STAT 702 - Final Project

Tuesday April 17, 2018

Contents

Abstract **2**

Introduction **2**

Methods and Results **2**

Exploratory Data Analysis and Variable Selection 2
Model Methodology 8
Parametric Modelling 9
Non- and Semi-Parametric 9
Support Vector Machine 11
Final Model with Unknown Data Set 11

Conclusions **12**

Report References **12**

Code References **12**

Appendices **12**

. . . . 12
. . . . 13
. . . . 14

Abstract

This paper provides an introduction on the background of this data set and indicates the goals of our modelling efforts in relation to the stated problem. Following this section, we provide a methodology of several various models after initial data set and variable consideration with the appropriate changes being made. We present the corresponding test error rates using the validation set approach and K-fold cross validation of these models along with other model analyzing metrics and provide a suggestion on the model which provides the best predictive ability for the *known* keystroke dynamic data set. Following this suggestion, the *unknown* keystroke dynamic data set was used to bolster the findings of the suggested model along with a summary of conclusions of our findings.

Introduction

The use of keystroke dynamics collects information on the typing patterns of a person(s) in a detailed timing manner, which can and has been used to develop additional security measures for online and/or passcode sites (i.e. banks, etc.) [1] and/or identify different individual(s). In this current paper, two data sets were provided with keystroke dynamic information in which subjects entered the same passcode, ".tie5Roan1" to gain access to a protected system. One data set contained known subjects who entered keystroke information and the other with unknown subjects entering the same information (hereafter referred to as *known* and *unknown*, respectively). The both data sets listed data in a hierarchal format to identify each time the passcode was entered (i.e., an individual observation) with the *known* data set consisting of the known subject (*subject*), a designated session (*sessionIndex*), and the number of times (i.e., replicates) each subject entered the passcode for a given session (*rep*). The *unknown* data set contained designated sessions (*sessionID*) along with number of times the passcode was entered for each session (*rep*) but no information on who entered it. Both data sets contained an additional 31 variables consisting of the keystroke dynamic information for the listed passcode; a detailed description for each of these variables is listed in Appendi .

The overall goals of this project and resulting paper were to first develop several different models (i.e., classifiers) following exploratory analysis and variable selection using the *known* data set in order to find a model with the best predictive ability for determining subjects from unknown keystroke dynamics information. Our second goal was to analyzing performance metrics of these developed models to determine the model with the best predictive ability (i.e., lowest test error rate) from the *known* data set. Our final goal included the use of the *unknown* data set within the final selected model to try to improve the model's accuracy. The steps used to address these goals are listed in the following sections.

Methods and Results

Exploratory Data Analysis and Variable Selection

Prior to developing any models, exploratory graphs were developed in order to assist in variable selection and model development. Also, to make plotting and visual analysis easier, the original names/values of the `subject` variable are replaced, this could be seen in Appendix . The first set of graphs developed consisted of a several series of ridge plots depicting the distributions of time (milliseconds) for each subject: individual character hold in the passcode (Hold), key transition for down-down (DD), and key transition for up-down (UD). Since this plotting looked at each individual, a large amount of plots were developed; therefore, only several sample ones (Figure 1) are presented here, while the remaining plots are listed in Appendix 2. While these plots were informative, the overall number of subjects and variables made these plots quite cumbersome to analyze; therefore, an additional heat map was created depicting the same information but allowing us to look at each subject across all three differing keystroke information sets as listed above (Figure 2). Along with exploring the *known* data set at the subject level, ridge, violin, and box plots were made depicting the total distributions of all combined subjects along with outliers with the same keystroke information sets (Figure 2, 3, 4).

```
plotHeatMap(known.Long.PerSubject)
```

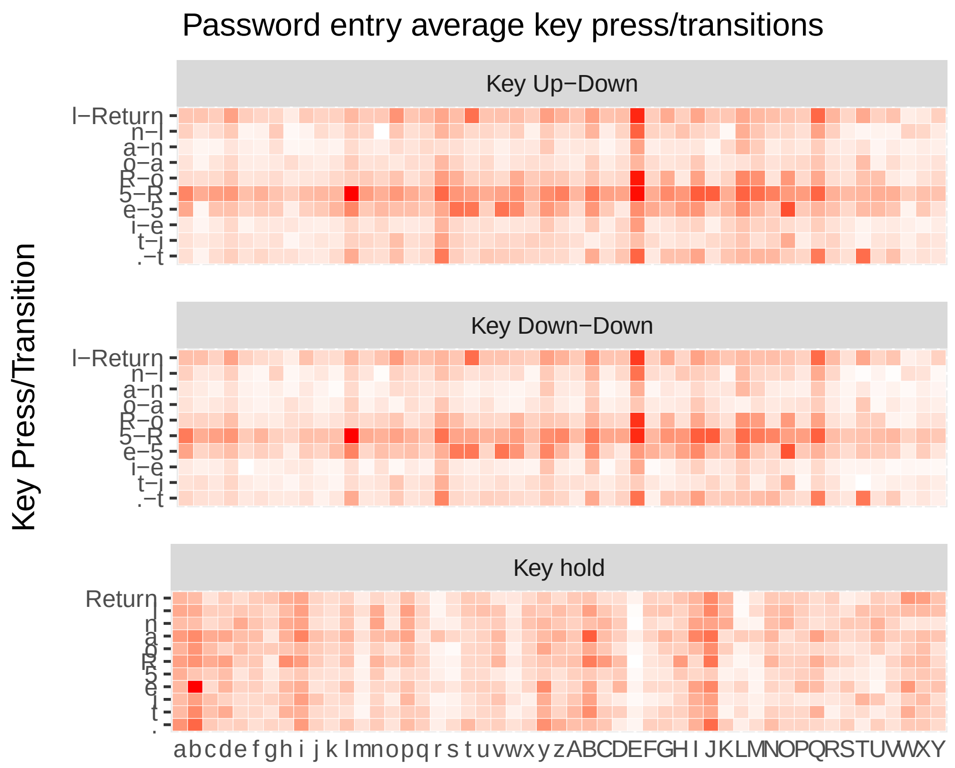

Figure 1: Heat map of every subject for three different sets of keystroke data sets (i.e., Key Up-Down, Key Down-Down, and Key hold).

```
plotKeyActionSummary(known.Long.PerSubject %>% filter(Key.Action == KEY_ACTION_HOLD), "Key Action - Hol
```

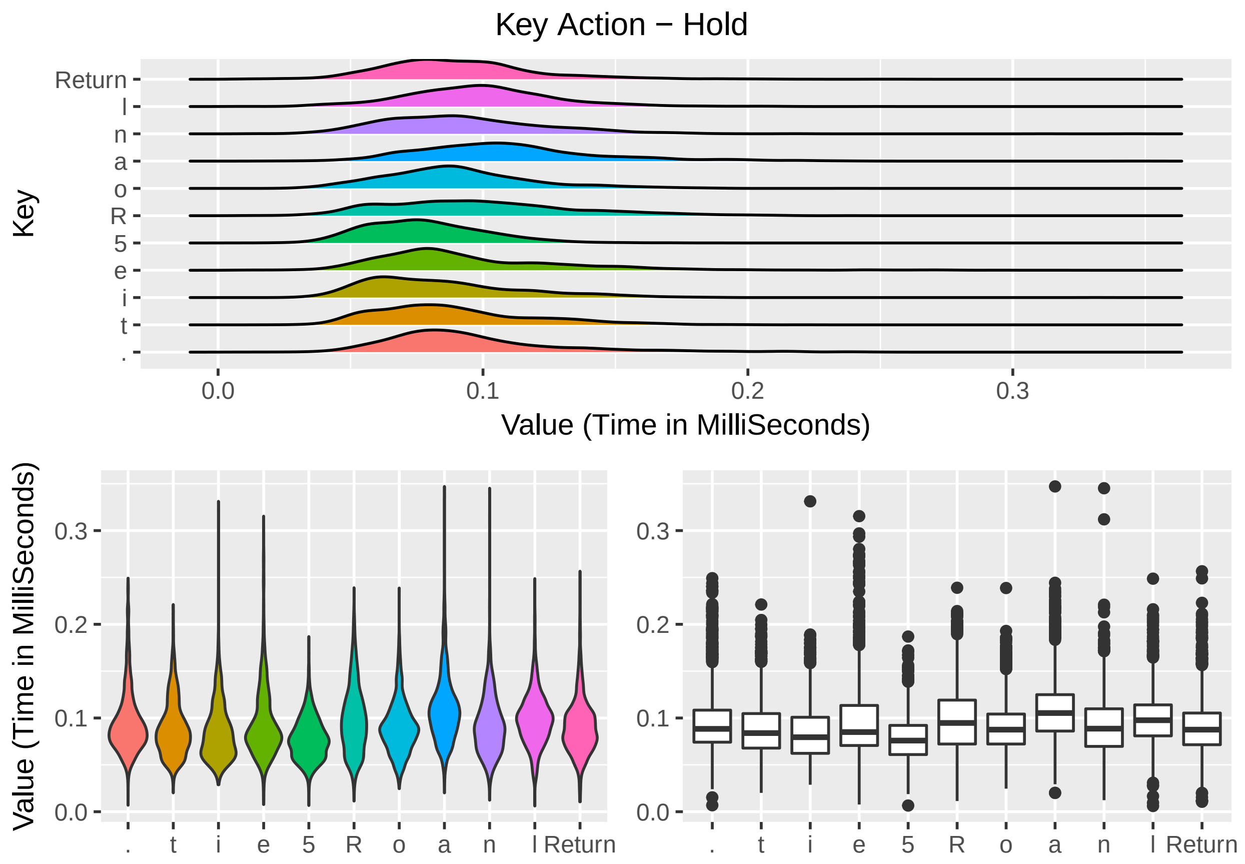

Figure 2: Ridgle plot, violin plot, and barplot of depicting overall distribution of Up-Down keystroke information.

```
plotKeyActionSummary(known.Long.PerSubject %>% filter(Key.Action == KEY_ACTION_DOWN_DOWN), "Key Action -
```

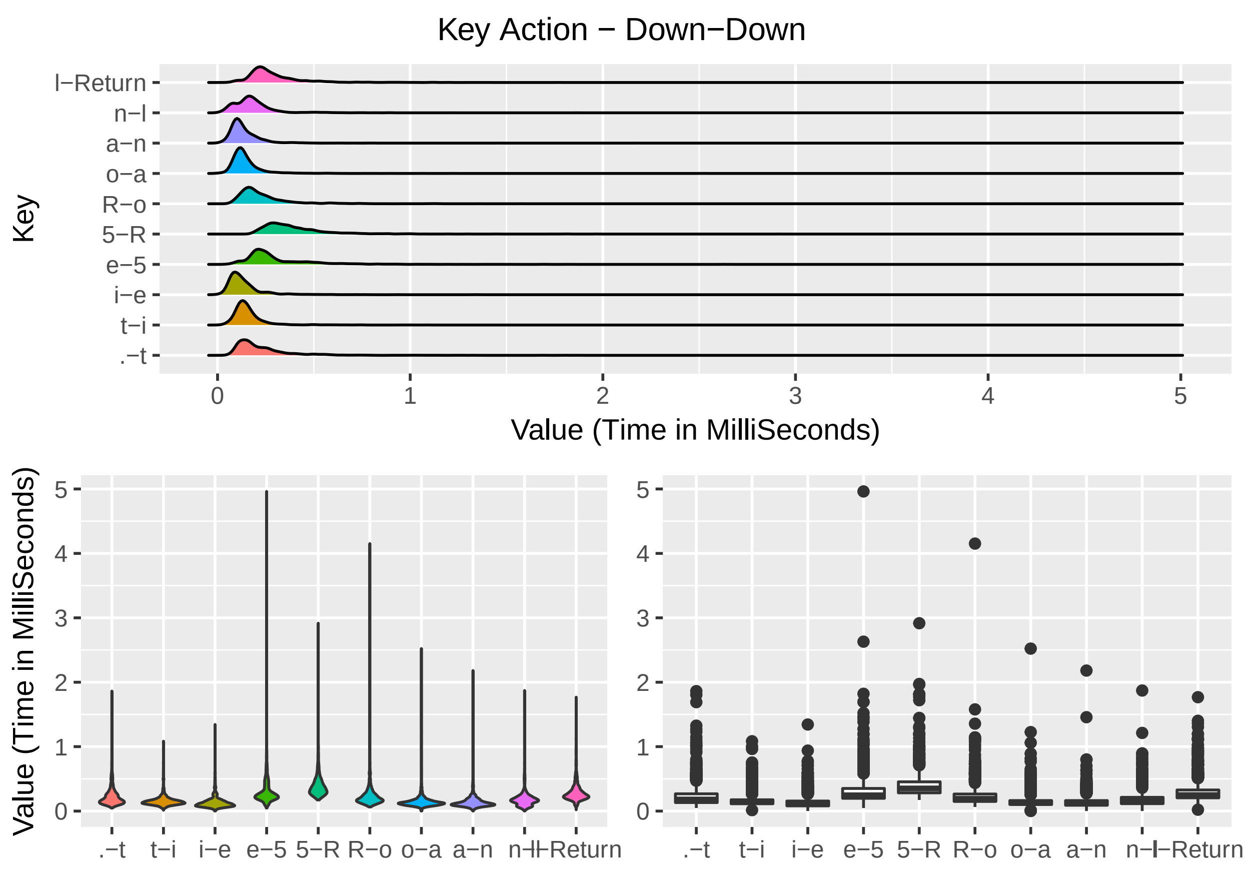

Figure 3: Ridgle plot, violin plot, and barplot of depicting overall distribution of Down-Down keystroke information.

```
plotKeyActionSummary(known.Long.PerSubject %>% filter(Key.Action == KEY_ACTION_UP_DOWN), "Key Action - U
```

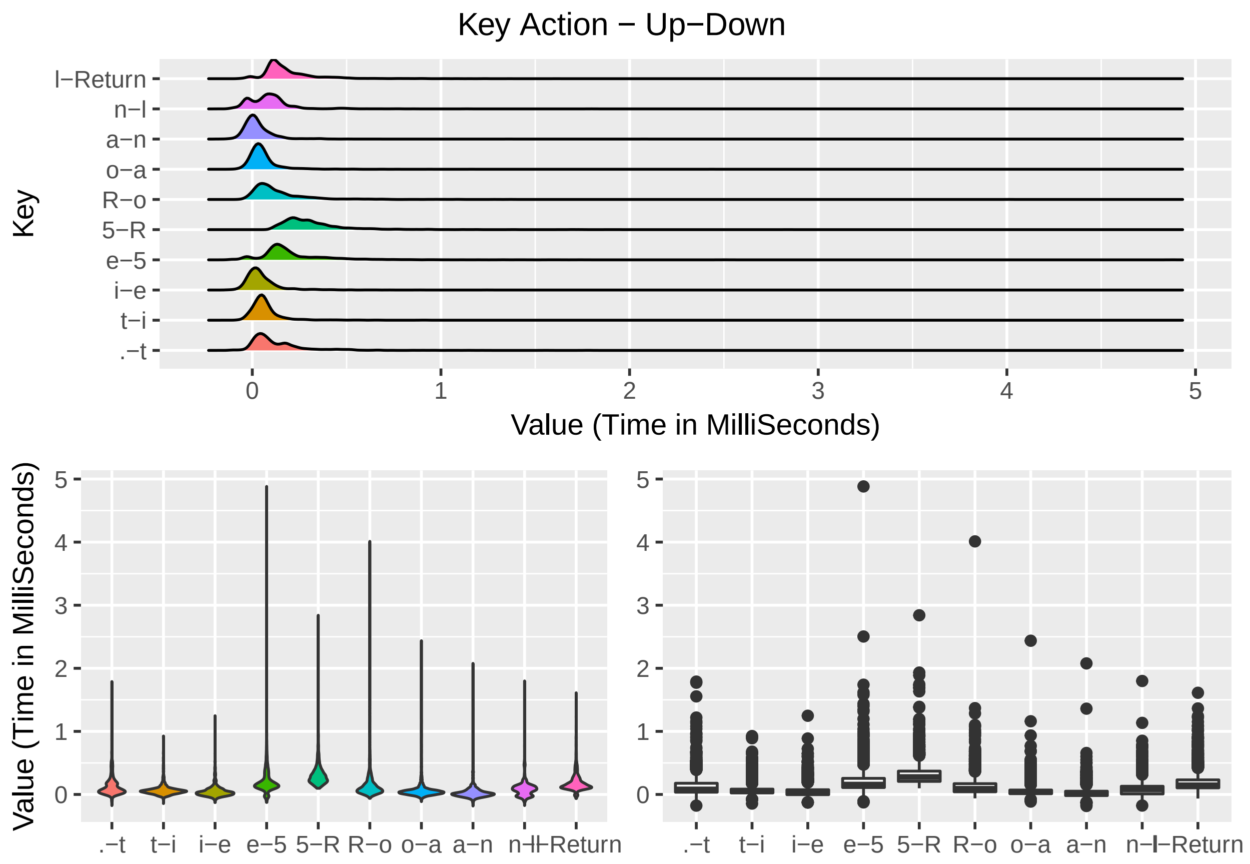

Figure 4: Ridgle plot, violin plot, and barplot of depicting overall distribution of Up-Down keystroke information.

Analysis of the ridge plots indicated a large amount of similar distributions occurring over a wide range of subjects and keystroke information. Likewise, the heat maps depicted differences occurring between different subjects within each keystroke set. However, this graph also pointed to the possibility of collinearity occurring do to the similar trends seen between the key transition for DD and UD, which is addressed later in this section. For the overall distribution plots, very similar patterns were seen in the UD and DD information, while the Hold information contained much wider distributions. Also, these same plots, especially the boxplots, indicated that the data set maintains a high amount of potential outliers. While several different approaches could be taken to address this issue such as removing observations for justifiable reasons, due to the limited amount of background knowledge on this data set, we felt it would be unwise to remove such observations without knowing the true nature behind their variation. Therefore, we decided to use a conservative approach by including all the data points within our models, knowing reduced model predictability may occur, in order to maintain a high level of statistical integrity. Such data points could be reviewed and removed at a later date after more background information is garnered on those observations in question.

As mentioned before, several of the plots suggested the possibility of collinearity between variables; therefore, both scatterplot and correlation matrices were created to test if this occurred. These matrices indicated collinearity occurred between the keystroke information between UD and DD keystroke times for all typed components of the passcode with a pearson correlation coefficient greater than 0.9 and clearly depicted linear relationship seen in Figure 5 (due to the overall size of the scatterplot matrix from the 51 subjects, only plots of the variables with collinearity are shown here).

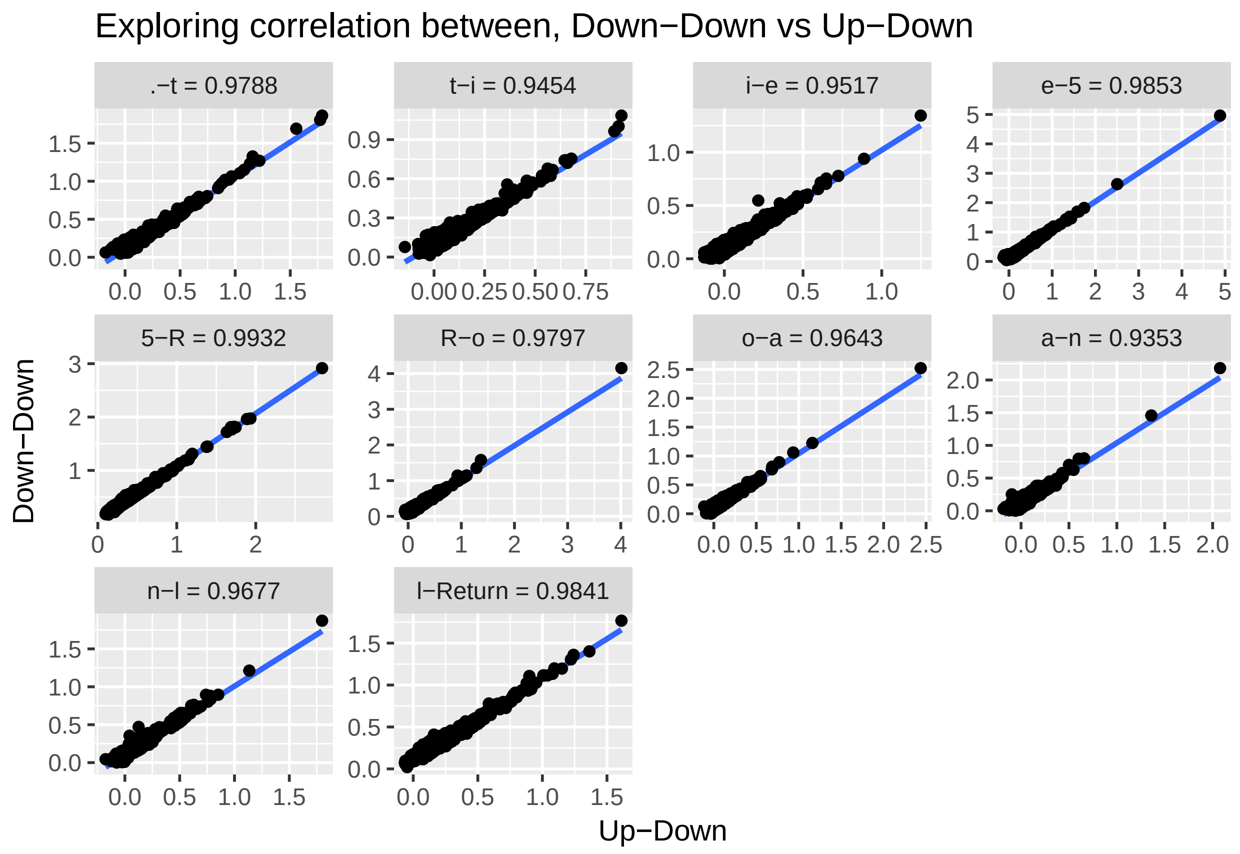

Figure 5: Scatterplot matrix between Down-Down and Up-Down keystroke infomation for all variables showing collinearity.

In order to reduce potential problem with model performance and overfitting from collinearity, we decided to remove either the UD or the DD variables. To help choose which variable to remove, we created two models with the first containing all variables except UD variables and the second with all variables except DD variables. A quadratic discriminant analysis (QDA) model was fit to determine which set of predictors made a better model based on calculated test errors. From the validation set approach (VSA), the test error was lower for the model that used the UD variable; therefore, it was decided that for all future models, all DD variables would be removed.

Another consideration taken into account was the *sessionIndex* variable in the *known* data set. This variable had only two different values; 7 or 8 with some subjects having a 7, some an 8, and others having both. Such differences led us to question if potential differences between the 7's and 8's for each covariate occurred. Because the distribution of the covariates for each of the groups is not known, 31 non-parametric tests were run to determine significance for each variable. Additionally, we decided to only use those observations that had both a 7 and an 8 which were analyzed with a Wilcoxon rank-sign test. The p-values for each of the variables can be found in the appendix. To adjust for multiple testing, we used the Bonferroni and Benjamini-Hochberg correction factors. The Bonferroni correction factor was extremely conservative, while the Benjamini-Hochberg P-value indicated a significant difference (i.e., rejected the null hypothesis) between means of the first 5 variables rejected the null hypothesis assuming an $\alpha = 0.05$. Therefore, we proceeded with caution, but choose use both the 7 and 8 together in the training data set. The *known* data set was split with a 70/30 ratio between training/test data sets; barplots of the training and test data sets were created and ensured that all subjects were represented in both data sets (Figure 6)

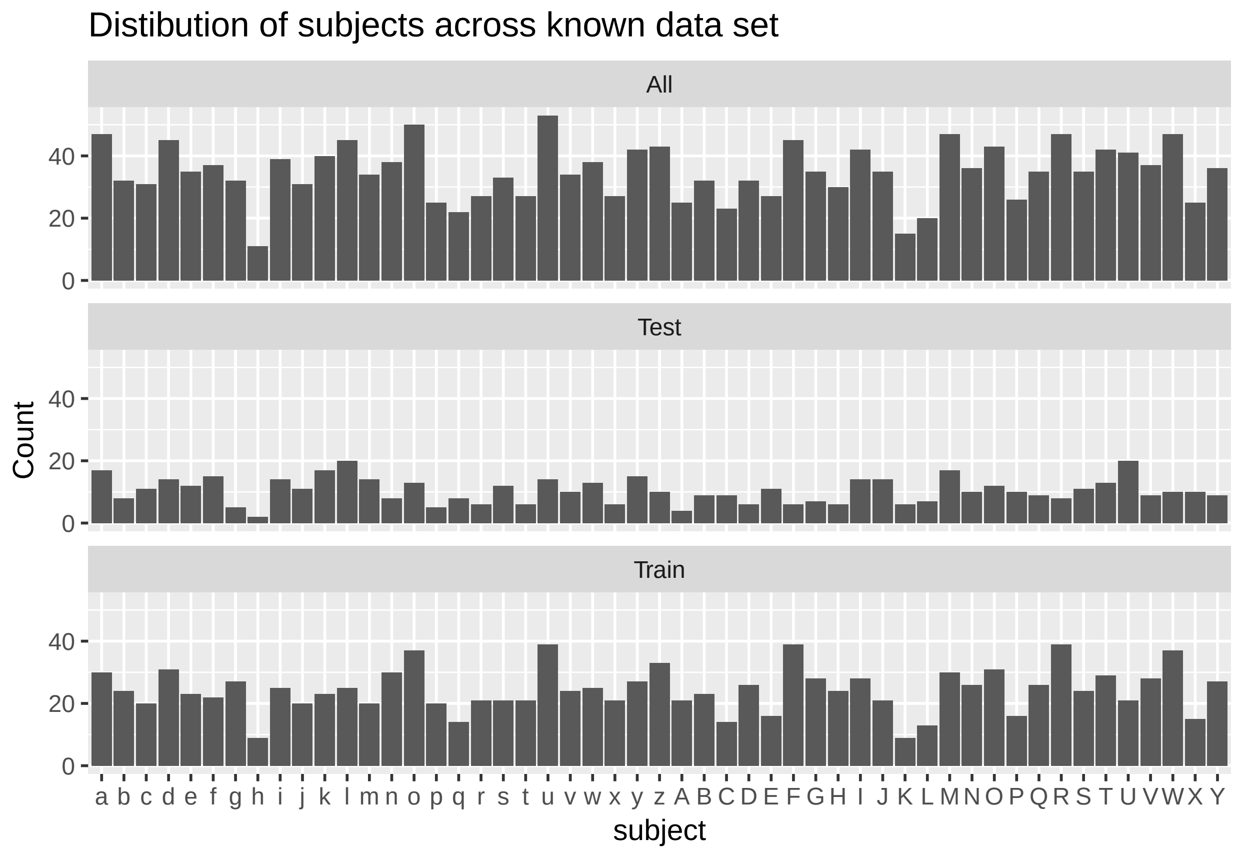

Figure 6: Barplots of number of subject observations included in the known, training, and test, datasets.

Model Methodology

Due to the nature of the categorical response variable, regular binary modeling techniques as they are structured in our book would be ill suited for modelling. However, we decided to use a one-vs-all approach in order to create a "quasi-binary" outcome for each of the 51 subjects for several of the models mentioned below. The process of this technique occurred as follows: 1) one class was chosen, 2) a new dependent variable is created with a value of 1 if the observation is in the class or 0 if otherwise and 3) a given model is fit using the original covariates on the created binary dependent variable [2]. This process is performed iteratively for all of the different subject values resulting in 51 different models each with a different set of estimated parameters that are estimated using the typical logistic regression maximum likelihood estimation.

Parametric Modelling

In order to implement this algorithm within a parametric model, we created a function that fits a logistic regression model which iterated through different cutoff points in order to determine the best cutoff for that particular logistic regression model. After running this function on the *known* data set, warnings occurred indicating the algorithm did not converge along with fitted probabilities of 0 or 1. Analysis of the model output found that many of the parameter estimates did not make sense including standard error estimates that were extremely large for many of the parameter estimates. These results suggested that the model created a situation where full separation or quasi-separation occurred, likely due to the one-vs-all approach. This situation resulted due to limited crossover between the covariates for the two response outcomes which caused the results to become unreliable. (A 2-d visualization of this is given below.)

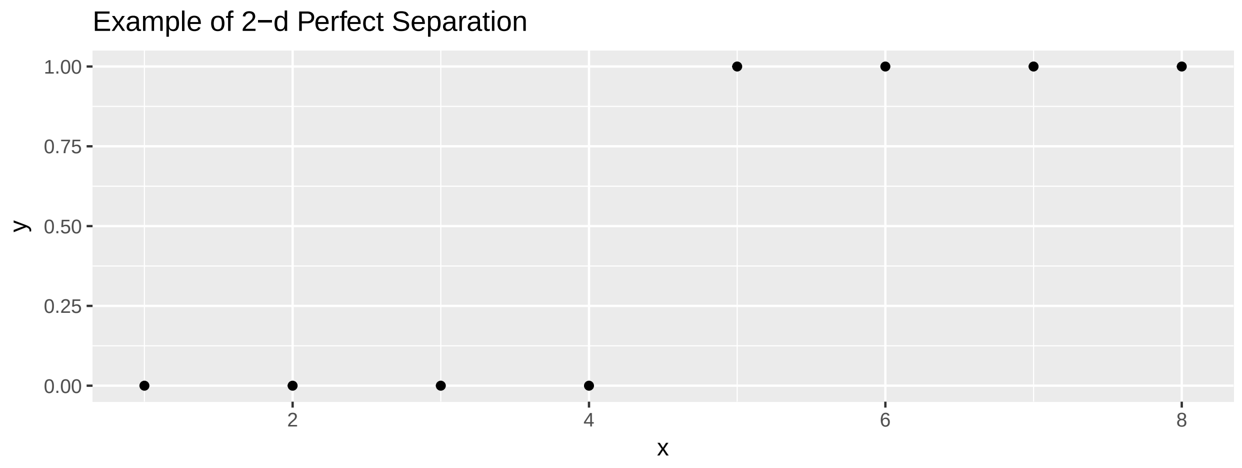

Figure 7: Example of 2-d separation

Two additional models considered in development using the known data set were a linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA) which were created, performed, and tested for accuracy using the VSA. Unlike, the logistic model mentioned above, this model produced reliable results which resulted in a VSA test error of 27.77%. Due to this relatively large error rate (given the current data set and goals) at hand along with unreliable results from the logistic regression model, it appeared that the usage of non-parametric techniques would be more suitable and are discussed in the next section.

Non- and Semi-Parametric

K-nearest neighbors

Another modeling method analyzed on the *known* data set was K-nearest neighbors (KNN) classification. We decided to used 10-fold cross validation to obtain 10 separate test error rates to determine a value for K. These test error rates were then averaged to obtain a single error rate for a given value of K, which were incremented through values of K from 1 to 100. After doing this, we get the following plot referenced in the Figure 8. This plot showed that a generally increasing trend in the error rate occurred as the value of K increased. The value of $K = 1$ resulted in the lowest error rate. This may be due to the correlation within each of the repetitions within each session. The error rate corresponding to a $K = 1$ was 23.29%, indicating poor model performance.

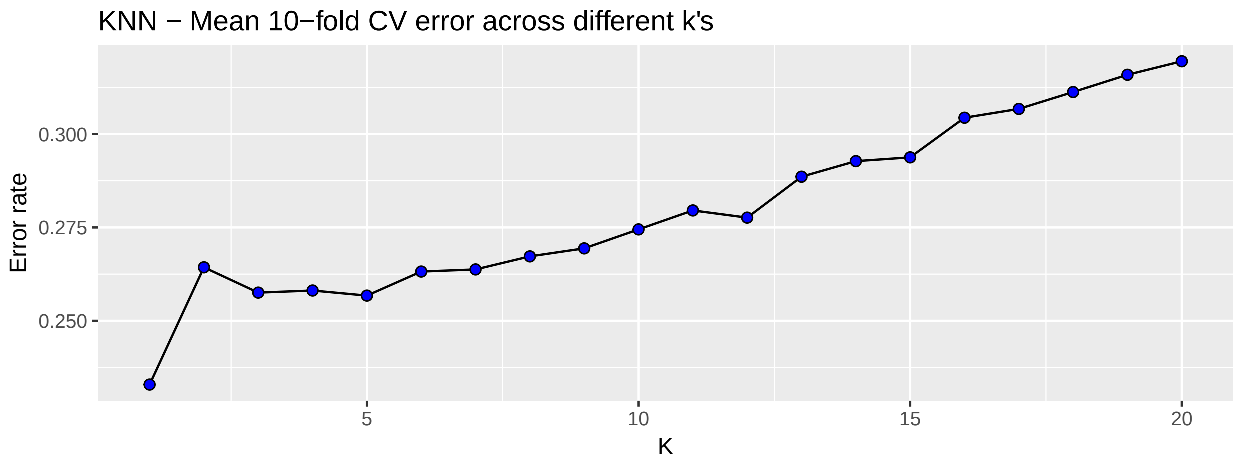

Figure 8: Mean 10-fold CV error rates for different K in KNN classifier.

Classification Tree

Using the full formular, a multi-class classification tree is created using the *rpart* library (*We tried tree, but it doesn't seem to be able to handle more that 32 classes*). Before fitting the tree, the *tune* function is used to tune the *cp*, *minbucket (min # of observation in terminal node)* and *maxdepth (max depth of tree)* hyperparameters, achieved through 10-fold cross validation. Using the optimal hyperparameters, a multi-class classification tree is created. Using 6-fold cross validation on all data in *known.csv*, an average test error of 35.75% is obtained. This is quite poor.

Random Forest

Given the poor performance of the *rpart* model, we'll explore some other forms of classification tree models. Specifically looking into Bagging, and RandomForest. Given the computation intensiveness of these algoriuthim, we decided to limit the space where we try to optimize our hyerparameters, in this case *ntree* and *mtry*. In Figure 9, we see the errors obtained from differtn combinatio of the hyperparameters. On each combination, an *Out-of-Bag (OOB)* and *test* error is obtained, using 6-fold cross validation, and VSA.

Although according to Leo Breiman, *...In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error...* Following this tredn of tought, it could be seen in 9, that for the errors obtained without CV, when compared to test error, the OOB is noticably bigger across the board. Hinting that using the OOB error as a form of model goodness, a better model could be achieved. With this, the mtry of 4 is used for fitting a random forest of 1000 trees.

Fiting this new model with the traiing dataset, a test error of 3.56% is obtained. This is quite an improvement when compared to

Using the full formular, a multi-class classification tree is created using the *rpart* library (*We tried tree, but it doesn't seem to be able to handle more that 32 classes*). Before fitting the tree, the *tune* function is used to tune the *cp*, *minbucket (min # of observation in terminal node)* and *maxdepth (max depth of tree)* hyperparameters, achieved through 10-fold cross validation. Using the optimal hyperparameters, a multi-class classiifcation tree is created. Using 6 fold cross validation on all data in *known.csv*, an average test error of 35.75% is obtained. This is quite poor.

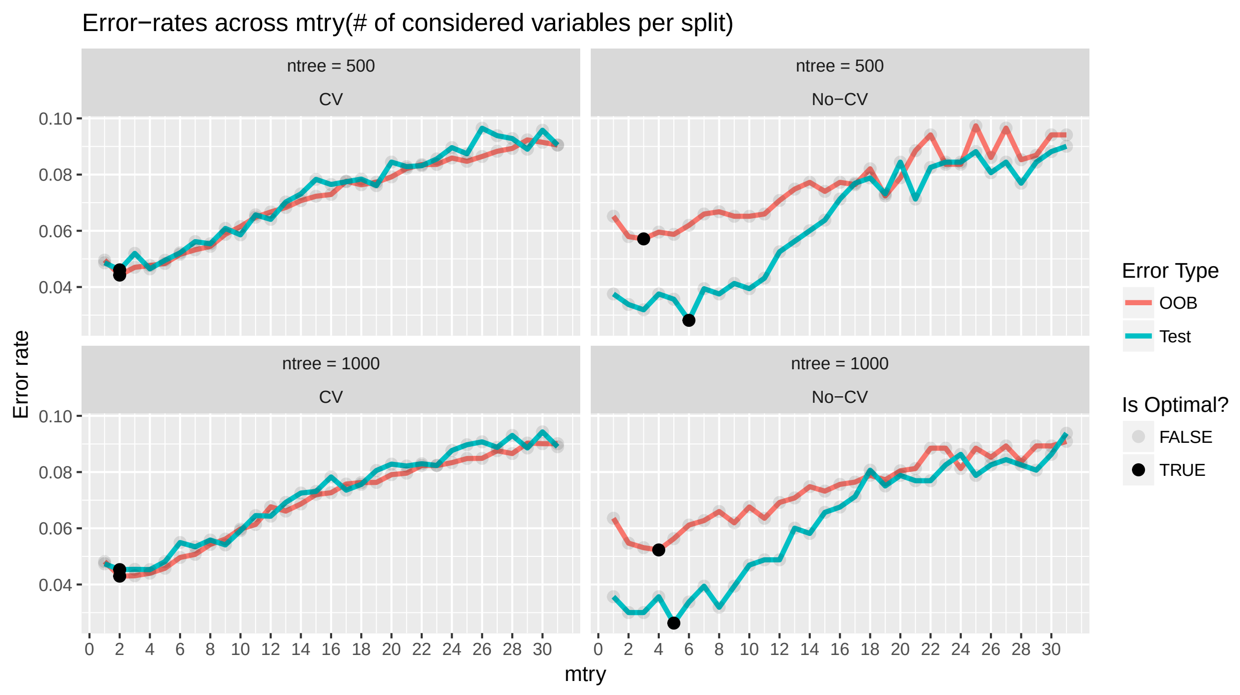

Figure 9: Error rates accross mtry values

Support Vector Machine

Random Forest

To fit a Support Vector Machine model, the *e1071* library is used. Unlike the One-vs-All approach we've used prior, this library uses One-vs-One to handle multiclass. For the SVM, we explore 3 kernel basis function, *linear*, *polynomial*, and *radial*. The tune function is used to obtain the optimal hyperparameters:

- linear - cost(c)
- polynomial - cost(c), degree
- radial - cost(c), gamma

Using the optimal hyperparameters, we fit 3 SVM models. Table 1 shows the performance of all 3 models, where the linear and radial kernels perform best.

Table 1: Error rates for all 3 SVM models, with various kernel function

linear	11.44%
polynomial	12.95%
radial	10.51%

Final Model with Unknown Data Set

(Add paragraph on why we selected random forest as best along with supporting evidence - table of test error rates from all models)

In an attempt to improve the predictive ability of our selected model, the *unknown* data set was analyzed and used as part of the model building process. To do this, initial attempts at clustering with both k-means and Mclust clustering algorithms, indicated that unclear cuts between groups were not formed. Therefore, a level of subjectivity would need to be incorporated in order to decide what cluster belonged to what subject. Instead of this procedure, we decided to infer predictions based on a random forest. A tuned random forest was first trained on the *known* data set then used to predict the *subjects* in the *unknown* data set. Each row associated with a unique *sessionID* was predicted, however it was not the same prediction for each row because of the differences in the covariates. The *subject* with the highest number of predictions within each *sessionID* was then assumed to represent the unkown subject for that *sessionID.* After the subjects were predicted and assigned in the *unknown* data set, this data was appended to the *known* training data set and validated on the *known* test data set. (Need to add results found from this procedure)

Conclusions

Report References

[1] Parimarjan, Negi. Adversarial machine learning against keystroke dynamics. PDF on http://www.Semanticsscholar.org. Accessed: April 23, 2018.

[2] Rifkin, Ryan. Multiclass classification. PDF on http://www.mit.edu/~9.520/spring09/Classes/ multiclass.pdf. Accessed: April 26, 2018.

[3] Breiman, Leo, and Adele Cutler. Random Forests. HTML on https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr. Accessed: April 26, 2018.

Code References

1) Source: https://tex.stackexchange.com/questions/2832/how-can-i-have-two-tables-side-by-side. Accessed April 26, 2018.
2) Source: https://stackoverflow.com/questions/38036680/align-multiple-tables-side-by-side. Accessed April 26, 2018.
3) Source: https://tex.stackexchange.com/questions/8652/what-does-t-and-ht-mean. Accessed April 26, 2018.
4) Source: https://stackoverflow.com/questions/45409750/get-rid-of-addlinespace-in-kable. Accessed April 26, 2018.
5) Source: https://www.r-bloggers.com/ggplot2-easy-way-to-mix-multiple-graphs-on-the-same-page/. Accessed April 27, 2018

Appendices

Table 2: Dictionary for the *known.csv*

Variable	Description
H.period	The amount of time that the "." is held down.
DD.period.t	The time between pressing down the "." key to the time to press down the "t" key.
UD.period.t	The time between the "." key coming up to the time to press down the "t" key.
H.t	The amount of time that the "t" is held down.
DD.t.i	The time between pressing down the "t" key to the time to press down the "i" key.
UD.t.i	The time between the "t" key coming up to the time to press down the "i" key.
H.i	The amount of time that the "i" is held down.
DD.i.e	The time between pressing down the "i" key to the time to press down the "e" key.
UD.i.e	The time between the "i" key coming up to the time to press down the "e" key.
H.e	The amount of time that the "e" is held down.
DD.e.five	The time between pressing down the "e" key to the time to press down the "5" key.
UD.e.five	The time between the "e" key coming up to the time to press down the "5" key.
H.five	The amount of time that the "5" is held down.
DD.five.Shift.r	The time between pressing down the "5" key to the time to press down the "shift+r" key combination.
UD.five.Shift.r	The time between the "5" key coming up to the time to press down the "shift+r" key combination.
H.Shift.r	The amount of time that the "shift+r" key combination is held down.
DD.Shift.r.o	The time between pressing down the "shift+r" key combination to the time to press down the "o" key.
UD.Shift.r.o	The time between the "shift+r" key combination coming up to the time to press down the "o" key.
H.o	The amount of time that the "o" is held down.
DD.o.a	The time between pressing down the "o" key to the time to press down the "a" key.
UD.o.a	The time between the "o" key coming up to the time to press down the "a" key.
H.a	The amount of time that the "a" is held down.
DD.a.n	The time between pressing down the "a" key to the time to press down the "n" key.
UD.a.n	The time between the "a" key coming up to the time to press down the "n" key.
H.n	The amount of time that the "n" is held down.
DD.n.l	The time between pressing down the "n" key to the time to press down the
UD.n.l	The time between the "n" key coming up to the time to press down the "l" key.
H.l	The amount of time that the "l" is held down.
DD.l.Return	The time between pressing down the "l" key to the time to press down the "return" key.
UD.l.Return	The time between the "l" key coming up to the time to press down the "return" key.
H.Return	The amount of time that the "return" is held down.
session	A session is a block of time where an individual has had access to the system over a continuous block of ti
rep	The individual passcode entries within a session are referred to as a repetition-within-session of the passco
subject	IDs for all 51 unique individuals.

Table 3: Mapping for the renamed *subject* variable

Old Subject	New Subject
s002	a
s003	b
s004	c
s005	d
s007	e
s008	f
s010	g
s011	h
s012	i
s013	j
s015	k
s016	l
s017	m
s018	n
s019	o
s020	p
s021	q
s022	r
s024	s
s025	t
s026	u
s027	v
s028	w
s029	x
s030	y
s031	z
s032	A
s033	B
s034	C
s035	D
s036	E
s037	F
s038	G
s039	H
s040	I
s041	J
s042	K
s043	L
s044	M
s046	N
s047	O
s048	P
s049	Q
s050	R
s051	S
s052	T
s053	U
s054	V
s055	W
s056	X
s057	Y

Table 4: Table of p-values for differences in SessionIndex for each variable

Variable	Test Statistic	P-Value	Bonferroni	Benjamini-Hochberg
UD.t.i	-4.7237521	0.0000012	0.0000359	0.0000359
DD.Shift.r.o	2.7433474	0.0030408	0.0942653	0.0471326
H.period	-2.7108818	0.0033552	0.1040120	0.0346707
DD.t.i	-2.7108818	0.0033552	0.1040120	0.0260030
H.l	-2.7108818	0.0033552	0.1040120	0.0208024
UD.Shift.r.o	2.2888283	0.0110447	0.3423846	0.0570641
H.five	-1.8018436	0.0357850	1.0000000	0.1584765
H.Shift.r	-1.8018436	0.0357850	1.0000000	0.1386669
DD.o.a	-1.8018436	0.0357850	1.0000000	0.1232595
UD.l.Return	-1.8018436	0.0357850	1.0000000	0.1109335
H.t	1.7693779	0.0384154	1.0000000	0.1082616
UD.period.t	1.1849962	0.1180095	1.0000000	0.3048578
DD.i.e	1.1849962	0.1180095	1.0000000	0.2814072
UD.five.Shift.r	1.1849962	0.1180095	1.0000000	0.2613067
DD.a.n	1.1849962	0.1180095	1.0000000	0.2438862
UD.a.n	1.1849962	0.1180095	1.0000000	0.2286433
H.i	-0.9577367	0.1690978	1.0000000	0.3083548
H.n	-0.9577367	0.1690978	1.0000000	0.2912240
UD.n.l	-0.9577367	0.1690978	1.0000000	0.2758964
H.Return	-0.9577367	0.1690978	1.0000000	0.2621016
DD.period.t	0.5356832	0.2960887	1.0000000	0.4370834
UD.i.e	0.5356832	0.2960887	1.0000000	0.4172160
DD.e.five	0.5356832	0.2960887	1.0000000	0.3990761

Variable	Test Statistic	P-Value	Bonferroni	Benjamini-Hochberg
UD.e.five	0.5356832	0.2960887	1.0000000	0.3824480
DD.five.Shift.r	0.5356832	0.2960887	1.0000000	0.3671500
H.a	0.5356832	0.2960887	1.0000000	0.3530289
DD.n.l	0.5356832	0.2960887	1.0000000	0.3399537
H.e	-0.1785611	0.4291412	1.0000000	0.4751206
H.o	-0.1785611	0.4291412	1.0000000	0.4587371
UD.o.a	-0.1785611	0.4291412	1.0000000	0.4434459
DD.l.Return	-0.1785611	0.4291412	1.0000000	0.4291412