.Shift.r', 'DD.Shift.r.o', 'UD.Shift.r.o', 'H.o',
       'DD.o.a', 'UD.o.a', 'H.a', 'DD.a.n', 'UD.a.n', 'H.n', 'DD.n.l',
       'UD.n.l', 'H.l', 'DD.l.Return', 'UD.l.Return', 'H.Return'],
      dtype='object')
```

- Target is column "subject", and its data type is "object". Data type in other columns are "float".
- Columns "sessionIndex" and "rep" are not used as it will not affect model training and testing.
- There are no missing values and "NaN" (refer to section 2.1).
- Columns of "Hold" duration shows low standard deviation compared to their corresponding columns of "Up-to-Down".
- Values in all columns of "Down-to-Down" are the sum of their corresponding values in columns of "Hold" and "Up-to-Down". Columns of "Up-to-Down" shows strong correlation with column "Down-to-Down". Calculated correlation coefficients for these pairs of 'Down-to-Down' and 'Up-to-Down' for each character keypress are above 0.9 as shown below.
- Hence all columns prefixed with "DD", are to be dropped to reduce the effects on model training due to correlated features. This reduces the number of features from 31 to 21.

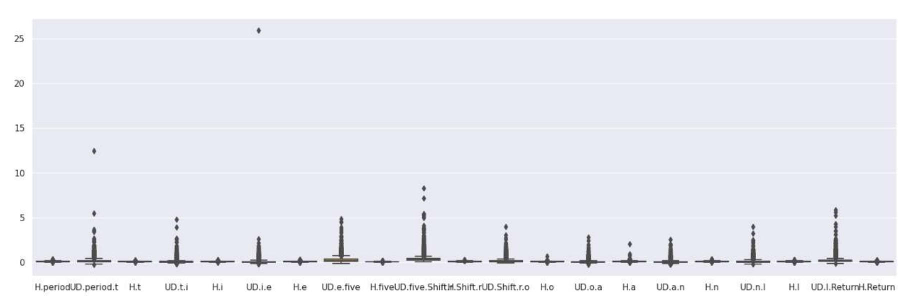
Feature #1	Feature #2	Correlation Coefficient (r)
DD.period.t	UD.period.t	0.980
DD.t.i	UD.t.i	0.968
DD.i.e	UD.i.e	0.978
DD.e.five	UD.e.five	0.993
DD.five.Shift.r	UD.five.Shift.r	0.995
DD.Shift.r.o	UD.Shift.r.o	0.963
DD.o.a	UD.o.a	0.971
DD.a.n	UD.a.n	0.938
DD.n.l	UD.n.l	0.977
DD.l.Return	UD.l.Return	0.990



- It is also noted that “Up-to-Down” values can be negative. The range of “UD.period.t” is from -0.24 to 12.45.



- All the “Up-to-Down” features have outliers. Will run experiments to know their effects on the model accuracy.



- All numerical features of the dataset were scaled to ensure that the features have comparable magnitudes. Jacob has used Min-Max scaler while Allen has used Standard scaler. Experiments will be run to study the effect of type of scaler on the model accuracy.

3.2. Feature Engineering

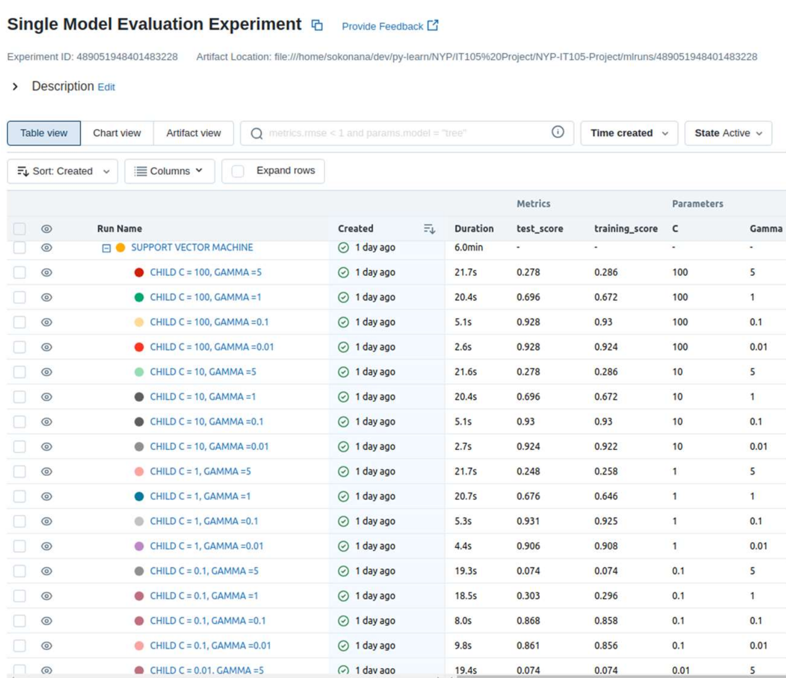
- Generating extra attributes is achievable through the calculation of the ratio between the "Hold" time and the total time for all characters present in the password.
- Experiments with engineered features will be done then to see their effects on model accuracies.

3.3. Modelling and experimental results

- Experiments were conducted with various models with hyper tuning and the results are summarized below.

			Best accuracy		
			Jacob		Allen
Experiment #	Classifier Model	Best hyperparameters	Training	Test	Test
1	Decision Tree	max_depth=15	0.83	0.69	0.85
2	K-Nearest Neighbours	n_neighbors=5	0.85	0.78	0.90
3	Support Vector Machine	C=20, gamma=10, kernel='rbf'	0.99	0.87	0.93
4	Gaussian Naive Bayes	var_smoothing=1e-09	0.74	0.74	0.83
5	Logistic Regression	C=1000; max_iter=5000	0.85	0.84	0.90
6	Random Forest	n_estimators=300, max_depth=6	0.75	0.73	0.90
7	Gradient Boosting (base: Decision Tree)	n_estimators=50 , learning_rate=0.05	0.93	0.86	0.96
8	Voting Classifier				0.93
		Average	0.85	0.79	0.90

- The best model accuracy obtained so far is with ensemble Gradient Boosting (with the default base classifier Decision Tree).
- Conducting experiments is a time-consuming process due to the significant amount of machine time needed for hyperparameter tuning.
- Experiment runs are logged using "MLFlow" as shown below.


Single Model Evaluation Experiment Provide Feedback

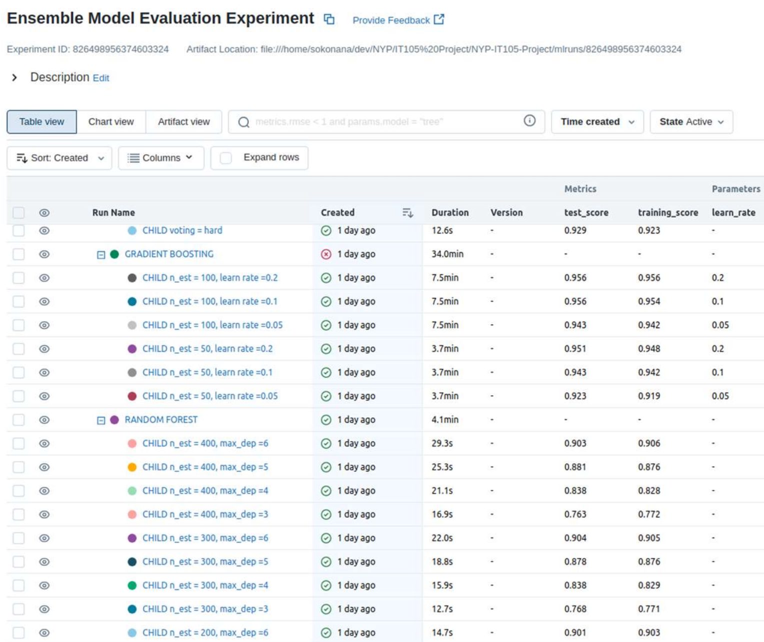
Experiment ID: 489051948401483228 Artifact Location: file:///home/sokonana/dev/py-learn/NYP/IT105%20Project/NYP-IT105-Project/mlruns/489051948401483228

Description Edit

Table view | Chart view | Artifact view | metrics.rmse < 1 and params.model = "tree" | Time created | State Active

Sort: Created | Columns | Expand rows

Run Name	Created	Duration	test_score (Metrics)	training_score (Metrics)	C (Parameters)	Gamma (Parameters)
SUPPORT VECTOR MACHINE	1 day ago	6.0min	-	-	-	-
CHILD C = 100, GAMMA =5	1 day ago	21.7s	0.278	0.286	100	5
CHILD C = 100, GAMMA =1	1 day ago	20.4s	0.696	0.672	100	1
CHILD C = 100, GAMMA =0.1	1 day ago	5.1s	0.928	0.93	100	0.1
CHILD C = 100, GAMMA =0.01	1 day ago	2.6s	0.928	0.924	100	0.01
CHILD C = 10, GAMMA =5	1 day ago	21.6s	0.278	0.286	10	5
CHILD C = 10, GAMMA =1	1 day ago	20.4s	0.696	0.672	10	1
CHILD C = 10, GAMMA =0.1	1 day ago	5.1s	0.93	0.93	10	0.1
CHILD C = 10, GAMMA =0.01	1 day ago	2.7s	0.924	0.922	10	0.01
CHILD C = 1, GAMMA =5	1 day ago	21.7s	0.248	0.258	1	5
CHILD C = 1, GAMMA =1	1 day ago	20.7s	0.676	0.646	1	1
CHILD C = 1, GAMMA =0.1	1 day ago	5.3s	0.931	0.925	1	0.1
CHILD C = 1, GAMMA =0.01	1 day ago	4.4s	0.906	0.908	1	0.01
CHILD C = 0.1, GAMMA =5	1 day ago	19.3s	0.074	0.074	0.1	5
CHILD C = 0.1, GAMMA =1	1 day ago	18.5s	0.303	0.296	0.1	1
CHILD C = 0.1, GAMMA =0.1	1 day ago	8.0s	0.868	0.858	0.1	0.1
CHILD C = 0.1, GAMMA =0.01	1 day ago	9.8s	0.861	0.856	0.1	0.01
CHILD C = 0.01, GAMMA =5	1 day ago	19.4s	0.074	0.074	0.01	5



Run Name	Created	Duration	Version	Metrics test_score	Metrics training_score	Parameters learn_rate
CHILD voting = hard	1 day ago	12.6s	-	0.929	0.923	-
GRADIENT BOOSTING	1 day ago	34.0min	-	-	-	-
CHILD n_est = 100, learn rate =0.2	1 day ago	7.5min	-	0.956	0.956	0.2
CHILD n_est = 100, learn rate =0.1	1 day ago	7.5min	-	0.956	0.954	0.1
CHILD n_est = 100, learn rate =0.05	1 day ago	7.5min	-	0.943	0.942	0.05
CHILD n_est = 50, learn rate =0.2	1 day ago	3.7min	-	0.951	0.948	0.2
CHILD n_est = 50, learn rate =0.1	1 day ago	3.7min	-	0.943	0.942	0.1
CHILD n_est = 50, learn rate =0.05	1 day ago	3.7min	-	0.923	0.919	0.05
RANDOM FOREST	1 day ago	4.1min	-	-	-	-
CHILD n_est = 400, max_dep =6	1 day ago	29.3s	-	0.903	0.906	-
CHILD n_est = 400, max_dep =5	1 day ago	25.3s	-	0.881	0.876	-
CHILD n_est = 400, max_dep =4	1 day ago	21.1s	-	0.838	0.828	-
CHILD n_est = 400, max_dep =3	1 day ago	16.9s	-	0.763	0.772	-
CHILD n_est = 300, max_dep =6	1 day ago	22.0s	-	0.904	0.905	-
CHILD n_est = 300, max_dep =5	1 day ago	18.8s	-	0.878	0.876	-
CHILD n_est = 300, max_dep =4	1 day ago	15.9s	-	0.838	0.829	-
CHILD n_est = 300, max_dep =3	1 day ago	12.7s	-	0.768	0.771	-
CHILD n_est = 200, max_dep =6	1 day ago	14.7s	-	0.901	0.903	-

4. Conventional Machine Learning with hybrid dataset

- Currently performing data collection
- EDA, Feature Engineering, Visualization, and modelling will be done based on the model that has provided best accuracies (train and test) with the benchmark dataset.

5. Further actions

- Compare the accuracies obtained by Allen and Jacob.
- Experiment with outliers, scalers, and feature engineering.
- Complete the data collection from 2 new users.
- EDA, feature engineering and visualization with hybrid dataset.
- Model selection and hyperparameter tuning with hybrid dataset.
- Model deployment and inference.

6. Contributions

Jacob:

- Sourced initial c code for “Key-Sniffer.
- Data visualization and Model Evaluation with Decision Tree, K-Nearest Neighbors, Support Vector Machine, Gaussian Naive Bayes, Logistic Regression, Random Forest, and Gradient Boosting.
- Platform: Google CoLab

Allen:

- Upgrading and fine tuning of “Key-Sniffer” program to collect keystroke data.
- Data visualization and Model Evaluation with Decision Tree, K-Nearest Neighbors, Naive Bayes, Logistic Regression, Support Vector Machine, Random Forest, Gradient Boosting, and Voting Classifier
- Platform: Jupyter Notebook

7. References

[1] Keystroke Dynamics - Benchmark Data Set: http://www.cs.cmu.edu/~keystroke/

[2] KDA on benchmark: https://www.kaggle.com/code/ashusrivastava/kda-on-benchmark

[3] Keystroke Dynamics Analysis and Prediction w/ XGB: https://www.kaggle.com/code/kartik2112/keystroke-dynamics-analysis-and-prediction-w-xgb

[4] Keystroke Dynamics Analysis and Prediction — Part 1/2 (EDA): https://towardsdatascience.com/keystroke-dynamics-analysis-and-prediction-part-1-eda-3fe2d25bac04