User Identification based on Mouse Cursor Movements

Aaditya Gupta
2020552
Rohan Kulkarni
2020537
Harjeet Singh Yadav
2020561
Hemang Dahiya
2020435

Abstract

According to Cornell University's research, there is a unique way in which every person uses their mouse pointer while visiting any website. Taking inspiration from this, we are developing and training our model on the cursor movements data of registered users. Then, by leveraging the classification process of our model, we would be able to identify fraudulent user activity and hence, prevent that particular user from performing any similar activity in future.

1. Introduction

Credentials are sensitive information that should not be shared with anyone. However, with the coming of AI bots and advances scammers, it has become increasingly easier to pose as someone else and enter the account as someone else. Hence the problem is: "How to add to security check mechanisms of existing login and payment portals?"

2. Literature Survey

The following are the research papers that we studied for our project.

2.1. Intrusion Detection Using Mouse Dynamics

This paper by Margit Antal, Elod Egyed-Zsigmond talks about this exact problem and proposes an intrusion detection model using a variety of features like 'elapsed time', 'distance traveled', average velocity in x', 'average velocity in y' etc. They have also published a collection of datasets namely "Balabit-Chaoshen-DFL" datasets, and we are using the same for our project.

2.2. Clustering Web Users by Mouse Movement to Detect Bots and Botnet Attacks

This paper by Justin Morgan: The objective of this project is to present an unsupervised approach for website administrators to detect web bots. The thesis presents an approach to cluster users and then tag them as being actual registered users, or as bots. K means clustering was used to cluster malicious traffic flows.

3. Dataset details & Pre-processing

3.1. Pre-processing steps:

1. Shuffling the dataset: The dataset had session records for users in a continuous fashion. We shuffle the data to increase the efficiency of our model.
2. Filling the null values: Incidentally there were no null values in the dataset.
3. Normalization: The dataset given to us had varying values for different attributes, and hence normalizing the dataset was a necessity.
4. Removing features which have 0 or less importance like:
   (a) minvx : 0
   (b) minvy : 0.003
   (c) max_curvature : 0.007

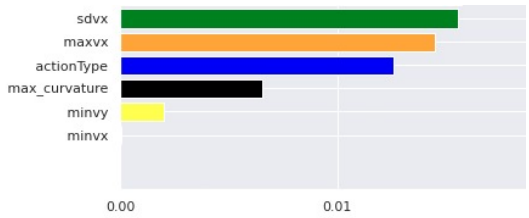


Figure 1. Comparative importance

3.2. Dataset description

We have used:

1. The Balabit Mouse Challenge data set
2. Chao Shen's Mouse Dynamics data set
3. DFL Mouse Dynamics data set

The dataset consists of 39 attributes, labeled for 28 users.

- The type of action (MM: Mouse movement, CC: Point click, DD: Drag & Drop)
- Traveled Distance, (total distance by mouse)
- Curvature (average, std, min, max curvature)
- Velocity (average, std, min, the max velocity of the mouse)
- Omega/Angular velocity (average, std, min, max omega)
- Deviation (largest deviation while using the mouse)
- Jerk (average, std, min, max jerk)
- Sum of angles (summation of angles of all the trajectories)
- number of points, which means the number of mouse events contained in an action
- a_beg_time, which is the acceleration time at the beginning of a mouse movement
- Directions (dir of end to end line), there are 8 directions defined as follows:-

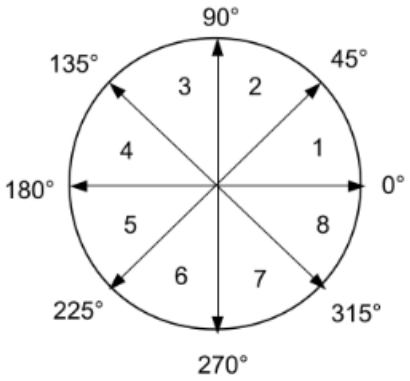


Figure 2. Directions

4. Methodology & Model-details

Our overall objective is to classify a given input as either one of the registered users or as an outsider. For that, we put a threshold over the predicted probabilities of each model. By fine-tuning the values, we found that a threshold value of 0.8 gives better results, i.e. we only consider those outputs from classifiers which are predicted with a probability

$$P >= 0.7 \quad (1)$$

Name	Description	#features
v_x	mean, std, min, max	4
v_y	mean, std, min, max	4
v	mean, std, min, max	4
a	mean, std, min, max	4
j	mean, std, min, max	4
ω	mean, std, min, max	4
c	mean, std, min, max	4
type	MM, PC, DD	1
elapsed time	$t_n - t_1$	1
trajectory length	s_n	1
dist_end_to_end	$\|P_1P_n\|$	1
direction	see Fig 6	1
straightness	$\frac{\|P_1P_n\|}{s_n}$	1
num_points	n	1
sum_of_angles	$\sum_{i=1}^{n}\Theta_i$	1
largest_deviation	$max_i\{d(P_i, \|P_1P_n\|)\}$	1
sharp angles	$\#\{\Theta_i \| \Theta_i < TH\}$	1
a_beg_time	accel. time at the beginning	1
Total		39

Figure 3. Attributes
source: Intrusion Detection Using Mouse Dynamics

4.1. Decision Tree Classifier

Decision Tree Classifier is a tree based model that has hyperparameters including max depth, max features used for splitting. Hyperparameters: [criterion= 'gini', splitter= 'best',]

4.2. Random forest Classifier

A type of ensemble learning, which uses a number of weak classifiers(decision trees) and then gives the label based on majority voting. Hyperparameters: RandomizedSearchCV for hyperparameter tuning

4.3. ADAboost Classifier

Also a type of ensemble learning, where trees are not independent of each other, and grow on top of each other. Hyperparameters: [base_estimator = decision_tree, n_estimators = 150, learning_rate=0.1]

4.4. Multi Layer Perceptron

A completely connected class of feedforward artificial neural networks is known as a multilayer

perceptron (MLP) (ANN). Hyperparameters [hidden_layer_sizes=(256,128,128,64),random_state=1, max_iter=300, activation = 'tanh']

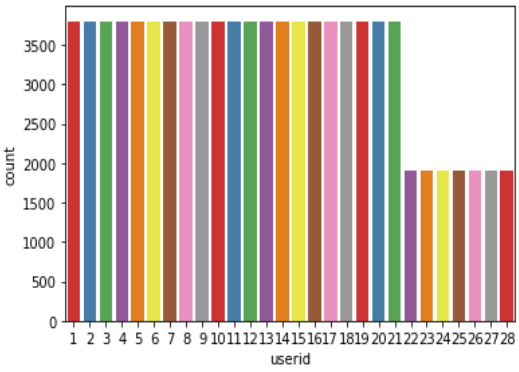
4.5. GBDT

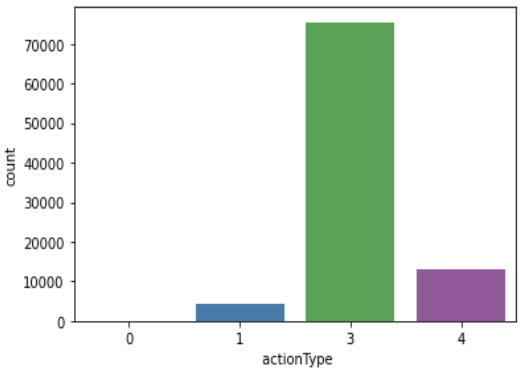
Combining decision trees with a technique called boosting. Thus, GBDT is also an ensemble method. Boosting means combining a learning algorithm in series to achieve a strong learner from many sequentially connected weak learners.

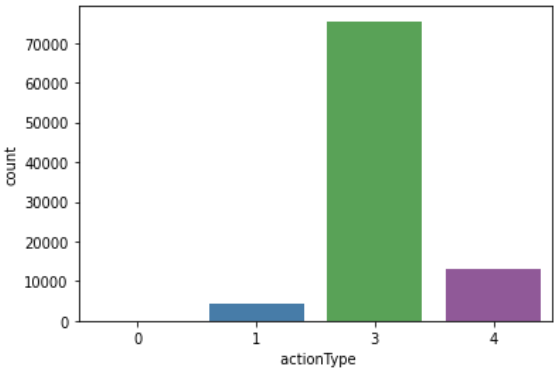
4.6. Results & Analysis

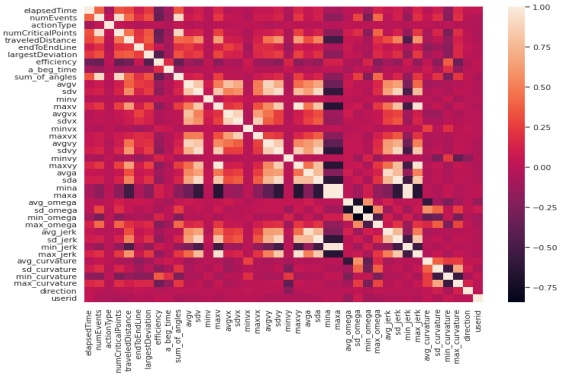
4.6.1 Analyzing the data

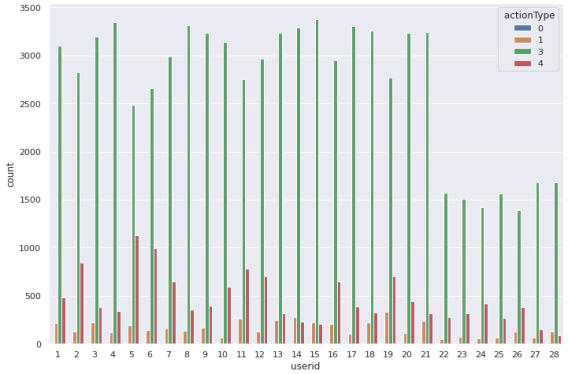
A number of insights from the data have been given here for visual representation. A notable analysis comes from the Balabit dataset and Chao shen dataset. When we apply t-SNE on the data, we observe the formation of groups or clusters, signifying that the dataset is separable(mostly) for the 28 users. t-SNE is a tool to visualize high-dimensional data in lower dimensions without losing much information.

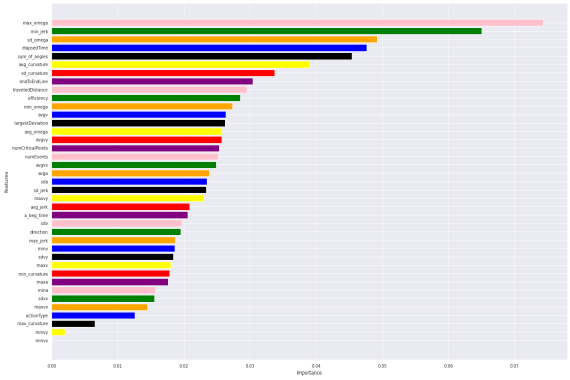












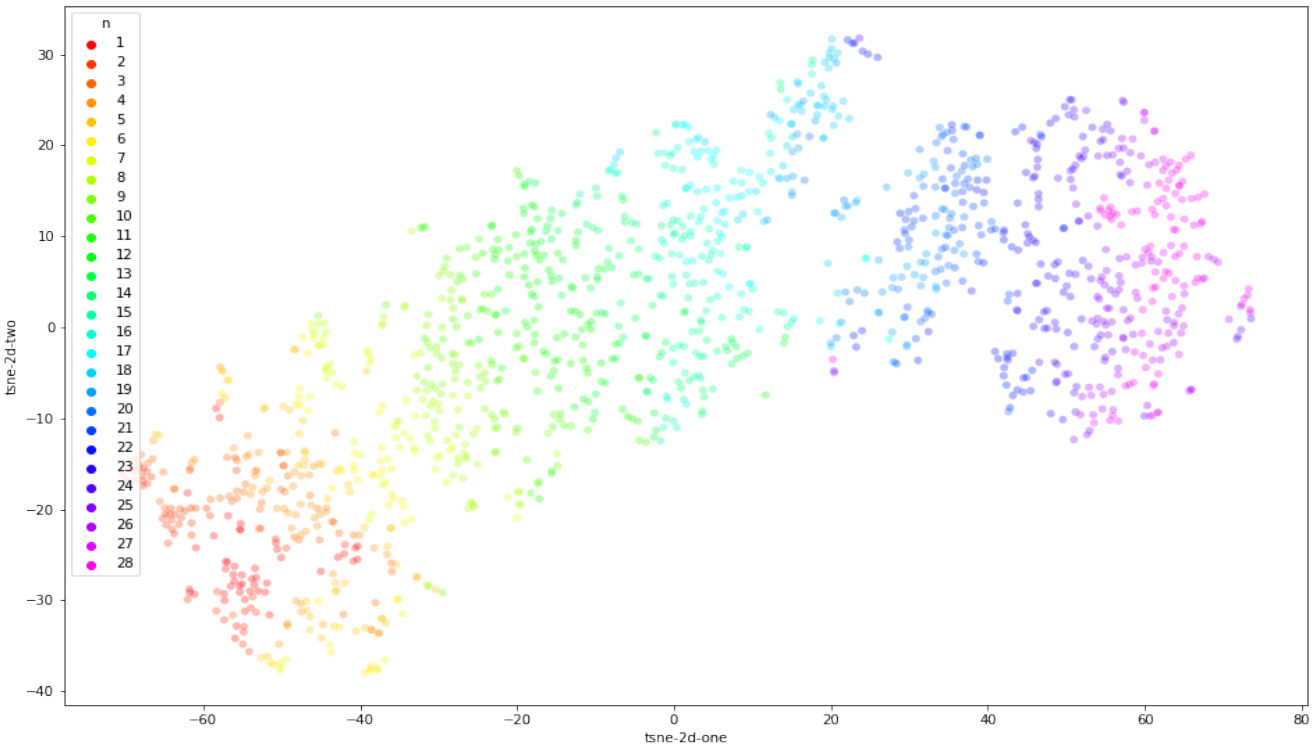


Figure 4. t-SNE plot for Chao Shen's Mouse Dynamics data, perplexity=30

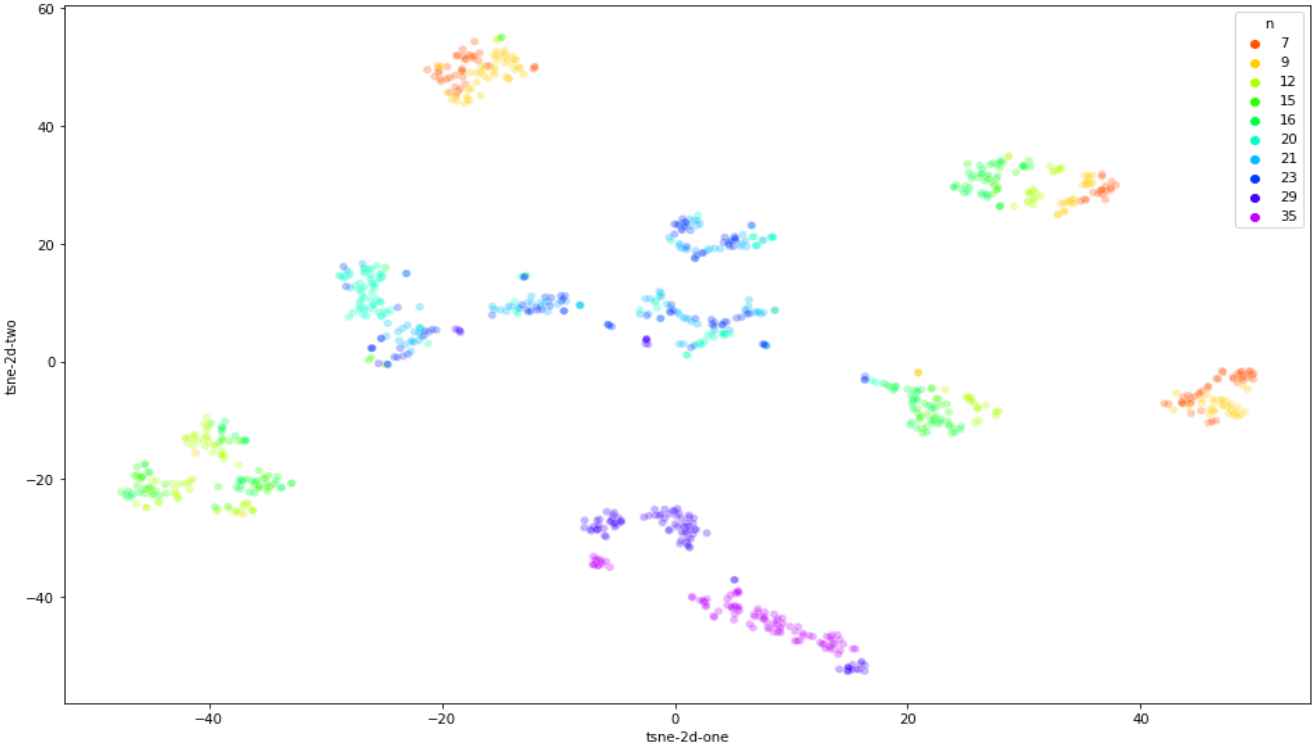


Figure 5. t-SNE plot for Balabit's Mouse Dynamics data, perplexity=30

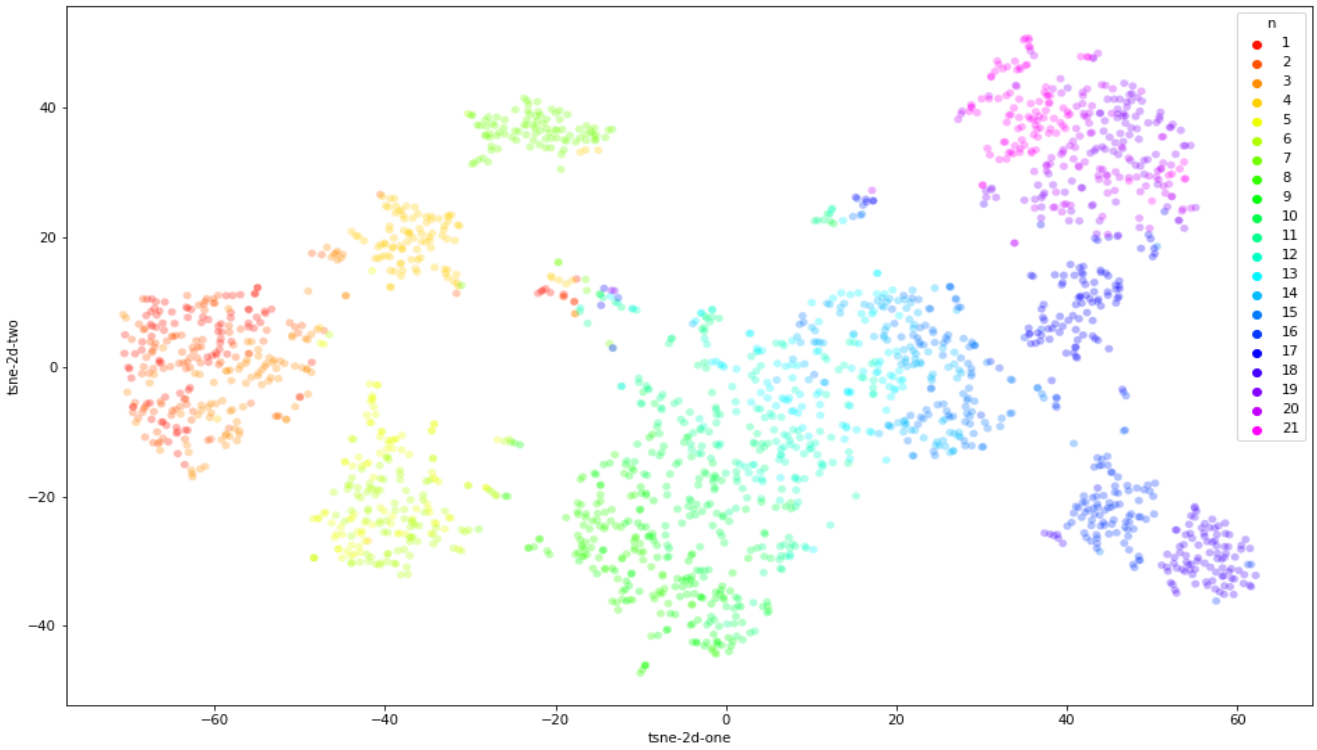


Figure 6. t-SNE plot for DFL Mouse Dynamics data, perplexity=30

4.6.2 Results

From a multitude of ML and DL classifiers, we found out that ADABoost with a threhold of 0.7 gave the best accuracy on the test(evaluation dataset).

Decision Trees appear to overfit, and hence give a small bias but a larger variance.

Random Forests, using these decision trees appeared to over fit, however, the accuracy on the test set increased.

Random forests with a threhold gave better accuracy as we segregated the legitimate outputs.

ADABoost with threshold performs better than RF because here, the weak classifiers are not independent on each other, rather grow on top of each other. Infact, the final output is the weighted sum of these classifiers. ADABoost works on set of weak classifiers which learn from each other, and hence give better outcome. The threshold of 0.7 means that we have taken just those predictions for which the output class predicted was predicted with a probability of atleast 0.7. This helped us to remove the poorly classified data points.

5. Conclusion

From the entire project we conclude that ADA boost classifier with the given hyperparameters and set at a threshold of 0.7 classified about 90% of users correctly. We can use this model on any real time interface, gather the cursor coordinates of registred users for each session, and then apply the model to predict whether a current user is registered in the system or not.

	Training Data			Test Data		
Technique	Precision	Accuracy	F1 Score	Precision	Accuracy	F1 Score
MLP	0.85	0.86	0.86	0.64	0.64	0.64
Decision Tree	0.99	0.99	0.99	0.72	0.71	0.72
Random Forest	0.99	0.99	0.98	0.76	0.77	0.77
Random Forest (threshold = 0.2)	0.98	0.99	0.99	0.88	0.88	0.88
AdaBoost	0.76	0.77	0.77	0.77	0.77	0.77
AdaBoost(threshold = 0.7)	0.98	0.99	0.99	0.89	0.89	0.89
GBDT	0.49	0.48	0.49	0.43	0.42	0.41
LightGBM	0.79	0.79	0.78	0.62	0.61	0.61

Figure 7. Model comparisons