Project 4

Objective

For this project, I'm utilizing a dataset used in a previous study by Carnegie Melon University where different machine learning models were tested to try to find a way to discriminate between users via their typing rhythms. The main goal of that study was to find a machine learning model with low enough error rates to obtain European Union approval so that it could be used legally and with partnership from European governments. The fields in the dataset include 51 subjects, 8 sessions per subject, 50 repetitions of typing the password: " $.tie5Roanl$", and 31 fields for actions required to type the password. Also, this dataset simulates typing the same password multiple times(50 in this case) in a given day, represented by repetition number, and 8 days in a row, represented by session number. (Killourhy & Maxion, 2009)

I'm starting with a null hypothesis that a person's typing dynamics change over time, short term and long term, which I'm going to attempt to reject. To do this, I will analyze the dataset via a mixed model approach. The response variable I am interested in here is total time to enter the password, while I am interested in the effects of the variables, rep which is the times the password was repeatedly typed in a day, session which is the days of the continuous repetitions of the password, and subject, which is the person typing the password. For this project, rep and session are fixed effects and subject is the nested random effect I will be testing, since rep and session are consistent actions but subject varies depending on who was available to be tested at the time. Assumptions with this dataset are that it has been cleaned and verified of any typos and entry errors.

```r
#Suppress warnings
options(warn=-1)
# function to install packages if they don't exist
usePackage <- function(p) {
    if (!is.element(p, installed.packages()[,1]))
        install.packages(p, dep = TRUE)
    require(p, character.only = TRUE)
}
# load packages used in this project
usePackage("readxl")
```

```
## Loading required package: readxl
```

```r
usePackage("ggplot2")
```

```
## Loading required package: ggplot2
```

```r
usePackage("knitr")
```

```
## Loading required package: knitr
```

```
usePackage("formattable")
```

```
## Loading required package: formattable
```

```
usePackage("dplyr")
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
usePackage("tidyr")
```

```
## Loading required package: tidyr
```

```
usePackage("lme4")
```

```
## Loading required package: lme4
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
usePackage("gee")
```

```
## Loading required package: gee
```

```
usePackage("Matrix")
usePackage("multcomp")
```

```
## Loading required package: multcomp
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: survival
```

```
## Loading required package: TH.data
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## The following object is masked from 'package:formattable':
##
##     area
```

```
##
## Attaching package: 'TH.data'
```

```
## The following object is masked from 'package:MASS':
##
##     geyser
```

```
usePackage("car")
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
## The following object is masked from 'package:dplyr':
## 
##     recode
```

```
usePackage("MASS")
usePackage("aod")
```

```
## Loading required package: aod
```

```
## 
## Attaching package: 'aod'
```

```
## The following object is masked from 'package:survival':
## 
##     rats
```

```
usePackage("DT")
```

```
## Loading required package: DT
```

```
usePackage("broom")
```

```
## Loading required package: broom
```

```
#Before being able to access the file, I had to open it in Excel and save it without making any changes. Not really sure why but doing this allowed me to open the file in R to add its contents to a dataframe
PasswordData.df <- as.data.frame(read_excel("DSL-StrongPasswordData.xls", sheet="Sheet1"))


#Here took a peak at the dataset, viewing the top 6 rows and a summary of the data

head(PasswordData.df)

summary(PasswordData.df)
```

Exploratory Analysis

```
#Suppress warnings
options(warn=-1)
#First, in order to explore the dataset, I'm going to create a few new datasets with
the original dataset so that I can create visualizations of the data.



#This dataset, PasswordDataEA1.df, sums all of the password input actions to create a
TotalTime column. Also, all of the password input action columns are removed.
PasswordDataEA1.3.df <- PasswordData.df
PasswordDataEA1.3.df$TotalTime <- rowSums(PasswordData.df[ , 4:ncol(PasswordData.d
f)])
PasswordDataEA1.df <- PasswordDataEA1.3.df[,c("subject","sessionIndex","rep","TotalTi
me")]


#This dataset, PasswordDataEA2.df, groups the original dataset by subject and session
Index and obtains the mean of each password input action under each subject/session g
roup.
PasswordDataEA2.df <- PasswordData.df

PasswordDataEA2.df <- PasswordDataEA2.df %>%
  group_by(subject, sessionIndex) %>%
  summarise(across(H.period:H.Return, mean, .names = "{col}_mean"))
```

```
## `summarise()` has grouped output by 'subject'. You can override using the `.groups
` argument.
```

```
#This dataset, PasswordData_long.df, is just a long form version of the original data
set with just 5 columns: subject, sessionIndex, rep, action, time_spent
PasswordData_temp.df <- PasswordData.df
PasswordDataCols.df <- colnames(PasswordData.df)
PasswordData_long.df <-  PasswordData_temp.df %>%                                   #
Apply pivot_longer function
  pivot_longer(PasswordDataCols.df[4:34], names_to = "action", values_to = "time_spen
t")


#This dataset, PasswordDataEA1.1.df, takes the dataset I created earlier, PasswordDat
aEA1.df, which obtained the total times per repetition, groups it by subject and sess
ionIndex, and obtains the mean of total times per each subject/session group
PasswordDataEA1.1.df <- PasswordDataEA1.df %>%
  group_by(subject, sessionIndex) %>%
  summarise(Mean_TotalTime = mean(TotalTime))
```

```
## `summarise()` has grouped output by 'subject'. You can override using the `.groups
` argument.
```

```
#This dataset, PasswordDataEA1.2.df, takes PasswordDataEA1.1.df and obtains the difference in average total time spent per repetition between the last session and the first session for each subject
PasswordDataEA1.2.df <- PasswordDataEA1.1.df %>%
  group_by(subject) %>%
  mutate(DiffBtwnLastFirst = Mean_TotalTime[sessionIndex == 1] - Mean_TotalTime[sessionIndex == 8])

PasswordDataEA1.2.df <- unique(PasswordDataEA1.2.df[,c(1,4)])



#This dataset, PasswordDataEA1.3.df, includes subject, sessionIndex, the difference between the first and last repetitions of each session
PasswordDataEA1.3.df <- PasswordDataEA1.df %>%
  group_by(subject,sessionIndex) %>%
  mutate(DiffBtwnLastFirst = TotalTime[rep == 1] - TotalTime[rep == 50])

PasswordDataEA1.3.df <- unique(PasswordDataEA1.3.df[,c(1,2,5)])

PasswordDataEA1.3.df <- PasswordDataEA1.3.df %>%
  group_by(subject) %>%
    summarise(DiffBtwnLastFirst = mean(DiffBtwnLastFirst))

#Datasets created

 # PasswordData.df
 #
 # PasswordDataEA1.df
 #
 # PasswordDataEA1.1.df
 #
 # PasswordDataEA1.2.df
 #
 # PasswordDataEA2.df
 #
 # PasswordDataEA1.3.df
 #
 # PasswordData_long.df




#Visualizations
#This box plot shows the difference in total time spent to type the password between the first and last repetition per session to show what improvement is seen within each session by each subject
boxplot(PasswordDataEA1.2.df$DiffBtwnLastFirst, density = 20,
        legend.text = rownames(PasswordDataEA1.2.df$subject), horizontal = TRUE, xlab = "Difference in Total Time Spent per Repetition (seconds)", ylab = "Subjects")
title(main = list("Average Improvement in Time Spent Between First and Last Sessions (repetitions per session were averaged)", font = 4))
```

```
#This box plot shows the average difference in total time spent to type the password between the first and last session of each subject
boxplot(PasswordDataEA1.3.df$DiffBtwnLastFirst, density = 20,
        legend.text = rownames(PasswordDataEA1.3.df), horizontal = TRUE, xlab = "Difference in Total Time Spent (seconds)", ylab = "Subjects")
title(main = list("Average Improvement in Time Spent Between First and Last Repetitions per Session", font = 4))
```

```
#This histogram with overlayed density plot shows the distribution of data and whether the Response Variable makes up a normal distribution
ggplot(PasswordDataEA1.df, aes(x=TotalTime)) +
            geom_histogram(aes(y = ..density..), binwidth=.25, colour="black", fill="white") +
            stat_function(fun = dnorm, lwd = 2, col = 'red',
                          args = list(mean = mean(PasswordDataEA1.df$TotalTime), sd = sd(PasswordDataEA1.df$TotalTime)))  +
    labs(title = "Distribution of the response Variable (Total Time)")
```

My exploratory analysis includes box plots showing the average improvement of the subjects between their first and last sessions as well as first and last repetition per session. These plots show that there are outliers in the response variable of the dataset where some subjects committed actions during the typing of the password that were much slower or faster than they usually would, or in some cases, much slower or faster than other subjects. Furthermore, these plots show that on average, there is a change in speed, almost always an improvement, between consecutive sessions and between consecutive repetitions. In fact, between the first and last sessions of the subjects, there is an average improvement of almost 2 seconds. Finally, I created a histogram overlayed with a density plot to check whether the data is normally distributed, which it seems to not be as it’s too heavily right-skewed.

For this project, as the outliers can be attributed to the randomness of the subjects, since people often make mistakes when typing ambiguous passwords, and the data is assumed to have been cleaned and verified, I have decided not to remove outliers.

Furthermore, I decided to use penalized quasi-likelihood along with random intercept and random intercept and slope models to model the data. This technique allows for the fitting of data to mixed effect models when the data is not of a normal distribution. It is an approximate inference technique that allows estimation of model parameters without knowledge of the error distribution of the response variable. (Everitt & Hothorn, 2014) More common methods such as restricted maximum likelihood and maximum likelihood require that the data be normally distributed. (Pilowsky, 2018) Penalized quasi-likelihood just requires that the response variable not have a mean less than five and the response variable does not fit a discrete count distribution such as Poisson or Binomial, or that the response variable is not binary. (Pilowsky, 2018) The response variable’s mean is greater than 5 and is not binary so this technique should work.

Penalized Quasi-Likelihood:

$$\hat{g}_n \in \arg\max_{g\in\delta}[\hat{Q}_n(F(g)) - \lambda_n^2 J^2(g)]$$

Random intercept and random intercept and slope models are commonly used as opposed to generalized linear models for datasets with repeated measurements. (Everitt & Hothorn, 2014) These models assume that correlation amongst independent repeated measurements on the same unit arises from the shared unobserved variables and that time has a fixed effect. (Everitt & Hothorn, 2014) The difference between random intercept and random intercept and slope models are that random intercept models measure dissimilarity in intercepts while random intercept and slope models measure dissimilarity in intercepts and slopes. (Everitt & Hothorn, 2014)

Random Intercept Model:

$$y_{ij} = \beta_0 + \beta_1 t_j + u_i + \varepsilon_{ij}$$

y_ij = observation made t_j = time i = individual

Random Intercept and Slope Model:

$$y_{ij} = \beta_0 + \beta_1 t_j + u_i + v_i t_j + \varepsilon_{ij}$$

u_i = intercepts v_i = slopes

Sources: (Everitt & Hothorn, 2014) (Mammen & van de Gee, 1997)

Decide what probability distribution best fits the dataset via quantile comparison plots

```r
#Suppress warnings
options(warn=-1)
# normal
qqp(PasswordDataEA1.df$TotalTime, "norm", main= "Normal")
```

```r
# lognormal
qqp(PasswordDataEA1.df$TotalTime, "lnorm", main= "Log Normal")
```

```r
# gamma
gamma <- fitdistr(PasswordDataEA1.df$TotalTime, "gamma")
qqp(PasswordDataEA1.df$TotalTime, "gamma", shape = gamma$estimate[[1]], rate = gamma$estimate[[2]], main= "Gamma")
```

```r
# negative binomial
nbinom <- fitdistr(round(PasswordDataEA1.df$TotalTime,0), "Negative Binomial")
qqp(round(PasswordDataEA1.df$TotalTime,0), "nbinom", size = nbinom$estimate[[1]], mu = nbinom$estimate[[2]], main= "Negative Binomial")
```

```r
# Poisson
poisson <- fitdistr(round(PasswordDataEA1.df$TotalTime,0), "Poisson")
qqp(round(PasswordDataEA1.df$TotalTime,0), "pois", lambda = poisson$estimate, main= "Poisson")
```

Next I utilized quantile comparison plots to check what probability distribution best fits the response variable. I tested with Normal, Lognormal, Gamma, Negative Binomial and Poisson. I decided to fit the dataset to a Gamma distribution as although Poisson seemed like a slightly closer fit, it also would require a transformation of the response variable from continuous to discrete, or in other words rounding to whole numbers.

Statistical Analysis

Here I fitted the dataset to a gamma model via penalized quasi-likelihood. I tested fit with a random intercept model as well as a random intercept and slope model. (Arbor Custom Analytics, 2020) Also, these were tested with identity and inverse link functions. The GLMMPQL function does not output AIC, BIC, or Log Likelihood and thus I tested fit via residuals charts. These charts showed that the random intercept model with the identity link function and the random intercept model with the identity link function are tied or close to tied for having the closest fit to the dataset.

```
#Suppress warnings
options(warn=-1)
#Testing the models
#Random intercept models with a random effect term consisting of a slope and cluster term
#Gamma Distribution with Identity Link Function
PQL1 <- glmmPQL(TotalTime ~ sessionIndex + rep, ~1 | subject, family=Gamma(link=identity), data = PasswordDataEA1.df, verbose = FALSE)
summary(PQL1)
#Gamma Distribution with Inverse Link Function
PQL2 <- glmmPQL(TotalTime ~ sessionIndex + rep, ~1 | subject, family=Gamma(link=inverse), data = PasswordDataEA1.df, verbose = FALSE)
summary(PQL2)
#Random Intercept and Slope models with random slope term for rep
#Gamma Distribution with Identity Link Function
PQL3 <- glmmPQL(TotalTime ~ sessionIndex + rep,~1+rep|subject, family=Gamma(link=identity), data = PasswordDataEA1.df, verbose = FALSE)
summary(PQL3)
#Gamma Distribution with Inverse Link Function
PQL4 <- glmmPQL(TotalTime ~ sessionIndex + rep,~1+rep|subject, family=Gamma(link=inverse), data = PasswordDataEA1.df, verbose = FALSE)
summary(PQL4)$Value









#Plot the residuals to check which models best fit the data
plot(PQL1, main = "Random Intercept  - Gamma Dist with Identity Link Function")
```

```
plot(PQL2, main = "Random Intercept  - Gamma Dist with Inverse Link Function")
```

```
plot(PQL3, main = "Random Intercept and Slope  - Gamma Dist with Identity Link Function")
plot(PQL4, main = "Random Intercept and Slope  - Gamma Dist with Inverse Link Functio
n")
```

```
#Results Table of final model results with a highlight on the chosen models
fixed.effect <- c('TotalTime ~ sessionIndex + rep','TotalTime ~ sessionIndex + rep','
TotalTime ~ sessionIndex + rep','TotalTime ~ sessionIndex + rep')
random.effect <- c('1 | subject','1 | subject','1+rep|subject','1+rep|subject')
family <- c('Gamma(link=identity)','Gamma(link=inverse)','Gamma(link=identity)','Gamm
a(link=inverse)')
fixed.vars <- c('sessionIndex / rep', 'sessionIndex / rep', 'sessionIndex / rep', 'se
ssionIndex / rep')
coefficient.estimate <- c('-0.186639 / -0.007104', '0.02501722 / 0.00130749', '-0.186
821    / -0.011385', '0.02638535 / 0.00050069')
std.error <- c('0.00293549 / 0.00046170','0.000200294 / 0.000011767','0.00289161 / 0.
00171970','0.000222314 / 0.000086832')
t_value <- c('63.58006 /    -15.38636', '124.90233 / 111.11277','   -64.60783 / -6.62
033','118.68508 / 5.76625')
p_value <- c('0 / 0', '0 / 0', '0 / 0', '0 / 0')
# Join the variables to create a data frame
df2 <- data.frame(fixed.effect,random.effect,family,fixed.vars,coefficient.estimate,s
td.error,t_value,p_value)

#Utilize datatable and formattable to highlight a row
datatable(df2) %>% formatStyle(
  'family',
  target = 'row',
  backgroundColor = styleEqual(c("Gamma(link=identity)"), c('lime'))
)
```

The results of these two models show that repetitions and consecutive sessions do influence password typing speed. In fact, the p value obtained shows that the null hypothesis has not been disproven. Results show that total time taken to type the password decreases as repetitions increase and to a greater extent as session increases. In other words, people tend to type a password faster the more they type it in a given day and when repeatedly typing the password on consecutive days.

Post Hoc Analysis

For the post hoc analysis, I decided to further confirm whether the response variable is not normally distributed and compare the fixed effects to determine if there's an interaction between them. I tested this via a Shapiro-Wilkins test for normality grouped by subject as this test has a limit of 3500 for observations, and by manipulating the closest fitting model to have a fixed variable of session:repetition (session:repetition).

```
#Suppress warnings
options(warn=-1)

# Run a Shapiro-Wilk test for normality on the dataset grouped by subject
ShapiroTestResults <- PasswordDataEA1.df %>%
  group_by(subject) %>%
  do(tidy(shapiro.test(.$TotalTime)))



#View the results in a boxplot to get an idea of where the data is centered at
boxplot(ShapiroTestResults$p.value, density = 20,
        legend.text = rownames(ShapiroTestResults$subject), horizontal = TRUE, xlab =
"P-Values", ylab = "Subjects")
title(main = list("Shapiro Test for Normality by Subject", font = 4))
```

```
#Check for an interaction term between sessionIndex and rep
posthocPQL1 <- glmmPQL(TotalTime ~ sessionIndex*rep + sessionIndex +rep, ~1 | subjec
t, family=Gamma(link=identity), data = PasswordDataEA1.df, verbose = FALSE)

#View Results
summary(posthocPQL1)

#Set up a formatted table for the results
fixed.effect <- c('TotalTime ~ sessionIndex*rep + sessionIndex + rep','TotalTime ~ se
ssionIndex*rep + sessionIndex + rep','TotalTime ~ sessionIndex*rep + sessionIndex + r
ep')
random.effect <- c('1 | subject','1 | subject','1 | subject')
family <- c('Gamma(link=identity)','Gamma(link=identity)','Gamma(link=identity)')
fixed.vars <- c('sessionIndex','rep','sessionIndex:rep')
value <- c('-0.268579', '-0.022724', '0.003146')
std.error <- c('0.00602308', '0.00109991', '0.00020085')
t_value <- c('-44.59160','-20.66014','15.66136')
p_value <- c('0','0','0')
# Join the variables to create a data frame
df2 <- data.frame(fixed.effect,random.effect,family,fixed.vars,value,std.error,t_valu
e,p_value)
#Highlight relevant row
datatable(df2) %>% formatStyle(
  'fixed.vars',
  target = 'row',
  backgroundColor = styleEqual(c("sessionIndex:rep"), c('lime'))
)
```

The results of my post-hoc analysis were that the data is definitely not of a normal distribution and that there is an interaction between session and repetition although it has a very small effect on the response variable compared to that of session and repetition individually.

Conclusions

The random effects model I created that fits the data the best indicates that the time taken to type a password, changes over time as one retypes it throughout the day and on consecutive days. Thus, I've failed to reject the null hypothesis and I have confirmed that at least within the boundaries of this sample and to a statistically significant extent, a person's typing dynamics change over time, short and long term. Also, the results of the post-hoc analysis showed that the data is extremely likely to not be normally distributed and that there is an interaction between session and repetition.