# Keystroke Dynamics

Md Mominul Islam

12/13/2021

## Project Objective

In our project, we are interested in incorporating *Keystroke Dynamics* into our *Password Based Security System*.One concern about this type of system is whether or not a user is consistent over time in how they type a given passcode. For this project we were given access to a group of 51 individuals who have typed a passcode for a specific system.

The passcode used by this code is : $.tie5Roanl$

Dr. Roy Maxian and colleagues recruited 51 subjects at CMU. Subjects completed 8 data collection sessions, with each session composed of 50 times of typing the above passcode. Each session was one day apart. We are asked to explore the conjecture that a person's typing dynamics changes over time using methods taught in STAT 601 Course.

## Keystroke Dynamics

The study of whether people can be recognized by their typing patterns, similar to how handwriting may be used to identify the author of a written piece, is known as keystroke dynamics. Acting as an electronic fingerprint or as part of an access-control system are two examples of possible uses. *Keystroke dynamics* concerns a user's typing pattern. When a person types, the latencies between successive keystrokes and their duration reflect the unique typing behavior of a person. The next time this user logs in, by comparing his current typing pattern with his previous typing patterns, companies can authenticate whether the logged in user is legitimate or fraudulent.

## Exploratory Data Analysis

```
library(readxl)
library(knitr)
library(reshape)
library(tidyverse)
library(dplyr)
library(ggplot2)
library(GGally)
library(car)
library(boot)
library(gridExtra)
library(MASS)
library(pROC)
library(Hmisc)
library("PerformanceAnalytics")
library(AppliedPredictiveModeling)
```

```
#loading data set from our work data
kd.main <-  read_excel("/Users/mominul/Desktop/Project 4/Work Data P4/workdat
a.xlsx")
cat("The Dimension of New Keystroke Dataframe is", dim(kd.main))

## The Dimension of New Keystroke Dataframe is 20400 34
```

Here, we have loaded our data set and showed the fisrt few rows as an output.

From the main paper, we can see that they have used some feature sets to train and test the detectors.

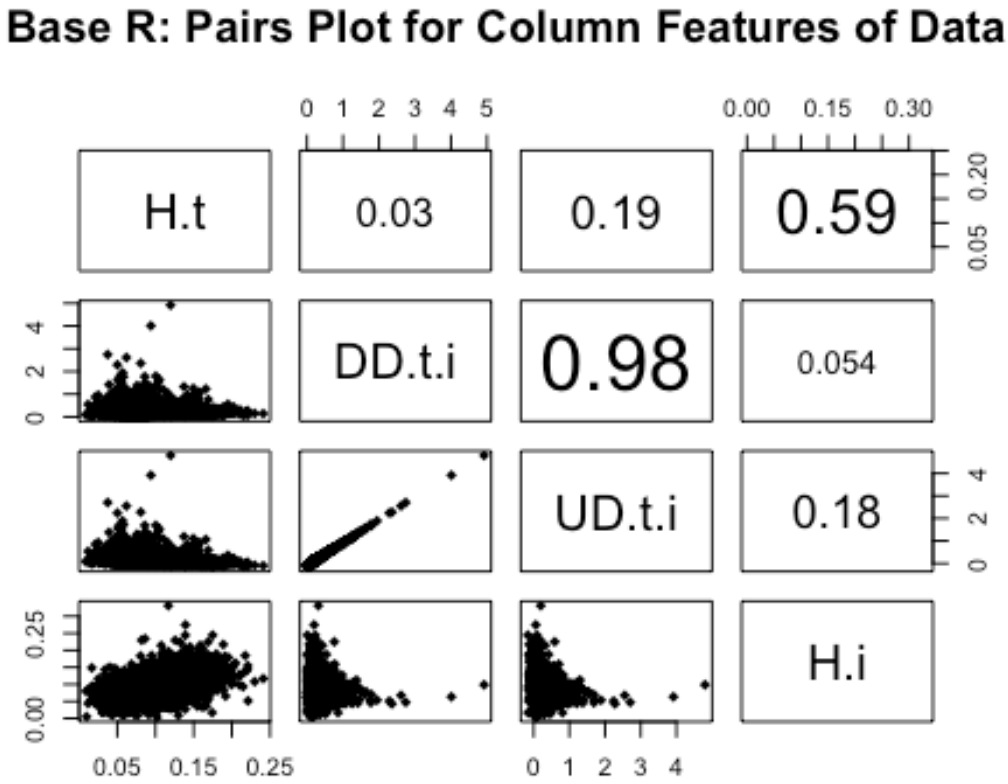
- Keydown-Keydown: time between the key presses of consecutive keys is used as a feature.
- Keyup-Keydown: time between the release of one key and the press of the next is used.
- Hold Time: Time between the press and release of each key is used

We have loaded the data set and sorted the data into three features which are also mentioned in the paper.

```
keystroke.main <- kd.main
# Getting total Time for every recitation
TT.Time <- rowSums(keystroke.main  [ , c(4:34)])
keystroke.main <- cbind(keystroke.main, TT.Time)

#selecting columns based on three features like DD time, UD time and hold tim
e
DD.Time <- keystroke.main[, c(1,2,3,5,8,11,14,17,20,23,26,29,32)]
UD.Time <- keystroke.main[,c(1,2,3,6,9,12,15,18,21,24,27,30,33)]
Hold.Time<- keystroke.main[,c(1,2,3,4,7,10,13,16,19,22,25,28,31,34)]

#Total DD, UD and Hold Time
DD.TT.Time <- rowSums(DD.Time[ , c(4:13)])
keystroke.main <- cbind(keystroke.main,DD.TT.Time)
UD.TT.Time <- rowSums(UD.Time[ , c(4:13)])
keystroke.main <- cbind(keystroke.main,UD.TT.Time)
Hold.TT.Time <- rowSums(Hold.Time[ , c(4:13)])
keystroke.main <- cbind(keystroke.main,Hold.TT.Time)
cat("The New Dimension of Dataframe is", dim(keystroke.main))

## The New Dimension of Dataframe is 20400 38
```

## Pairs Plot for Key Pressig Time Features

We have simple pairs plot to see correlation among the pressing time.




*Pairs Plot for Key Pressig Time Features*

*Correlation Matrix for Multiplex Data*

| | H.t | DD.t.i | UD.t.i | H.i |
|---|---|---|---|---|
| H.t | 1.0000000 | 0.0297135 | -0.1888858 | 0.5894718 |
| DD.t.i | 0.0297135 | 1.0000000 | 0.9759530 | -0.0536673 |
| UD.t.i | -0.1888858 | 0.9759530 | 1.0000000 | -0.1812750 |
| H.i | 0.5894718 | -0.0536673 | -0.1812750 | 1.0000000 |

All other boxes display a scatterplot of the relationship between each pairwise combination of variables.

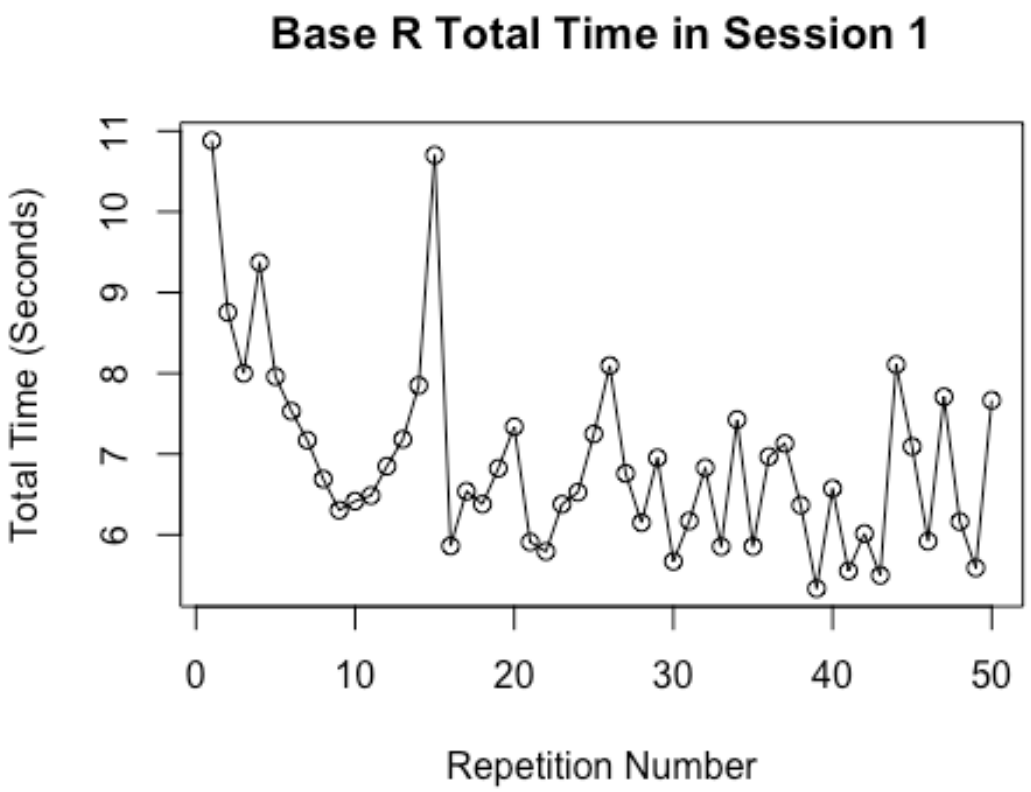
- The box in the left top corner of the matrix displays a scatterplot of values for H.t and DD.t.i features.
- From our correlation matrix, we can say that how the different pressing time are related to each other. Like the correlation between H.t and DD.t.i feature is 0.03.

Like the original paper, we looked into three different time slots;(Variable description found here: [Code Reference : https://github.com/rakshithca/KeyStroke-Dynamics/blob/master/Analysis.ipynb]). In their work they define *H_columns, DD_columns and UD_columns*. We have also done same work like them. Then, we dealt with individual coulmns and took their summation and named them *DD.TT.Time - For Total DD Time*, *UD.TT.Time- For Total UD Time* , *Hold.TT.Time - Total Hold Time* and lastly, *TT.Time -Total Time to Type the Password*. We added them with our main data frame named *keystroke.main*

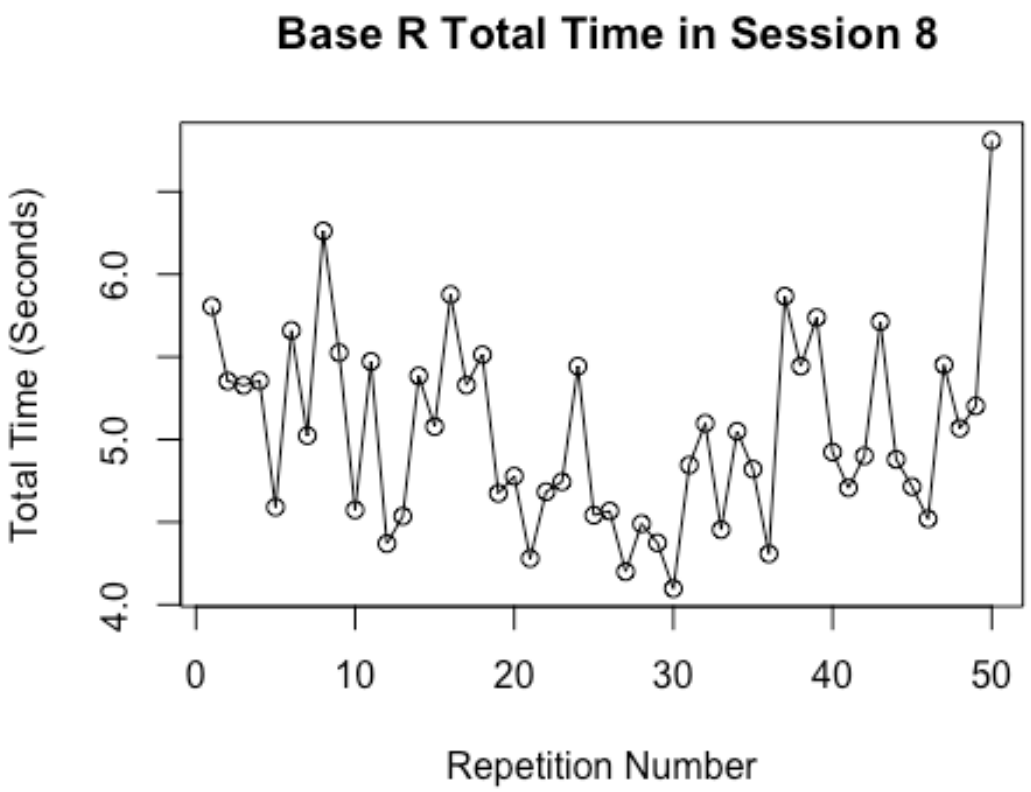
## Total Down-Down, Up-Down and Hold Time Analysis

### EDA with Total Time vs Repetitiom Number

```
#Selecting variables for total time analysis
EDA.TT <- keystroke.main[, c(1, 2, 3, 35)]
attach(EDA.TT)
#Plotting Total Time vs Repitation Number for Session 1 Subject5
sub2.session1 <- EDA.TT[which(subject=='s002' & sessionIndex == 1),]
plot(sub2.session1$rep,sub2.session1$TT.Time,
     type ='o',
     ylab = 'Total Time (Seconds)'  ,
     xlab = 'Repetition Number',
     main= 'Base R Total Time in Session 1')
```




```
sub2.session8 <- EDA.TT[which(subject=='s002' & sessionIndex == 8),]
plot(sub2.session8$rep,sub2.session8$TT.Time,
     type ='o', ylab = 'Total Time (Seconds)'  ,
     xlab = 'Repetition Number',
     main = ' Base R Total Time in Session 8')
```



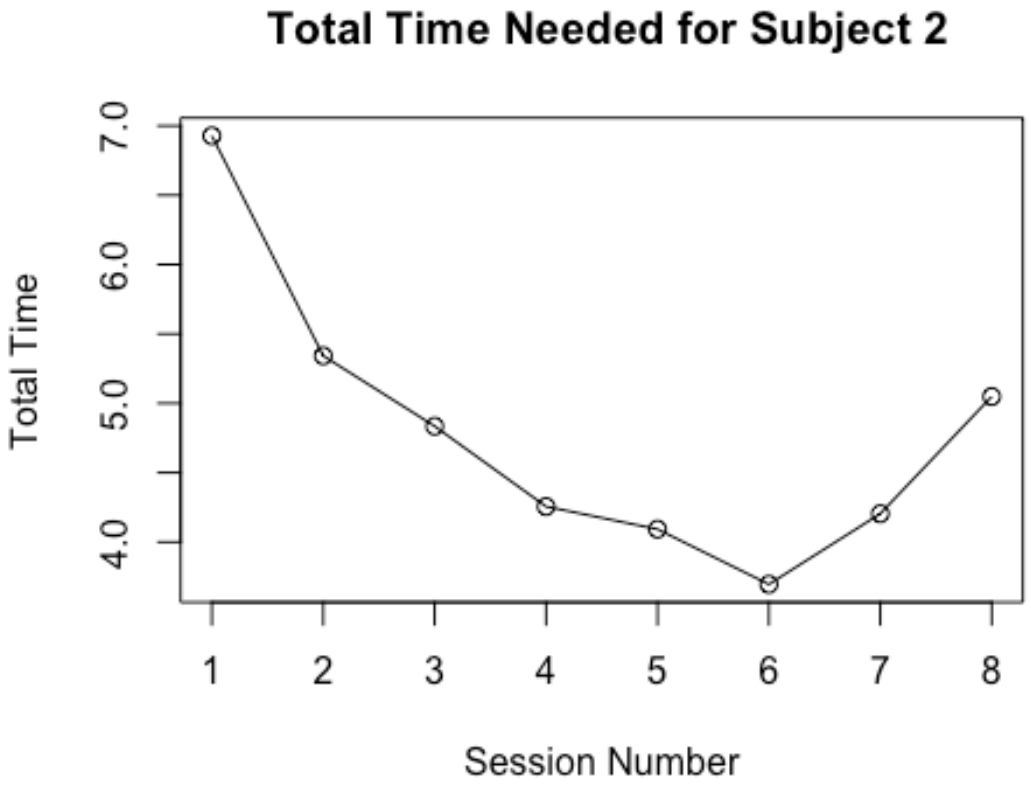

We have plotted our base R plotting using only one subject 02 and different session index like 1 and 8. With every session, total time is decreasing. We can see the proof of our analysis clearly from the graphs.

From the plot depicted above, we can see that the trend of the Total Time vs Repetition Number is decreasing, This means, with each repetition, the user get used to keystrokes of the password hence reducing the the total time. If you see closely, total time range for the first session was around 6~11 Seconds and at 8th session, the was reduced and it was around 4~6.5 Seconds.

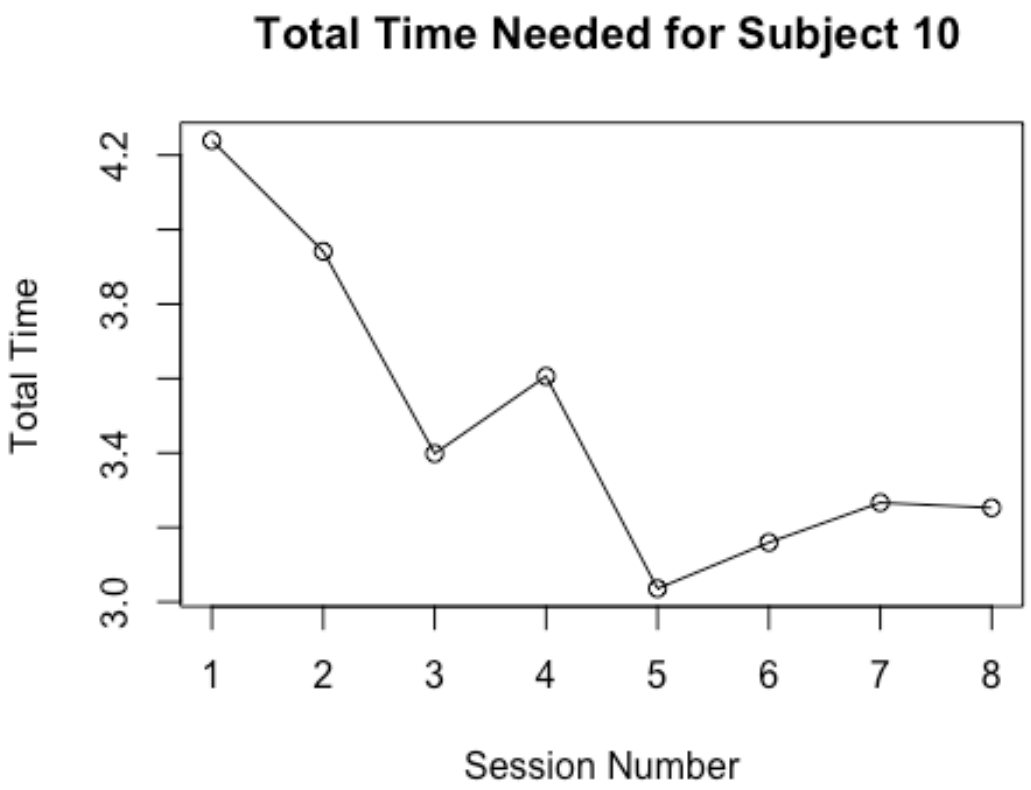
### EDA with Total Time vs Session Index

```
#Total time for Subject 02
sub02.tt <- EDA.TT[which(subject=='s002'),]
#Aggregating Mean Total Time by Session Index
sub02.tt.time <- aggregate(TT.Time ~  sessionIndex, data = sub02.tt, mean)

#Total time for Subject 10
sub10.tt <- EDA.TT[which(subject=='s010'),]
#Aggregating Mean Total Time by Session Index
sub10.tt.time <- aggregate(TT.Time ~  sessionIndex, data = sub10.tt, mean)

#Plotting Total Time vs Session Number for subject 02
```

```
plot(sub02.tt.time$sessionIndex,sub02.tt.time$TT.Time,
     type ='o', ylab = 'Total Time'  ,
     xlab = 'Session Number',
     main = ' Total Time Needed for Subject 2')
```




```
#Plotting Total Time vs Session Number for subject 10
plot(sub10.tt.time$sessionIndex,sub10.tt.time$TT.Time,
     type ='o', ylab = 'Total Time'  ,
     xlab = 'Session Number',
     main = ' Total Time Needed for Subject 10')
```

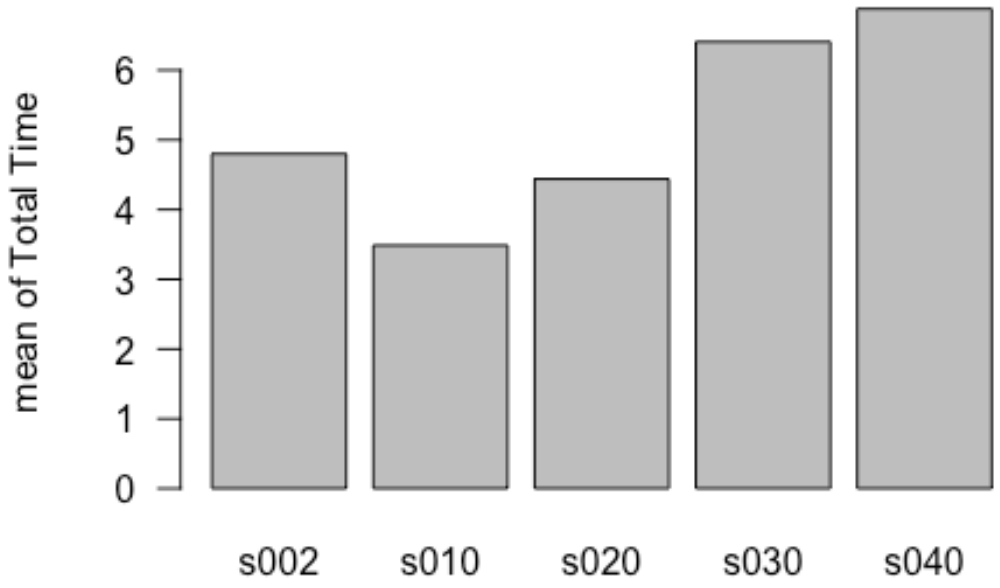



From the curve above, we can see that the average total time for per session index reduces indicating that with each session the users are more familiar with the keystroke of the password pattern, hence reducing the total average time. ALso we can see that total time for subject 02 and subject 10 are not same for each session which suggests variability of typing per subject. You can get more idea from [https://r-coder.com/aggregate-r/] about the *aggregate* function.

### Barplot with Total Time for Different Subject

```
attach(EDA.TT)
#Selecting Total Time for various users
sub2.tt <- EDA.TT[which(subject=='s002'),]
sub10.tt <- EDA.TT[which(subject=='s010'),]
sub20.tt <- EDA.TT[which(subject=='s020'),]
sub30.tt <- EDA.TT[which(subject=='s030'),]
sub40.tt <- EDA.TT[which(subject=='s040'),]

#Merging the total time for selected users
sub.all <- rbind(sub2.tt, sub10.tt,sub20.tt,sub30.tt,sub40.tt)
#Aggregating Mean Total Time by Subject
sub.all.mean <- aggregate(TT.Time ~  subject, data = sub.all, mean)
#Plotting Bar Plot
```

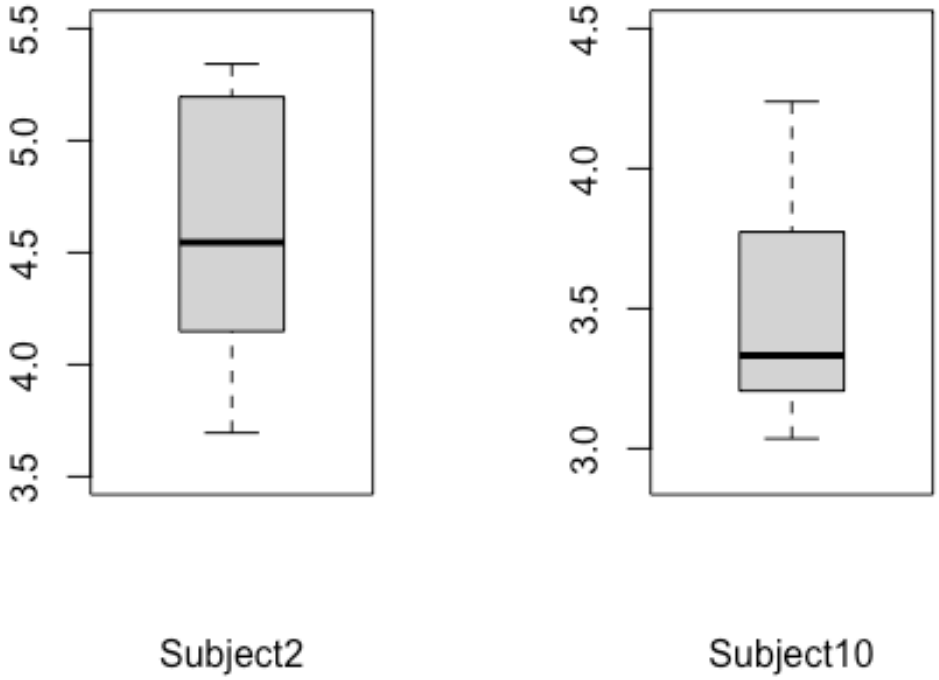
```
barplot(sub.all.mean[,2], names.arg=sub.all.mean$subject, ylab="mean of Total
Time", las=1)
```




From the barplot, we can observe difference in the total mean time for various users.

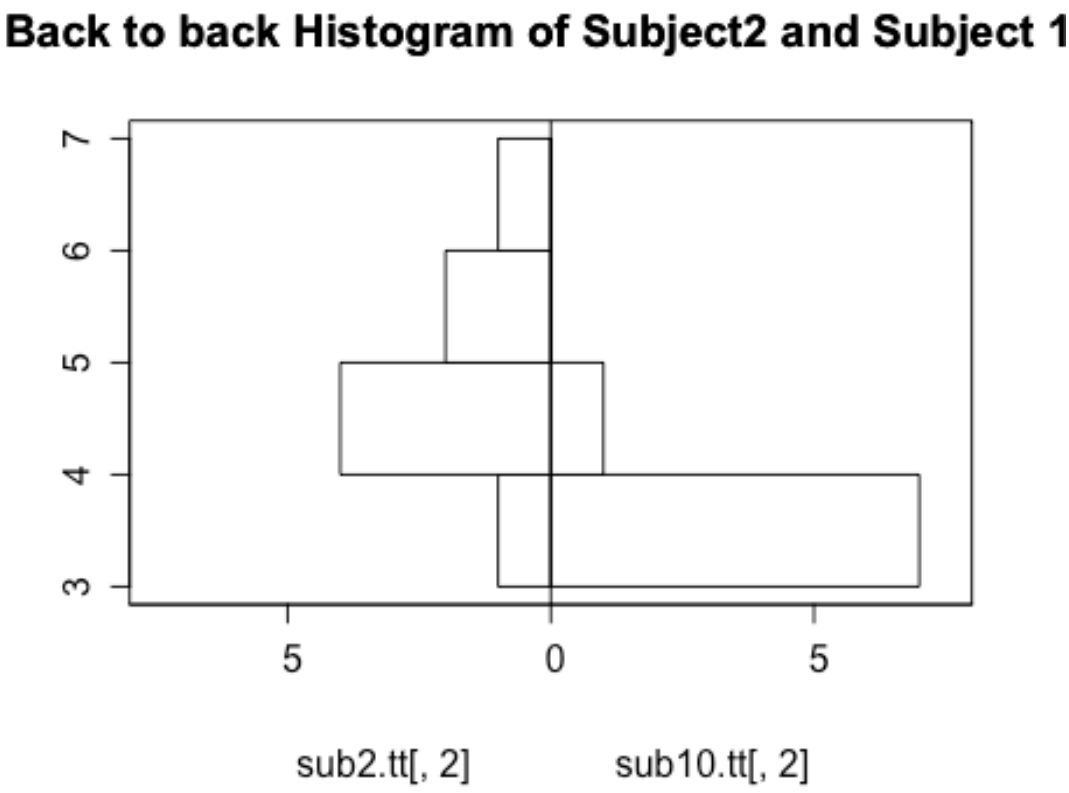
### Boxplot with Total Time for Different Subject

```
sub2.tt<- aggregate(TT.Time ~  sessionIndex, data = sub2.tt, mean)
sub10.tt<- aggregate(TT.Time ~  sessionIndex, data = sub10.tt, mean)

par(mfrow=c(1,2))
boxplot(sub2.tt[,2], ylim = c(3.5,5.5), xlab= 'Subject2')
boxplot(sub10.tt[,2], ylim = c(2.9,4.5), xlab= 'Subject10')
```



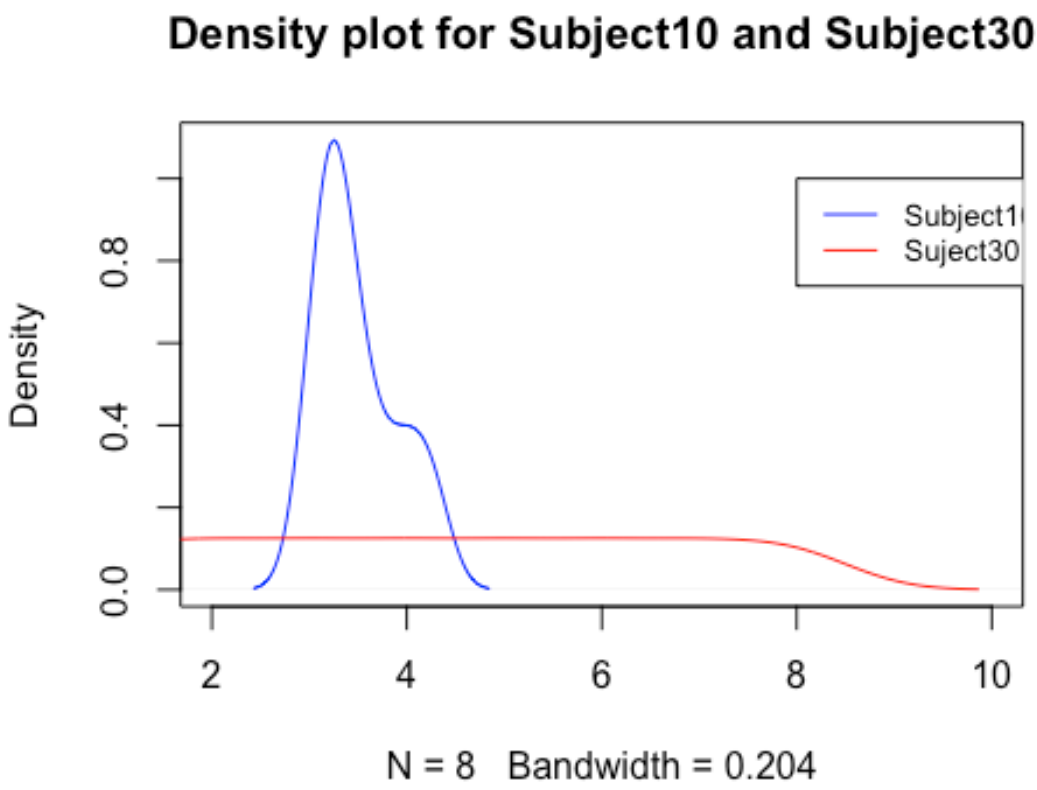

From the boxplot, we can see that the boxes of the boxplots do not overlap with each other. So ther is difference between the three groups.

```
#Plotting Back to back Histogram
histbackback(sub2.tt[,2],sub10.tt[,2], xlim = c(-8,8), main = 'Back to back H
istogram of Subject2 and Subject 10')
```




From the backtoback histogram plot above, Comparisons within the total mean time of subject 2 and subject 10 are made on a "common scale." It clearly indicates difference between the total time for the users.

```
plot(density(sub10.tt[,2]), xlim= c(2,10), main = 'Density plot for Subject10
and Subject30', col= 'blue')
lines(density(sub30.tt[,2]), col = 'red')
legend(8, 1, legend=c("Subject10", "Suject30"),
       col=c("blue", "red"), lty=1:1, cex=0.8)
```




From the density plot, we can see that the average total time for subject30 and subject 10 does not overlap indicating the difference between the groups.

### Test Statistic for Different Subjects

One of the most common tests in statistics is the t-test, used to determine whether the means of two groups are equal to each other. The assumption for the test is that both groups are sampled from normal distributions with equal variances. The null hypothesis is that the two means are equal, and the alternative is that they are not. It is known that under the null hypothesis, we can calculate a t-statistic that will follow a t-distribution with n1 + n2 - 2 degrees of freedom.[Link: https://statistics.berkeley.edu/computing/r-t-tests]

```
attach(EDA.TT)
#Selecting Total Time for various users
sub30.tt <- EDA.TT[which(subject=='s030'),]
sub40.tt <- EDA.TT[which(subject=='s040'),]
# Compute t-test
sub30.40 <- rbind(sub30.tt,sub40.tt)
res <- t.test(TT.Time ~ subject, data = sub30.40, var.equal = TRUE)
res

##
##  Two Sample t-test
```

```
##
## data:  TT.Time by subject
## t = -4.169, df = 798, p-value = 3.395e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6950286 -0.2500449
## sample estimates:
## mean in group s030 mean in group s040
##           6.406774           6.879311
```

The differences between ~ and , is the type of statistical test you are running. ~ gives you the mean differences. This is for dependent samples (e.g. before and after). , gives you the difference in means. This is for independent samples (e.g. treatment and control). These two tests are not interchangeable. From the Welch Two Sample t-test, we found p-value less than the significance level so we can reject our null hypothesis and say that the mean differences for the total time between user 30 and 40 are not same.

```
#Do the two populations have the same variances?
res.ftest <- var.test(TT.Time ~ subject, data = sub30.40)
res.ftest

##
##  F test to compare two variances
##
## data:  TT.Time by subject
## F = 0.6934, num df = 399, denom df = 399, p-value = 0.0002679
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.5696853 0.8439803
## sample estimates:
## ratio of variances
##          0.6933997
```

The p-value of F-test is p = 0.0002679. It's less than the significance level alpha = 0.05. In conclusion, there is significant difference between the variances of the two sets of data.

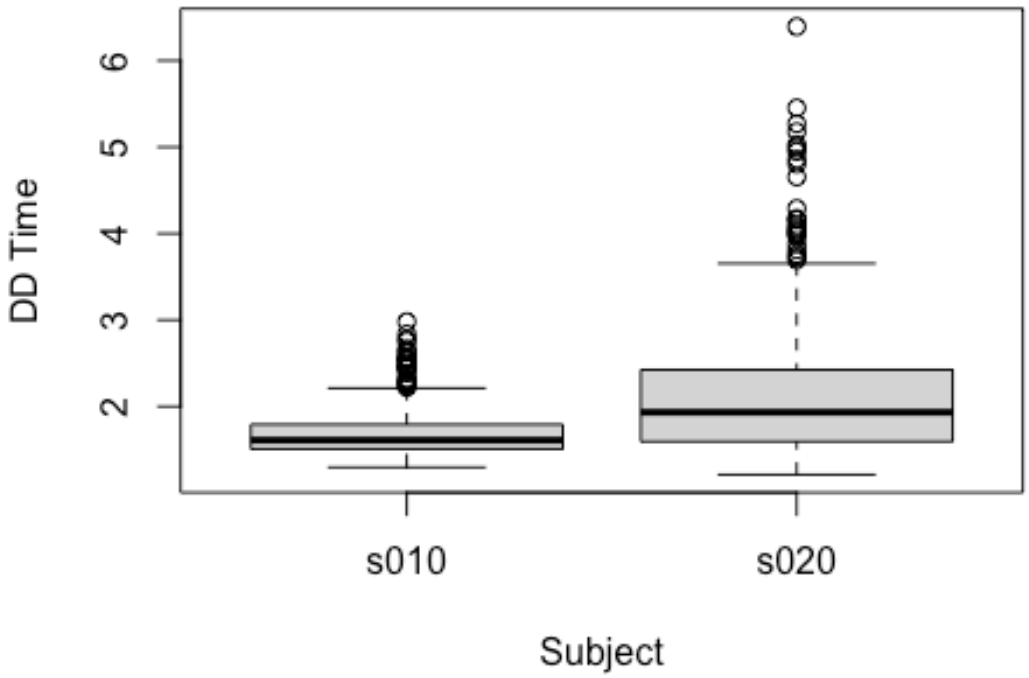
### EDA with Total Down-Down, Up-Down and Hold Time

```
#Total DD, UD and Hold Time
DD.TT.Time <- rowSums(DD.Time[ , c(4:13)])
DD.TT.Time <- cbind(DD.Time[,1:3],DD.TT.Time)
UD.TT.Time <- rowSums(UD.Time[ , c(4:13)])
UD.TT.Time <- cbind(UD.Time[,1:3],UD.TT.Time)
Hold.TT.Time <- rowSums(Hold.Time[ , c(4:13)])
Hold.TT.Time <- cbind(Hold.Time[,1:3],Hold.TT.Time)

attach(DD.TT.Time)
sub10.dd.tt <- DD.TT.Time[which(subject=='s010'),]
sub20.dd.tt <- DD.TT.Time[which(subject=='s020'),]
sub.dd.all <- rbind(sub10.dd.tt,sub20.dd.tt)
sub.dd.all.mean <- as.data.frame(aggregate(DD.TT.Time ~  subject, data = sub.
```

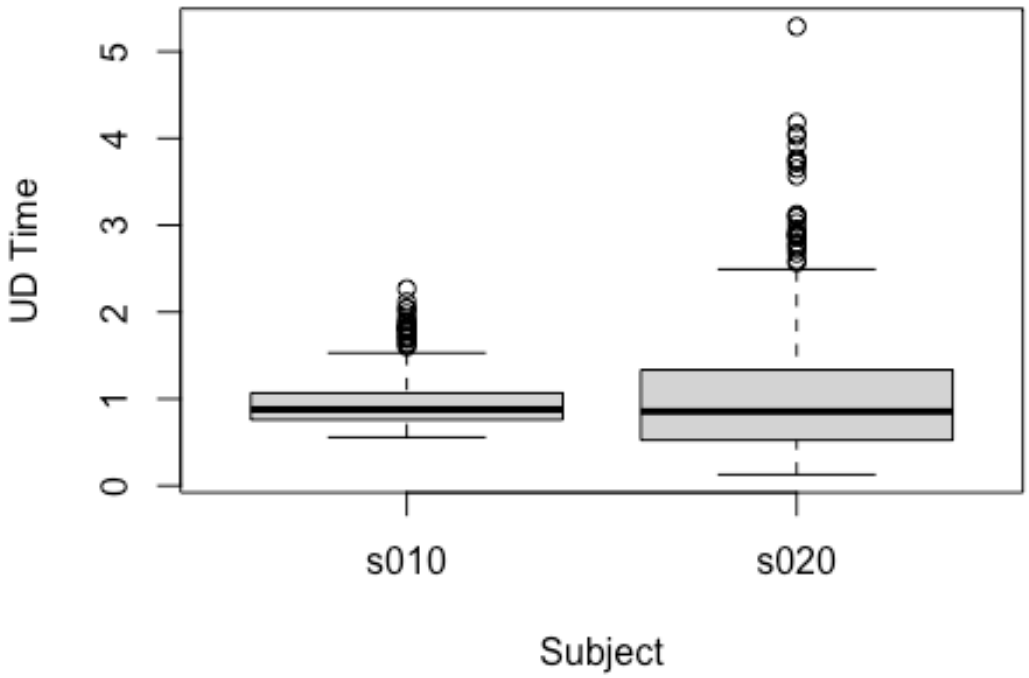
```
dd.all, mean))

attach(UD.TT.Time)
sub10.ud.tt <- UD.TT.Time[which(subject=='s010'),]
sub20.ud.tt <- UD.TT.Time[which(subject=='s020'),]
sub.ud.all <- rbind(sub10.ud.tt,sub20.ud.tt)
sub.ud.all.mean <- as.data.frame(aggregate(UD.TT.Time ~  subject, data = sub.
ud.all, mean))

attach(Hold.TT.Time)
sub10.hold.tt <- Hold.TT.Time[which(subject=='s010'),]
sub20.hold.tt <- Hold.TT.Time[which(subject=='s020'),]
sub.hold.all <- rbind(sub10.hold.tt,sub20.hold.tt)
sub.hold.all.mean <- as.data.frame(aggregate(Hold.TT.Time ~  subject, data =
sub.hold.all, mean))

sub.all.mean <- cbind(sub.dd.all[,1],sub.dd.all[,4],sub.ud.all[,4], sub.hold.
all[,4])
sub.all.mean <- as.data.frame(sub.all.mean)
colnames(sub.all.mean) <- c('subject', 'DD Time',
                    'UD Time','Hold Time')

sub.all.mean[,2] <- as.numeric(sub.all.mean[,2])
sub.all.mean[,3] <- as.numeric(sub.all.mean[,3])
sub.all.mean[,4] <- as.numeric(sub.all.mean[,4])

typeof(sub.all.mean[,4])
```

```
## [1] "double"
```

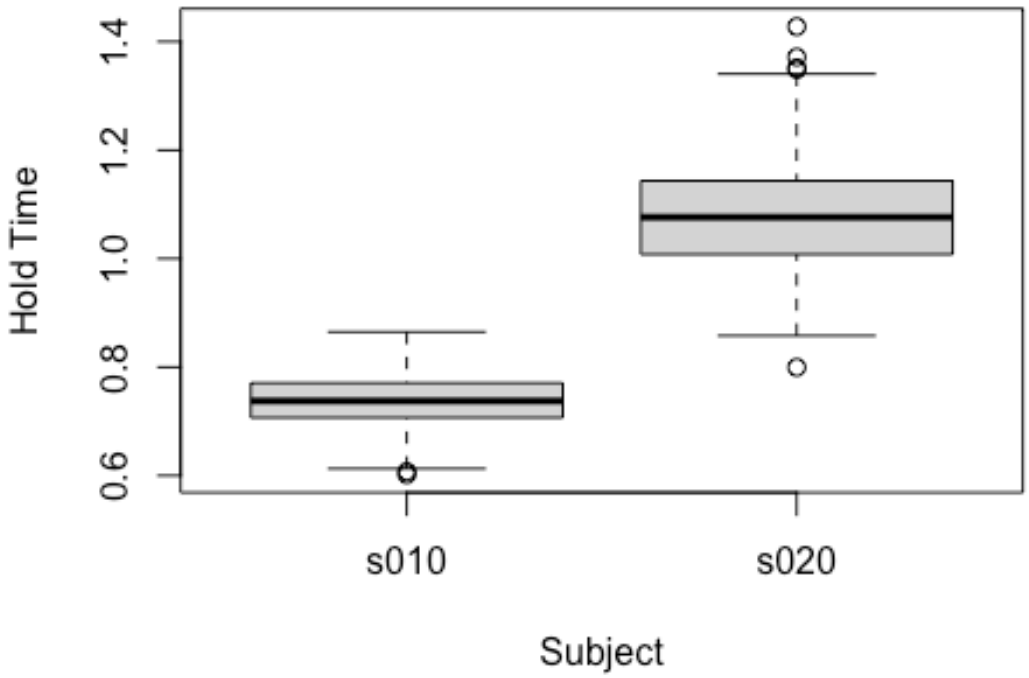
```
#boxplots and t-tests for the 4 variables at once
for (i in 2:4) { # variables to compare are variables 1 to 4
  boxplot(sub.all.mean[, i] ~ sub.all.mean$subject, # draw boxplots by group
    ylab = names(sub.all.mean[i]), # rename y-axis with variable's name
    xlab = "Subject"
  )
  print(t.test(sub.all.mean[, i] ~sub.all.mean$subject)) # print results of t
-test
}
```




```
##
##  Welch Two Sample t-test
##
## data:  sub.all.mean[, i] by sub.all.mean$subject
## t = -10.493, df = 499.42, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.550232 -0.376673
## sample estimates:
## mean in group s010 mean in group s020
##           1.704527           2.167979
```




```
##
##  Welch Two Sample t-test
##
## data:  sub.all.mean[, i] by sub.all.mean$subject
## t = -2.8321, df = 508.3, p-value = 0.004807
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.20435213 -0.03695737
## sample estimates:
## mean in group s010 mean in group s020
##          0.9689473          1.0896020
```



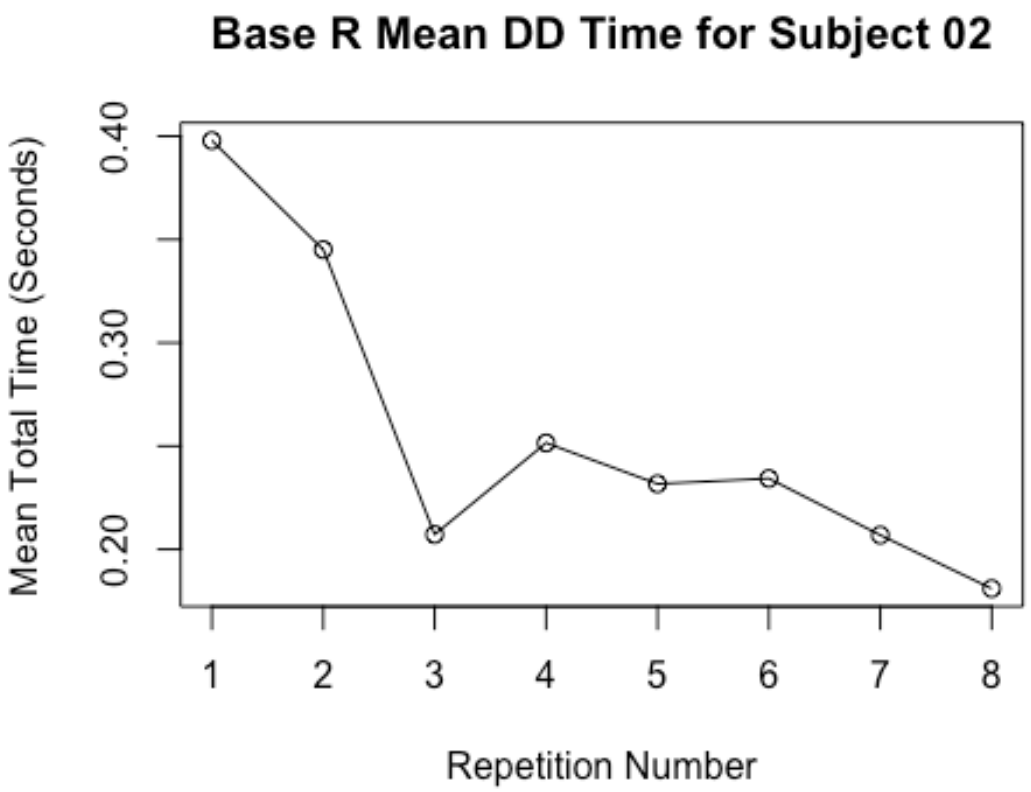

```
##
##  Welch Two Sample t-test
##
## data:  sub.all.mean[, i] by sub.all.mean$subject
## t = -60.413, df = 557.85, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.3539432 -0.3316523
## sample estimates:
## mean in group s010 mean in group s020
##          0.7355793          1.0783770
```

We have summed up all the total time for DD, UD and Hold Time Features and plotted boxplots for two different users. From the boxplot, depicted above, we can clearly see that DD Time Feature is different for different users.

### EDA of Keydown-Keydown Feature

```
#subsetting variables based on different subject
attach(DD.Time) # Keydown-Keydown data
# Selecting subject 002 with all repitions: DD pressing time
sub2.rep1 <- DD.Time[which(subject=='s002' & rep == 1),]
sub2.rep2 <- DD.Time[which(subject=='s002' & rep == 2),]
sub2.rep3 <- DD.Time[which(subject=='s002' & rep == 3),]
```

```r
sub2.rep4 <- DD.Time[which(subject=='s002' & rep == 4),]
sub2.rep5 <- DD.Time[which(subject=='s002' & rep == 5),]
sub2.rep6 <- DD.Time[which(subject=='s002' & rep == 6),]
sub2.rep7 <- DD.Time[which(subject=='s002' & rep == 7),]
sub2.rep8 <- DD.Time[which(subject=='s002' & rep == 8),]
# combining all rows of the sorted DD time data
sub2.combined <- rbind(sub2.rep1, sub2.rep2,
                       sub2.rep3,sub2.rep4,
                       sub2.rep5,sub2.rep6,
                       sub2.rep7,sub2.rep8)

#taking average of feature columns
s2rep1.mean <- colMeans(sub2.rep1[1,c(3:12)])
s2rep2.mean <- colMeans(sub2.rep2[1,c(3:12)])
s2rep3.mean <- colMeans(sub2.rep3[1,c(3:12)])
s2rep4.mean <- colMeans(sub2.rep4[1,c(3:12)])
s2rep5.mean <- colMeans(sub2.rep5[1,c(3:12)])
s2rep6.mean <- colMeans(sub2.rep6[1,c(3:12)])
s2rep7.mean <- colMeans(sub2.rep7[1,c(3:12)])
s2rep8.mean <- colMeans(sub2.rep8[1,c(3:12)])
# combining mean values
sub2.combm <- rbind(s2rep1.mean,s2rep2.mean,
                    s2rep3.mean, s2rep4.mean,
                    s2rep5.mean,s2rep6.mean,
                    s2rep7.mean,s2rep8.mean)
plot(sub2.combm,
     type ='o',
     ylab = 'Mean Total Time (Seconds)'  ,
     xlab = 'Repetition Number',
     main= 'Base R Mean DD Time for Subject 02')
```




```
#first five DD columns for the same user
s2 <- as.data.frame(sub2.combm[,c(1:5)])
rownames(s2) <- c('Session1', 'Session2',
                  'Session3','Session4',
                  'Session5','Session6',
                   'Session7','Session8')

kable(s2,
      caption = "Five DD Time Mean values of User Subject 2")
```

*Five DD Time Mean values of User Subject 2*

| | rep | DD.period.t | DD.t.i | DD.i.e | DD.e.five |
|---|---|---|---|---|---|
| Session1 | 1 | 0.3979 | 0.1674 | 0.2212 | 1.1885 |
| Session2 | 2 | 0.3451 | 0.1283 | 0.1357 | 1.1970 |
| Session3 | 3 | 0.2072 | 0.1291 | 0.1542 | 1.0408 |
| Session4 | 4 | 0.2515 | 0.2495 | 0.2038 | 1.0556 |
| Session5 | 5 | 0.2317 | 0.1676 | 0.1589 | 0.8629 |
| Session6 | 6 | 0.2343 | 0.1299 | 0.1412 | 0.9373 |
| Session7 | 7 | 0.2069 | 0.1368 | 0.1407 | 0.7967 |

```
 Session8      8        0.1810   0.1378   0.1367      0.6447
```

```
#Summary Statictics
kable(summary(s2),
      caption = "Summary Statistics of First 5 DD Time: Subject 02")
```

*Summary Statistics of First 5 DD Time: Subject 02*

| rep | DD.period.t | DD.t.i | DD.i.e | DD.e.five |
|---|---|---|---|---|
| Min. :1.00 | Min. :0.1810 | Min. :0.1283 | Min. :0.1357 | Min. :0.6447 |
| 1st Qu.:2.75 | 1st Qu.:0.2071 | 1st Qu.:0.1297 | 1st Qu.:0.1397 | 1st Qu.:0.8464 |
| Median :4.50 | Median :0.2330 | Median :0.1373 | Median :0.1477 | Median :0.9890 |
| Mean :4.50 | Mean :0.2570 | Mean :0.1558 | Mean :0.1615 | Mean :0.9654 |
| 3rd Qu.:6.25 | 3rd Qu.:0.2749 | 3rd Qu.:0.1674 | 3rd Qu.:0.1701 | 3rd Qu.:1.0888 |
| Max. :8.00 | Max. :0.3979 | Max. :0.2495 | Max. :0.2212 | Max. :1.1970 |

In our table 1, we have shown mean values of only five columns where

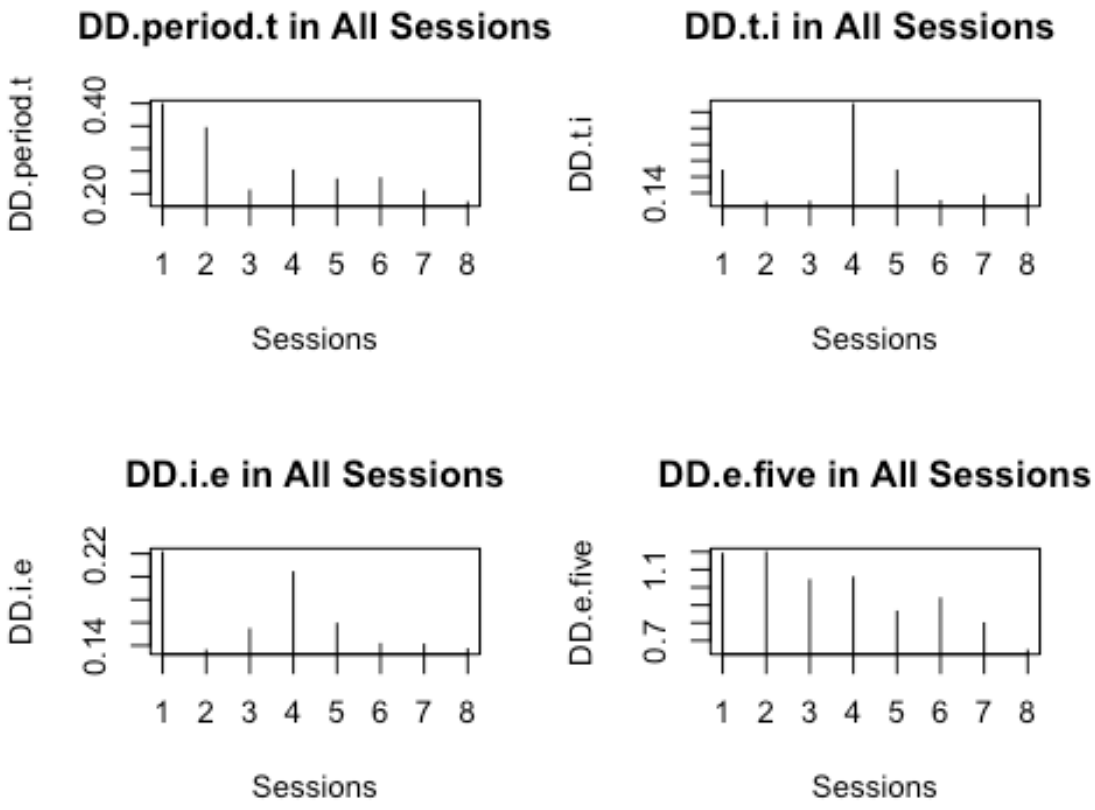
- [DD.period.t:] The time between pressing down the "." key to the time to press down the "t" key.
- [DD.t.i:] The time between pressing down the "t" key to the time to press down the "i" key.
- [DD.i.e:] The time between pressing down the "i" key to the time to press down the "e" key.
- [DD.e.five:] The time between pressing down the "e" key to the time to press down the "5" key.
- [DD.five.Shift.r:] The time between pressing down the "5" key to the time to press down the "shift+r" key combination.

We are considering only mean values of typing and recored the mean values of down pressing time. For the *DD.period.t*, when subject 2 typed for the first time, he/she took 0.3979 seconds and then for the 4th repetition, time was reduced to 0.2515 seconds and lastly for the 8th session time was only 0.1810. So with every session, subject 2 was more and more used to typing which suggests a definite pattern for subject 2. Same scenario is observed for the rest of the used columns.

In our second table we have given a summary statistics with minimum, maximum, mean and median values of first 5 Down-Down pressing time mean. Measures of central tendency are a class of statistics used to identify a value that falls in the middle of a set of data. From our summary statistic for the numeric variables, we can say that

- Average value of *DD.period.t Mean* suggests that typical pressing time for the subject 02 would go around 0.2768 seconds and we can also say that data is symmetrical as mean (0.2768) and median (0.2515 Seconds) are close. We can observe similar scenario for the rest of the columns stating that data is symmetrical.

```
attach(s2)
par(mfrow=c(2,2))    # set the plotting area into a 1*2 array
plot(DD.period.t,
     type = "h",
     xlab = "Sessions",
     main = "DD.period.t in All Sessions")
plot(DD.t.i,
     type = "h",
     xlab = "Sessions",
     main = "DD.t.i in All Sessions ")
plot(DD.i.e,
     type = "h",
     xlab = "Sessions",
     main = "DD.i.e in All Sessions")
plot(DD.e.five,
     type = "h",
     xlab = "Sessions",
     main = "DD.e.five in All Sessions")
```

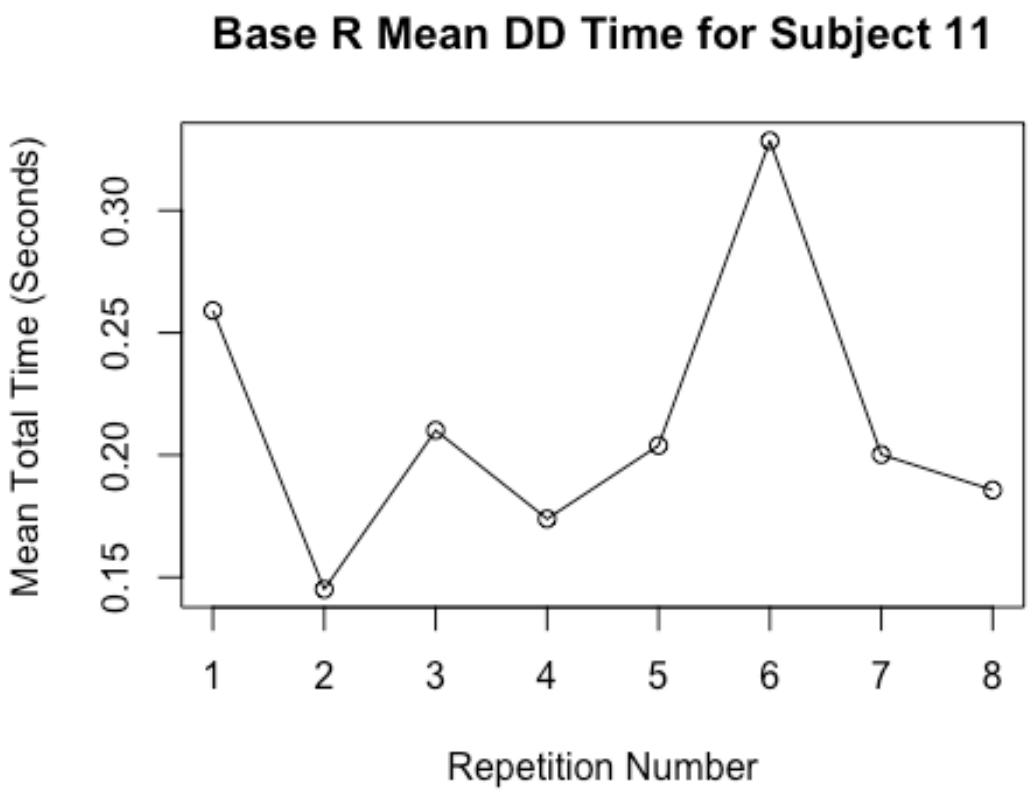



*Base R plot of 8 Sessions of Subject 02*

We have plotted features like *DD.period.t, DD.t.i* in Base R. If we look at session 1, it took more time than the session 8 for subject 2. For every feature, last session timing was reduced. For an unknown user it would not be a same scenario.

```
#subsetting variables based on different subject
attach(DD.Time) # Keydown-Keydown data
# Selecting subject 11 with all repitions: DD pressing time
sub11.rep1 <- DD.Time[which(subject=='s011' & rep == 1),]
sub11.rep2 <- DD.Time[which(subject=='s011' & rep == 2),]
sub11.rep3 <- DD.Time[which(subject=='s011' & rep == 3),]
sub11.rep4 <- DD.Time[which(subject=='s011' & rep == 4),]
sub11.rep5 <- DD.Time[which(subject=='s011' & rep == 5),]
sub11.rep6 <- DD.Time[which(subject=='s011' & rep == 6),]
sub11.rep7 <- DD.Time[which(subject=='s011' & rep == 7),]
sub11.rep8 <- DD.Time[which(subject=='s011' & rep == 8),]
# combining all rows of the sorted DD time data
sub11.combined <- rbind(sub11.rep1, sub11.rep2,
                        sub11.rep3,sub11.rep4,
                        sub11.rep5,sub11.rep6,
                        sub11.rep7,sub11.rep8)
```

```r
#taking average of feature columns
s11rep1.mean <- colMeans(sub11.rep1[1,c(3:12)])
s11rep2.mean <- colMeans(sub11.rep2[1,c(3:12)])
s11rep3.mean <- colMeans(sub11.rep3[1,c(3:12)])
s11rep4.mean <- colMeans(sub11.rep4[1,c(3:12)])
s11rep5.mean <- colMeans(sub11.rep5[1,c(3:12)])
s11rep6.mean <- colMeans(sub11.rep6[1,c(3:12)])
s11rep7.mean <- colMeans(sub11.rep7[1,c(3:12)])
s11rep8.mean <- colMeans(sub11.rep8[1,c(3:12)])
# combining mean values
sub11.combm <- rbind(s11rep1.mean,s11rep2.mean,
                     s11rep3.mean, s11rep4.mean,
                     s11rep5.mean,s11rep6.mean,
                     s11rep7.mean,s11rep8.mean)

plot(sub11.combm,
     type ='o',
     ylab = 'Mean Total Time (Seconds)'  ,
     xlab = 'Repetition Number',
     main= 'Base R Mean DD Time for Subject 11')
```




```r
#first five DD columns for the same user
s11 <- as.data.frame(sub11.combm[,c(1:5)])
```

```
rownames(s11) <- c('Session1', 'Session2',
                   'Session3','Session4',
                   'Session5','Session6',
                    'Session7','Session8')

kable(s11,
      caption = "Five DD Time Mean values of User Subject 11")
```

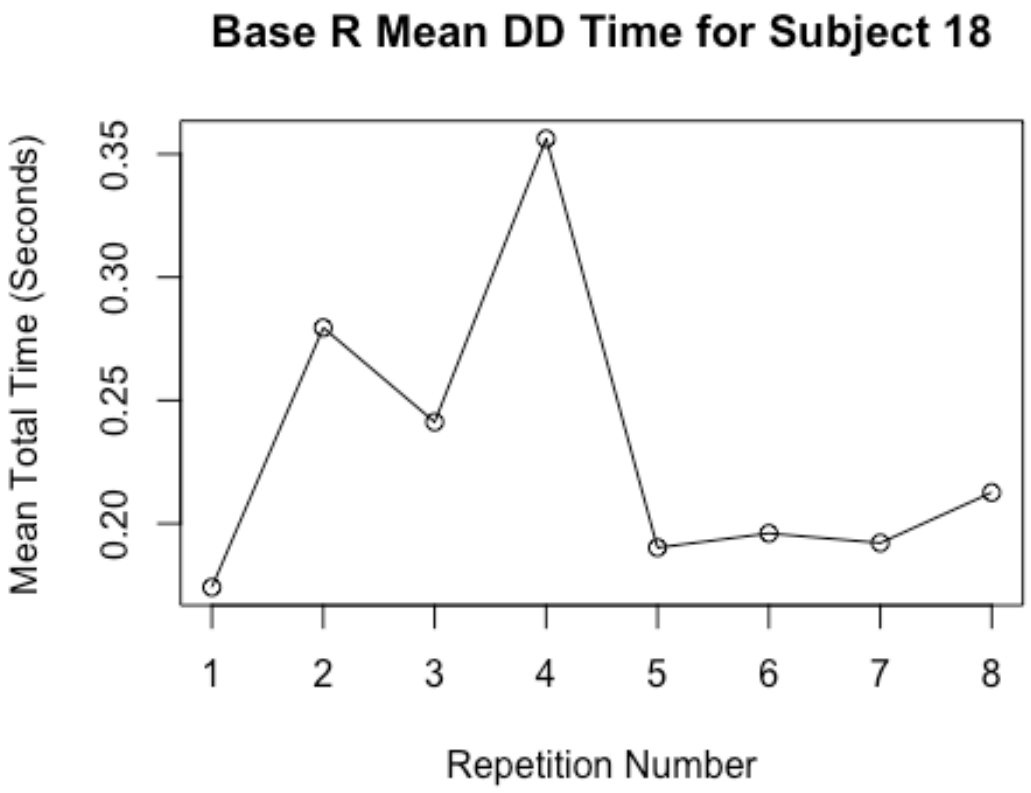
*Five DD Time Mean values of User Subject 11*

| | rep | DD.period.t | DD.t.i | DD.i.e | DD.e.five |
|---|---|---|---|---|---|
| Session1 | 1 | 0.2591 | 0.1035 | 0.1545 | 0.7056 |
| Session2 | 2 | 0.1451 | 0.2397 | 0.1898 | 0.5288 |
| Session3 | 3 | 0.2101 | 0.2093 | 0.1075 | 0.4151 |
| Session4 | 4 | 0.1739 | 0.1075 | 0.1418 | 0.5793 |
| Session5 | 5 | 0.2039 | 0.1350 | 0.1471 | 1.3035 |
| Session6 | 6 | 0.3285 | 0.1325 | 0.8421 | 0.2734 |
| Session7 | 7 | 0.2002 | 0.2360 | 0.1111 | 0.4420 |
| Session8 | 8 | 0.1857 | 0.1481 | 0.6703 | 0.5478 |

The table represents mean values of five DD Time Mean values of User Subject 11 which is different from subject 2. The results mentioned in this table also tells us that keyboard typing pattern of subject 2 and subject 11 are not same.

```
#subsetting variables based on different subject
attach(DD.Time) # Keydown-Keydown data
# Selecting subject 18 with all repitions: DD pressing time
sub18.rep1 <- DD.Time[which(subject=='s018' & rep == 1),]
sub18.rep2 <- DD.Time[which(subject=='s018' & rep == 2),]
sub18.rep3 <- DD.Time[which(subject=='s018' & rep == 3),]
sub18.rep4 <- DD.Time[which(subject=='s018' & rep == 4),]
sub18.rep5 <- DD.Time[which(subject=='s018' & rep == 5),]
sub18.rep6 <- DD.Time[which(subject=='s018' & rep == 6),]
sub18.rep7 <- DD.Time[which(subject=='s018' & rep == 7),]
sub18.rep8 <- DD.Time[which(subject=='s018' & rep == 8),]
# combining all rows of the sorted DD time data
sub18.combined <- rbind(sub18.rep1, sub18.rep2,
                        sub18.rep3,sub18.rep4,
                        sub18.rep5,sub18.rep6,
                        sub18.rep7,sub18.rep8)

#taking average of feature columns
s18rep1.mean <- colMeans(sub18.rep1[1,c(3:12)])
s18rep2.mean <- colMeans(sub18.rep2[1,c(3:12)])
s18rep3.mean <- colMeans(sub18.rep3[1,c(3:12)])
s18rep4.mean <- colMeans(sub18.rep4[1,c(3:12)])
s18rep5.mean <- colMeans(sub18.rep5[1,c(3:12)])
```

```
s18rep6.mean <- colMeans(sub18.rep6[1,c(3:12)])
s18rep7.mean <- colMeans(sub18.rep7[1,c(3:12)])
s18rep8.mean <- colMeans(sub18.rep8[1,c(3:12)])
# combining mean values
sub18.combm <- rbind(s18rep1.mean,s18rep2.mean,
                     s18rep3.mean, s18rep4.mean,
                     s18rep5.mean,s18rep6.mean,
                     s18rep7.mean,s18rep8.mean)
plot(sub18.combm,
     type ='o',
     ylab = 'Mean Total Time (Seconds)'  ,
     xlab = 'Repetition Number',
     main= 'Base R Mean DD Time for Subject 18')
```




```
#first five DD columns for the same user
s18 <- as.data.frame(sub18.combm[,c(1:5)])
rownames(s18) <- c('Session1', 'Session2',
                   'Session3','Session4',
                   'Session5','Session6',
                    'Session7','Session8')

kable(s18,
      caption = "Five DD Time Mean values of User Subject 18")
```

*Five DD Time Mean values of User Subject 18*

| | rep | DD.period.t | DD.t.i | DD.i.e | DD.e.five |
|---|---|---|---|---|---|
| Session1 | 1 | 0.1741 | 0.1365 | 0.1323 | 1.2326 |
| Session2 | 2 | 0.2795 | 0.1811 | 0.1318 | 0.8136 |
| Session3 | 3 | 0.2412 | 0.1051 | 0.1262 | 0.9981 |
| Session4 | 4 | 0.3562 | 0.1286 | 0.5827 | 0.3465 |
| Session5 | 5 | 0.1902 | 0.1847 | 0.0819 | 0.2987 |
| Session6 | 6 | 0.1960 | 0.2589 | 0.1088 | 0.3526 |
| Session7 | 7 | 0.1921 | 0.0787 | 0.1911 | 0.9241 |
| Session8 | 8 | 0.2126 | 0.1144 | 0.1027 | 0.3549 |

Our next step is to find the percentage difference among the user to find an imposter or fraudulance.

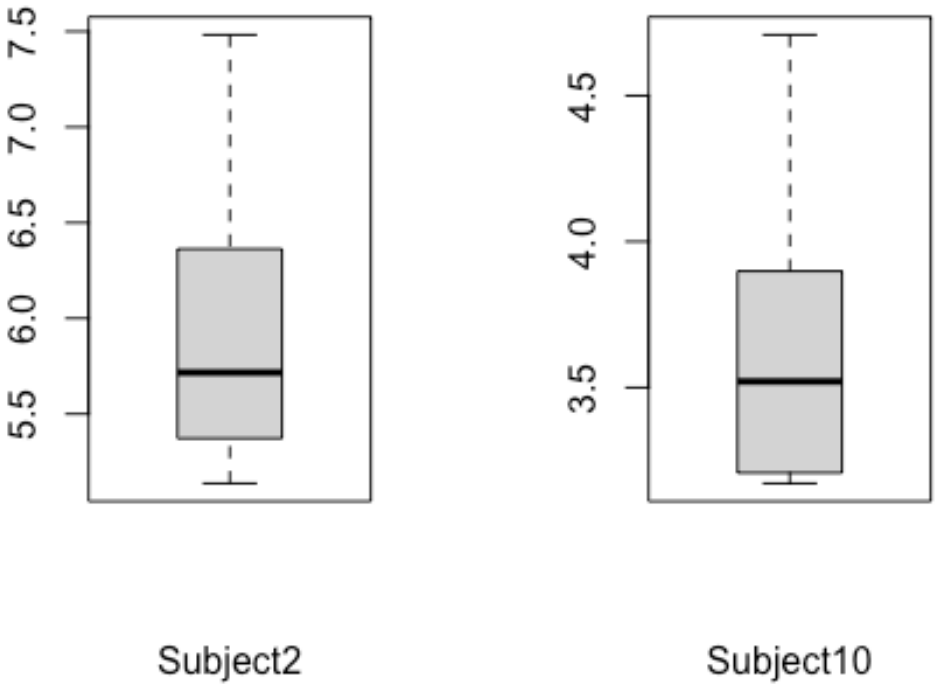
```
s2.ddpt <- as.data.frame(c(0.3979,0.3451,0.2072,0.2515,0.2317,0.2343,0.2069,0
.1810))
s11.ddpt <- as.data.frame(c(0.2591,0.1451,0.2101,0.1739,0.2039,0.3285,0.2002,
0.1857))
# run ordinary t-test
t.test(s2.ddpt, s11.ddpt, alternative = "two.sided")

##
##  Welch Two Sample t-test
##
## data:  s2.ddpt and s11.ddpt
## t = 1.3097, df = 13.03, p-value = 0.2129
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.02832785  0.11560285
## sample estimates:
## mean of x mean of y
## 0.2569500 0.2133125

library(compare)
compare(s2.ddpt, s11.ddpt, allowAll=TRUE)

## FALSE [FALSE]
##   sorted
##   renamed
##   renamed rows
##   dropped names
##   dropped row names
```

We have done a comparison analysis using t.test and ended up having a p-value = 0.2129 which is greater than the significance level kd.main = 0.05}. So we fail to reject our null hypothesis. But using *compare* function, we have seen that they are not same. Therefore,

using t-statistic would not be an appropriate way in comparing different subject results for the keystroke dynamics.

In our next step we will deal with the total time. In our initial step, we just dealt with some of the features. Now we will only deal with the total time and their exploratory data analysis.

```
TotalTime <-cbind(keystroke.main[,1:3],keystroke.main[,35])
colnames(TotalTime) <- c('subject', 'sessionIndex',
                  'rep','TotalTime')
sub5.total <- TotalTime[which(subject=='s005'),]
sub15.total <- TotalTime[which(subject=='s015'),]
sub25.total <- TotalTime[which(subject=='s025'),]
sub35.total <- TotalTime[which(subject=='s035'),]
#Boxplot Comparison
sub5.tt<- aggregate(TotalTime ~  sessionIndex, data = sub5.total, mean)
sub15.tt<- aggregate(TotalTime ~  sessionIndex, data = sub15.total, mean)
par(mfrow=c(1,2))
boxplot(sub5.tt[,2], xlab= 'Subject2')
boxplot(sub15.tt[,2], xlab= 'Subject10')
```

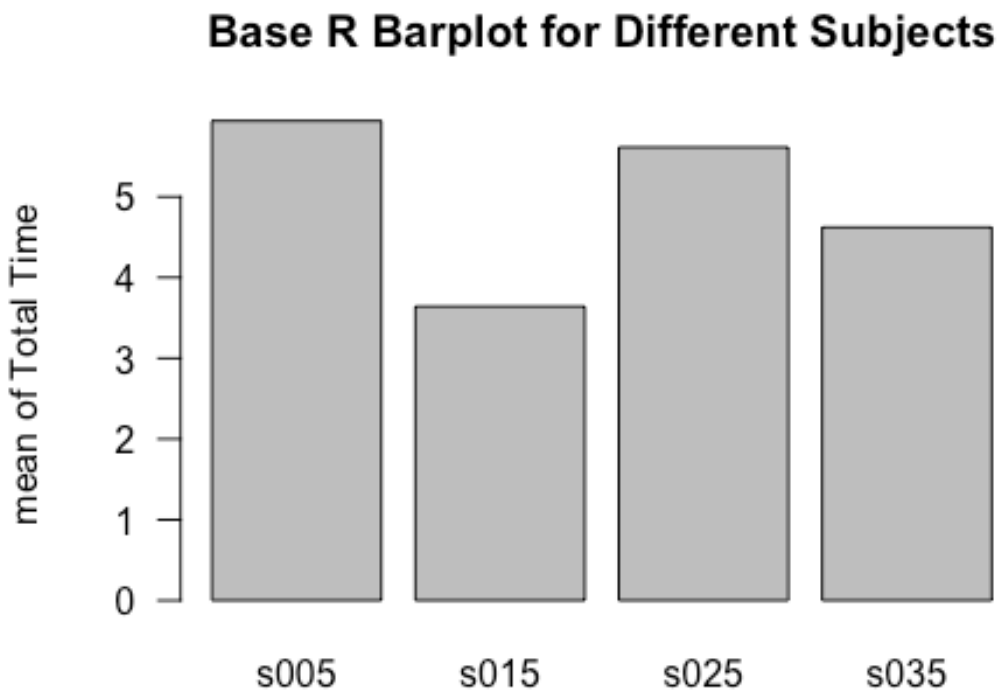



```
#combining all with using rbind
all.subject <- rbind(sub5.total, sub15.total,sub25.total,sub35.total)
```

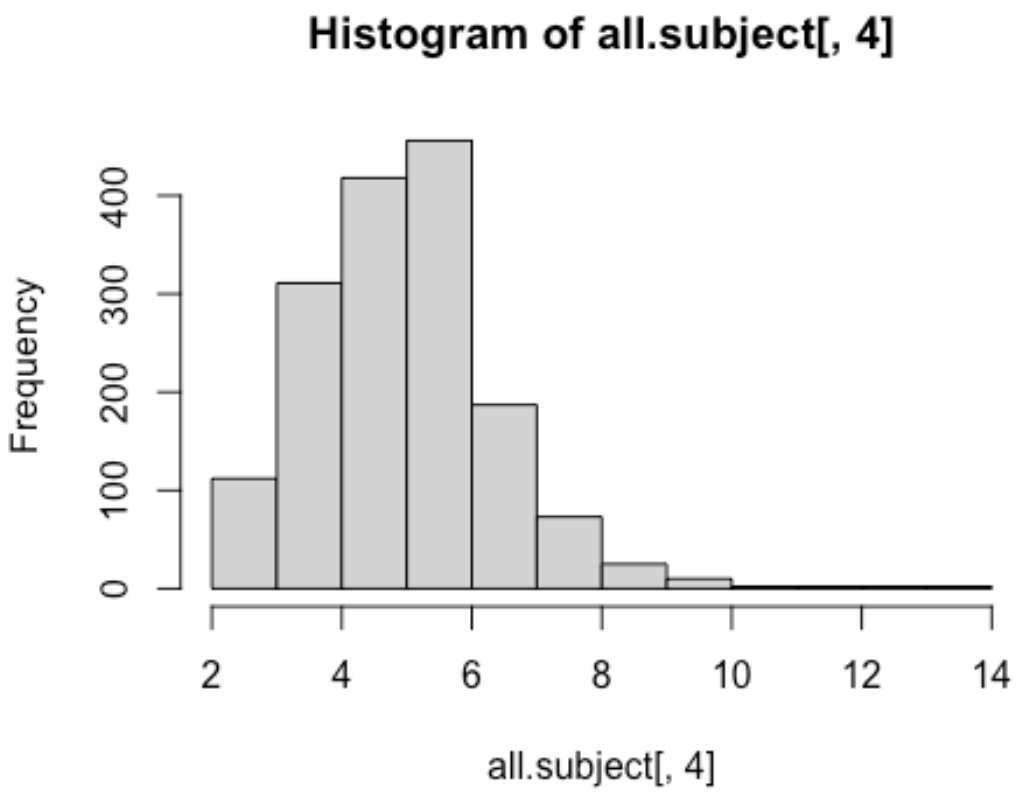
```r
# taking mean of the selected subjects
subject.mean <- aggregate(TotalTime ~  subject, data = all.subject, mean)
```

### Basic plotting For Subject

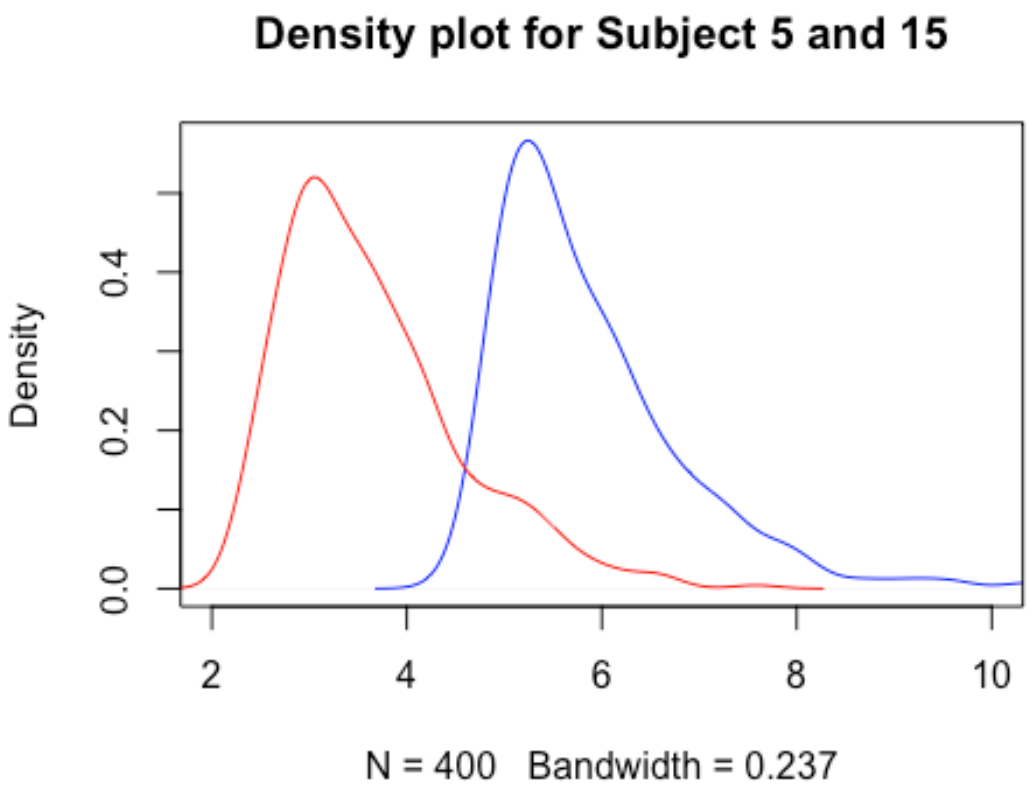
```r
barplot(subject.mean[,2],
        names.arg=subject.mean$subject,
        ylab="mean of Total Time",
        las=1,
        main = 'Base R Barplot for Different Subjects')
```




```r
hist(all.subject[,4])
```




```
t.test(subject.mean[,2])

##
##  One Sample t-test
##
## data:  subject.mean[, 2]
## t = 9.5352, df = 3, p-value = 0.002446
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  3.301022 6.608337
## sample estimates:
## mean of x
##   4.95468

plot(density(sub5.total[,4]),
     xlim= c(2,10),
     main = 'Density plot for Subject 5 and 15',
     col= 'blue')
lines(density(sub15.total[,4]), col = 'red')
```




From the density plot, we can see that the average total time for subject 5and subject 15 does not have any similarity, indicating the difference between the groups.

```
sub40.tt <- TotalTime[which(subject=='s040'),]
sub50.tt <- TotalTime[which(subject=='s050'),]
sub.all <- rbind(sub40.tt, sub50.tt)
sub.all.mean <- aggregate(TotalTime ~  subject+sessionIndex, data = sub.all,
mean)

#Shapiro-Wilk normality test
with(sub.all.mean, shapiro.test(TotalTime[subject == "s040"]))

##
##  Shapiro-Wilk normality test
##
## data:  TotalTime[subject == "s040"]
## W = 0.94189, p-value = 0.6298

with(sub.all.mean, shapiro.test(TotalTime[subject == "s050"]))

##
##  Shapiro-Wilk normality test
```

```
##
## data:  TotalTime[subject == "s050"]
## W = 0.92576, p-value = 0.4783
```

```
var.test(sub.all.mean[,3] ~ subject, data = sub.all.mean)
```

```
##
##  F test to compare two variances
##
## data:  sub.all.mean[, 3] by subject
## F = 2.0951, num df = 7, denom df = 7, p-value = 0.3502
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##   0.4194558 10.4650535
## sample estimates:
## ratio of variances
##           2.095144
```

```
#t-test between subject7 and subject37
t.test(sub.all.mean[,3]~sub.all.mean$subject)
```

```
##
##  Welch Two Sample t-test
##
## data:  sub.all.mean[, 3] by sub.all.mean$subject
## t = 8.9594, df = 12.442, p-value = 8.834e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.835284 3.008641
## sample estimates:
## mean in group s040 mean in group s050
##           6.879311           4.457348
```

- Shapiro-Wilk test is used to detect normality in a model, From the output, the two p-values are greater than the significance level 0.05 implying that the distribution of the data are not significantly different from the normal distribution. In other words, we can assume the normality.
- From the t test, The p-value of the test is 8.834e-078, which is less than the significance level alpha = 0.05. We can conclude that the average total time for subject 50 is significantly different from the average total time for subject 40 with t-statistics value of 8.9594.

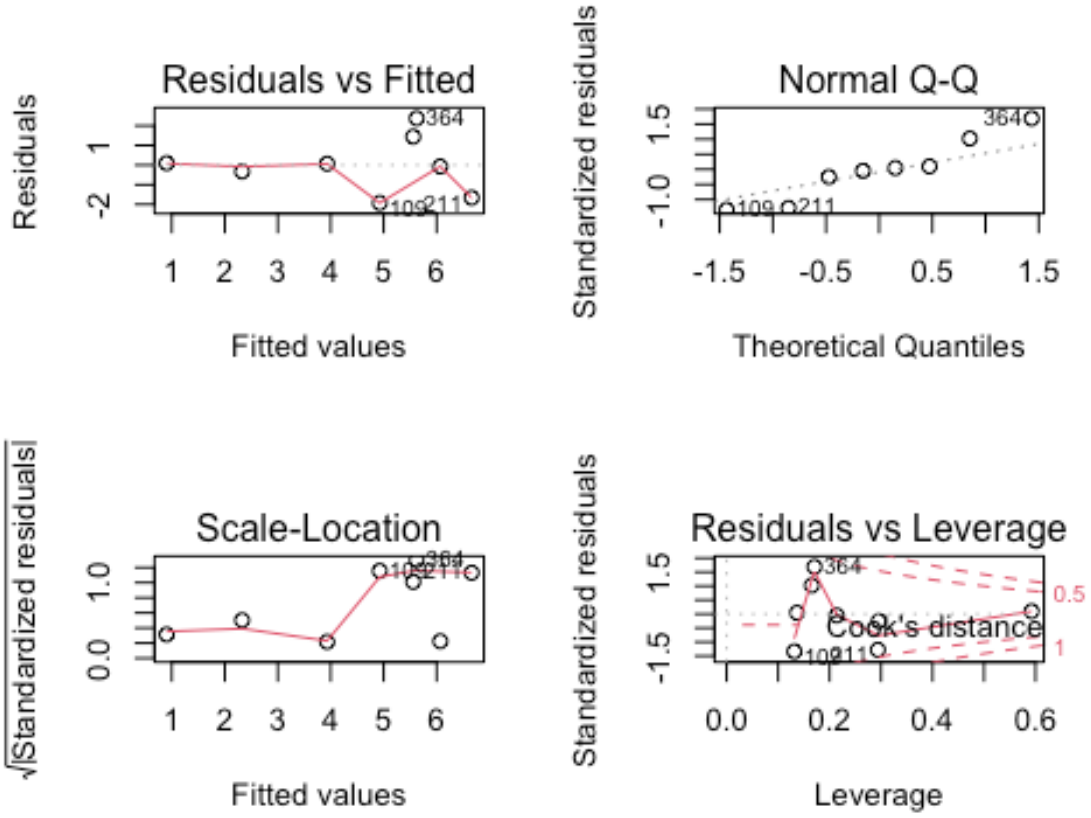
## Post Hoc Analysis

## Linear Regression Model

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable,

and the other is considered to be a dependent variable. For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model.

In this session, we will develop a certain linear model for only one test subjcet. We have chosen subject 10 for developing a regression model where we have taken the average of total time for password input for per session and the sessionIndex as the response variable.




| | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|
| (Intercept) | 21.187696 | 4.931837 | 4.296106 | 0.0051145 |
| TotalTime | -4.785094 | 1.405471 | -3.404619 | 0.0144137 |

```
##
## Call:
## lm(formula = sessionIndex ~ TotalTime, data = model1.lm.10, model = TRUE)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.92575 -0.66122 -0.00081  0.43360  2.37668
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   21.188      4.932   4.296  0.00511 **
## TotalTime     -4.785      1.405  -3.405  0.01441 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.545 on 6 degrees of freedom
## Multiple R-squared:  0.6589, Adjusted R-squared:  0.6021
## F-statistic: 11.59 on 1 and 6 DF,  p-value: 0.01441

## [1] 33.36362
```

Here, the p-value is 0.01441 for TT.Time. We can reject null hypothesis.

The Akaike Information Criterion (AIC) provides a method for assessing the quality of our model through comparison of related models. Much like adjusted R-squared, it's intent is to prevent us from including irrelevant predictors.

Our model with total time has AIC of 33.363

```
#Predicting Linear Models
set.seed(100)  # setting seed to reproduce results of random sampling
trainingRowIndex <- sample(1:nrow(model1.lm.10), .70*nrow(model1.lm.10))  # r
ow indices for training data
trainingData <- model1.lm.10[trainingRowIndex, ]  # model training data
testData  <- model1.lm.10[-trainingRowIndex, ]    # test data
lmMod <- lm(sessionIndex ~ TotalTime, data=trainingData)  # build the model
distPred <- predict(lmMod, testData)  # predict distance
actuals_preds <- data.frame(cbind(actuals=testData$TotalTime, predicteds=dist
Pred))  # make actuals_predicteds dataframe.
correlation_accuracy <- cor(actuals_preds)
head(actuals_preds)

##      actuals predicteds
## 160 3.606010   3.775428
## 211 3.035732   6.577358
## 364 3.252678   5.511444

#min_max_accuracy
(mean(apply(actuals_preds, 1, min) / apply(actuals_preds, 1, max))) *100

## [1] 66.89457

#mean absolute percentage deviation
(mean(abs((actuals_preds$predicteds - actuals_preds$actuals))/actuals_preds$a
ctuals)  )*100

## [1] 63.60204
```

Firstly, We splitted our kd.main into 70 % training data and 30% test data.

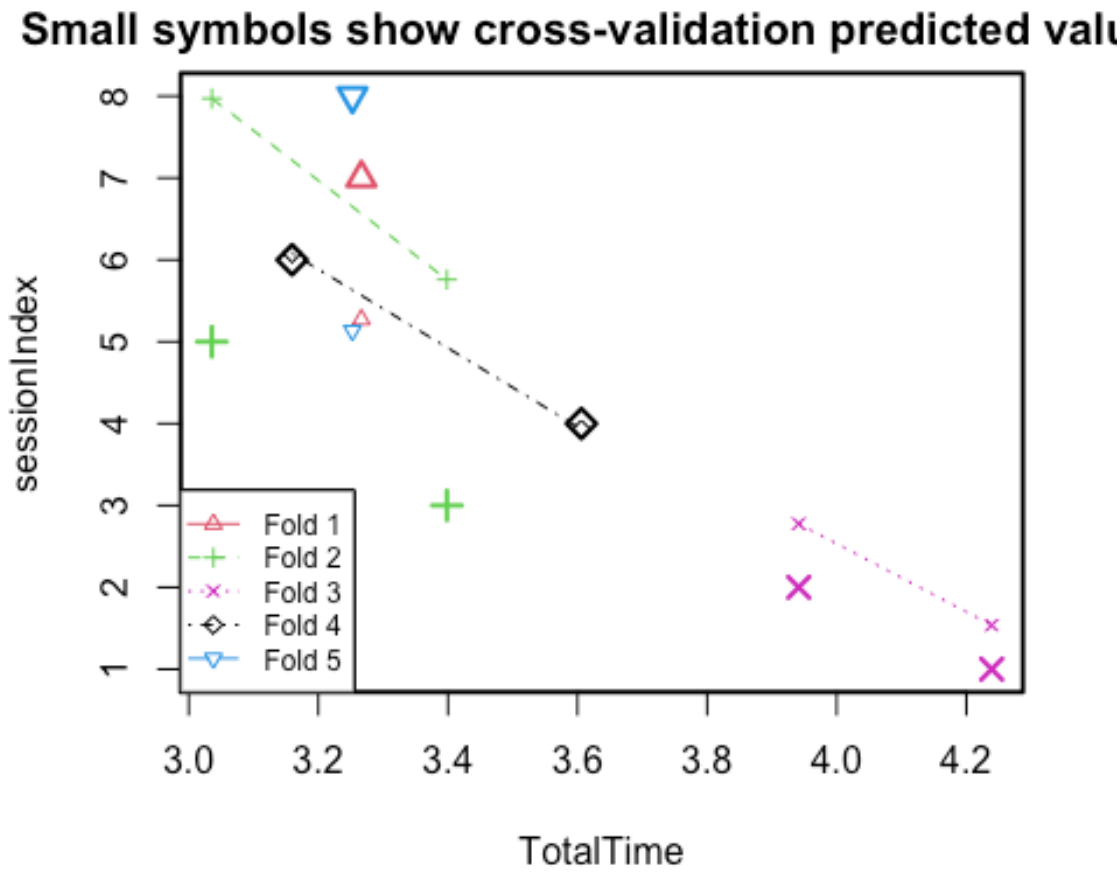
- We have developed our model based on 70 % training data and 30% test data and then we used our developed model to predict the dependent variable on test data.

- By calculating accuracy measures (like min_max accuracy) and error rates (MAPE or MSE), we can find out the prediction accuracy of the model.
- In our case, the min_max accuracy is 66.89457%.

K-fold cross-validation is one of the most commonly used model evaluation methods. Even though this is not as popular as the validation set approach, it can give us a better insight into our data and model.

While the validation set approach is working by splitting the kd.main once, the k-Fold is doing it five or ten times. Imagine you are doing the validation set approach ten times using a different group of data.otherother

```
#k- Fold Cross validation
library(DAAG)
cross.valid <- suppressWarnings(CVlm(data=model1.lm.10, form.lm=linearModel,
m=5,  seed=100,
                                     legend.pos="bottomleft",
                                     printit=FALSE))  # performs the CV
```



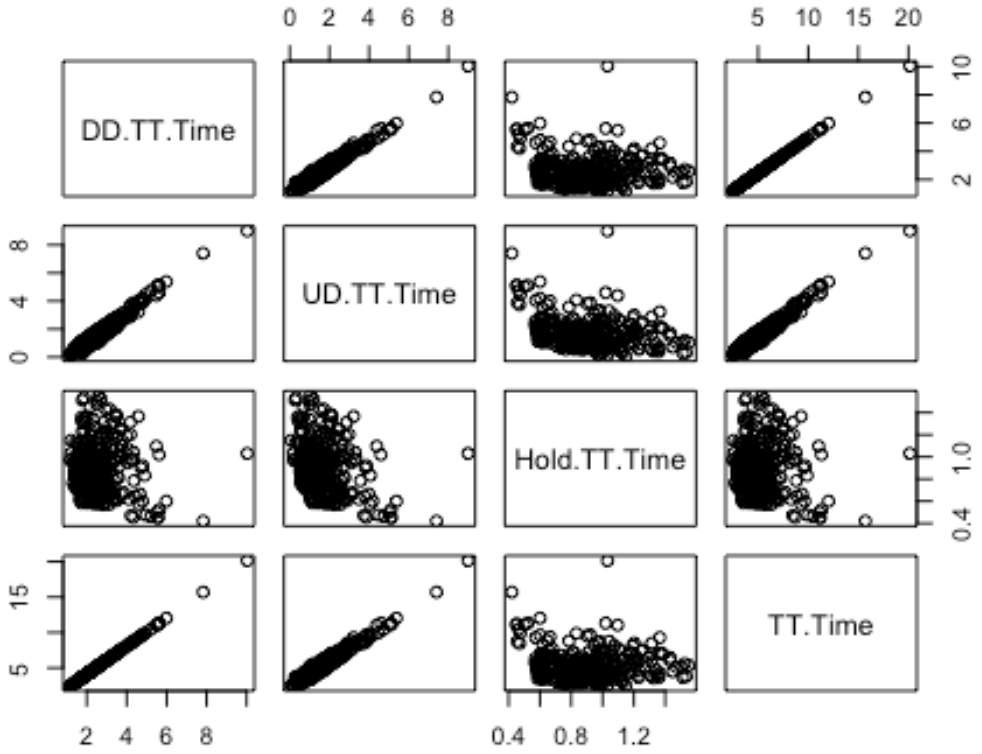

It is important to rigorously test the model's performance as much as possible. One way is to ensure that the model equation you have will perform well, when it is 'built' on a different

subset of training data and predicted on the remaining data. From the plot, we can see the lines of best fit are parallel and as close to each other as possible.

## Multiple Linear Regression

Multiple regression is almost like a linear regression that takes into account the relationship between more than two variables. We have one predictor and one response variable in a basic linear relationship, but we have many predictor variables and one response variable in multiple regression. We may use DD time, UD time, and Hold time as predictor variables in a multiple linear regression with total time as the response variable. For this, we build a subset of these variables from the data set.

```
main.data <-  read_excel("/Users/mominul/Desktop/Project 4/Work Data P4/workd
ata.xlsx")
dataset <- main.data
# Getting total Time for every recitation
TT.Time <- rowSums(dataset[ , c(4:34)])
dataset <- cbind(dataset, TT.Time)
#selecting columns based on three features like DD time, UD time and hold tim
e
DD.Time <- dataset[, c(1,2,3,5,8,11,14,17,20,23,26,29,32)]
UD.Time <- dataset[,c(1,2,3,6,9,12,15,18,21,24,27,30,33)]
Hold.Time<- dataset[,c(1,2,3,4,7,10,13,16,19,22,25,28,31,34)]
#Total DD, UD and Hold Time
DD.TT.Time <- rowSums(DD.Time[ , c(4:13)])
dataset <- cbind(dataset,DD.TT.Time)
UD.TT.Time <- rowSums(UD.Time[ , c(4:13)])
dataset <- cbind(dataset,UD.TT.Time)
Hold.TT.Time <- rowSums(Hold.Time[ , c(4:13)])
dataset <- cbind(dataset,Hold.TT.Time)

#Aggregating Data
mlm.data.dd <- aggregate( DD.TT.Time ~  subject+sessionIndex, data = dataset,
mean)
mlm.data.ud <- aggregate( UD.TT.Time ~  subject+sessionIndex, data = dataset,
mean)
mlm.data.hold <- aggregate( Hold.TT.Time ~  subject+sessionIndex, data = data
set, mean)
mlm.data.tt <- aggregate( TT.Time ~  subject+sessionIndex, data = dataset, me
an)
#Combining Model Data
mlm.data <- cbind(mlm.data.dd, mlm.data.ud[,3], mlm.data.hold[,3],mlm.data.tt
[,3])
colnames(mlm.data)[4] <- "UD.TT.Time"
colnames(mlm.data)[5] <- "Hold.TT.Time"
colnames(mlm.data)[6] <- "TT.Time"
#Checking Correlation Data
plot(mlm.data[,3:6])
```



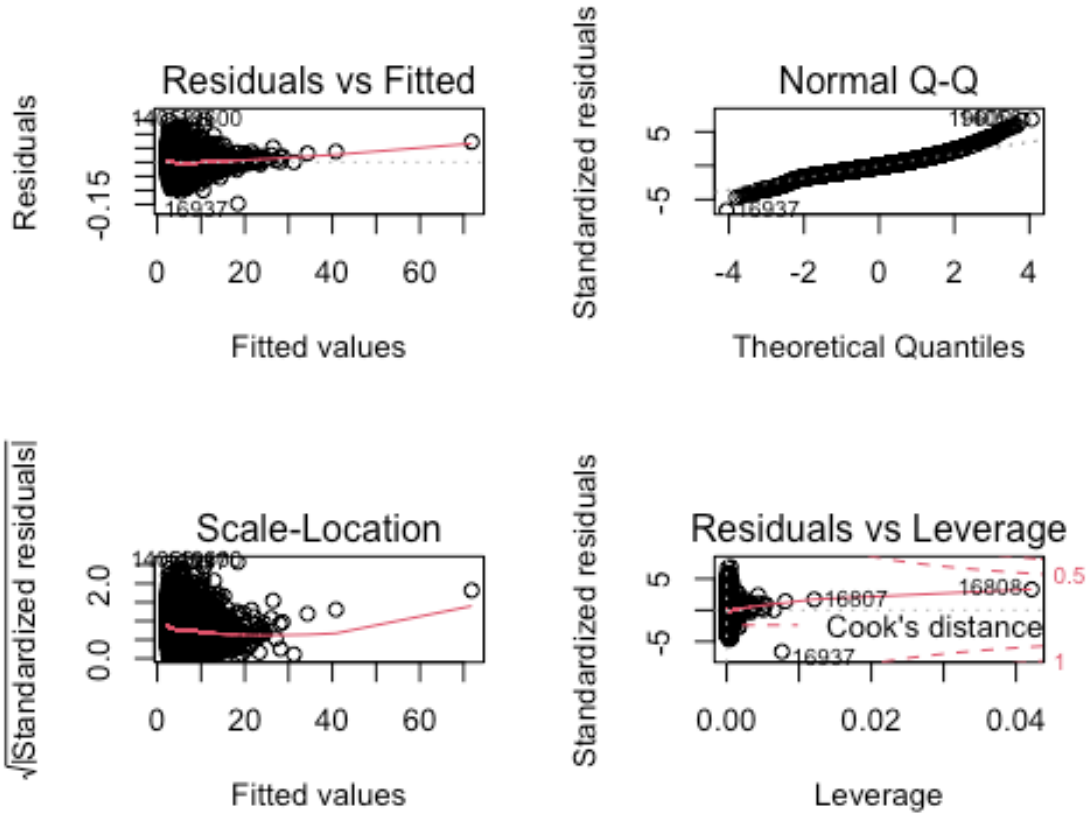

From the plot, we can see both DD time and UD time are highly correlated to TT.time. On the other hand , hold time shows weak corelation with TT.time. Hence we can drop hold time from the regression model for improved result.

```
#Regression
multiple.reg <- lm(TT.Time ~DD.TT.Time+ UD.TT.Time, data =)
kable(summary(multiple.reg)$coefficients,
      caption = "Multiple Regression Summary")
```

*Multiple Regression Summary*

| | Estimate | Std. Error | t value | Pr(>\|t\|) |
|---|---|---|---|---|
| (Intercept) | 0.0296641 | 0.0007816 | 37.95229 | 0 |
| DD.TT.Time | 2.0694658 | 0.0007561 | 2736.85899 | 0 |
| UD.TT.Time | -0.0720259 | 0.0007349 | -98.01399 | 0 |

```
#Diagonistics Plots
par(mfrow=c(2,2))
plot(multiple.reg)
```

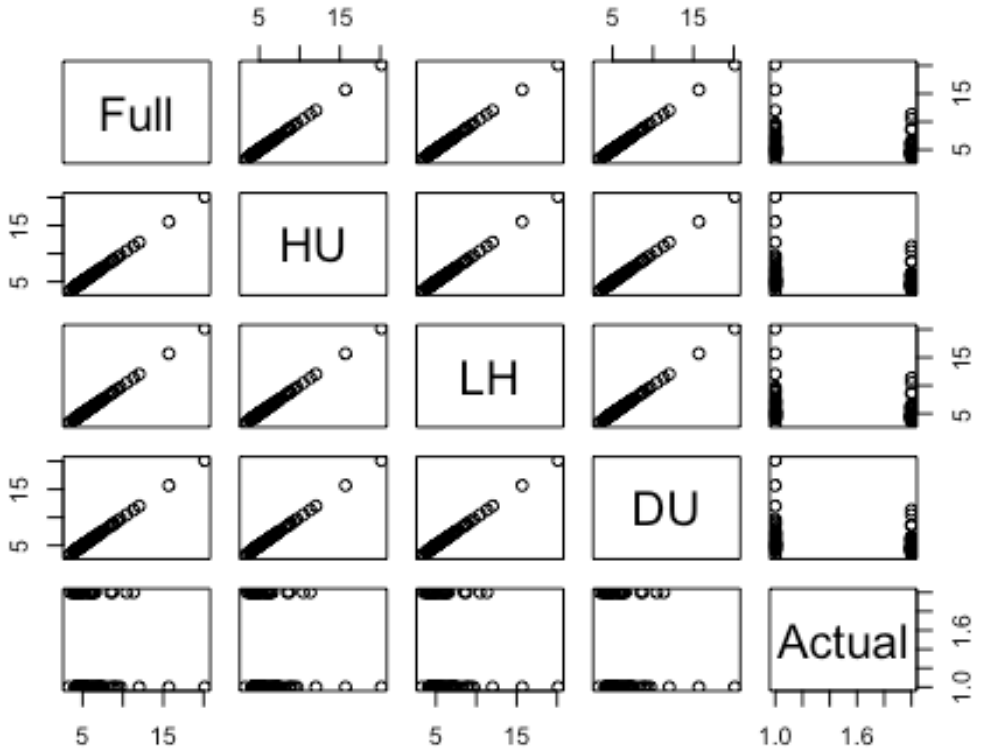



Here, it can be seen that p-value of the F-statistic is < 2.2e-16, which is highly significant. This means that, at least, one of the predictor variables is significantly related to the outcome variable. In our model, with DD time and UD time as predictor variables, the adjusted R2 = 0.99, meaning that "99% of the variance in the measure of TT.time can be predicted by DD time and UD time.

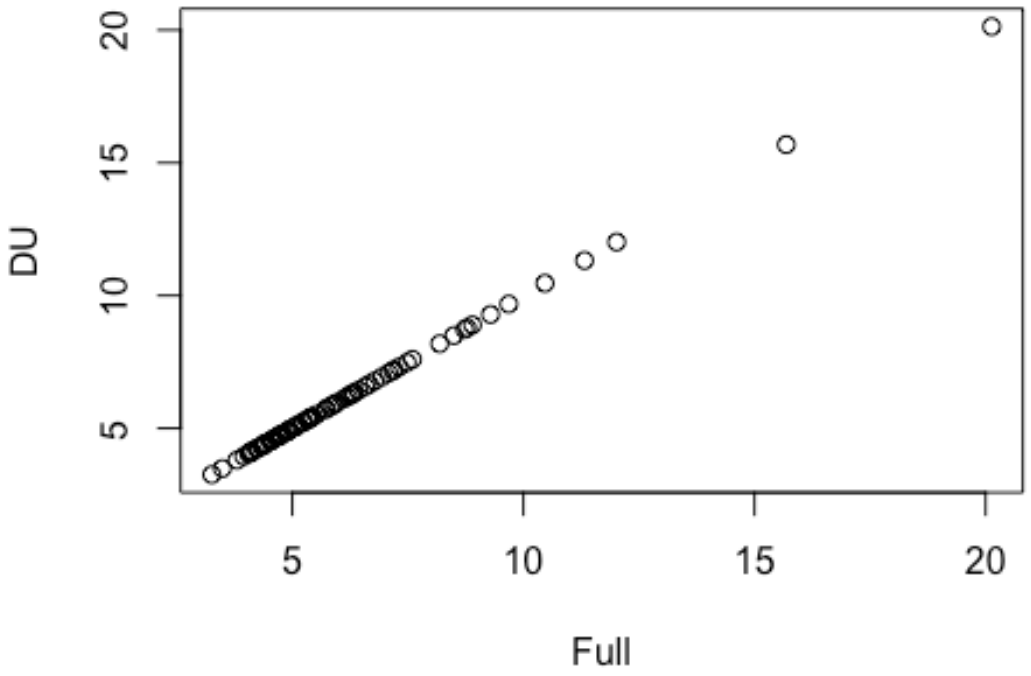
```
#confidence interval of the model coefficient can be extracted
confint(multiple.reg)

##                    2.5 %      97.5 %
## (Intercept)  0.02813209  0.03119615
## DD.TT.Time   2.06798372  2.07094794
## UD.TT.Time  -0.07346624 -0.07058550

#Residual Standard Error
al<- sigma(multiple.reg)/mean(multiple.reg$TT.Time)
al

## [1] NA
```

In our multiple regression example, the RSE is 0.01562 corresponding to 0.3% error rate.

```
preds = NULL

for(i in 1:84){
  lm.mod.full.i=
    lm(TT.Time~DD.TT.Time+UD.TT.Time+Hold.TT.Time, data = mlm.data[-i,], fami
ly = binomial())
  lm.mod.lh.i =
    lm(TT.Time~DD.TT.Time+UD.TT.Time, data = mlm.data[-i,], family = binomial
())
  lm.mod.lw.i =
    lm(TT.Time~DD.TT.Time+Hold.TT.Time, data = mlm.data[-i,], family = binomi
al())
  lm.mod.wh.i=
    lm(TT.Time~UD.TT.Time+Hold.TT.Time, data = mlm.data[-i,], family = binomi
al())

  preds =rbind(preds,
               c(predict(lm.mod.full.i, mlm.data[i,], type = 'response'),
                 predict(lm.mod.lh.i, mlm.data[i,], type = 'response'),
                 predict(lm.mod.lw.i, mlm.data[i,], type = 'response'),
                 predict(lm.mod.wh.i, mlm.data[i,], type = 'response'), mlm.d
ata[i,2]))
}

colnames(preds) = c("Full", "HU", "LH", "DU","Actual")
pairs(preds)
```




```
plot(preds[,c(1,4)])
abline(h= 0.5)
abline(v = 0.5)
```




The above plot shows that the full model and the model based of DD time and UD Time are effectively the same for the LOOCV.

---

## References


1. Xie, Yihui. 2015. Dynamic Documents with R and Knitr. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.name/knitr/.

2. 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, Estoril, Lisbon, Portugal, June 29-July 2, 2009. IEEE Computer Society Press, Los Alamitos, California, 2009. (pdf - http://www.cs.cmu.edu/~keystroke/KillourhyMaxion09.pdf)

3. T. Sing, O. Sander, N. Beerenwinkel, T. Lengauer. "ROCR: visualizing classifier performance in R," Bioinformatics 21(20):3940-3941 (2005). (link - https://ipa-tys.github.io/ROCR/)

4. Fawcett, T. 2004. "ROC Graphs: Notes and Practical Considerations for Researchers." In HPL-2003-4., 89–96. HP Labs, Palo Alto, CA.

5. Sing, Tobias, Niko Beerenwinkel, and Thomas Lengauer. 2004. “Learning Mixtures of Localized Rules by Maximizing the Area Under the Roc Curve.” In In et Al José Hernández-Orallo, Editor, 1st International Workshop on Roc Analysis in Artificial Intelligence, 89–96.

6. https://appliedmachinelearning.blog/2017/07/26/user-verification-based-on-keystroke-dynamics-python-code/

7. https://www.cs.cmu.edu/~keystroke/#sec1

8. https://towardsdatascience.com/keystroke-dynamics-analysis-and-prediction-part-1-eda-3fe2d25bac04

9. https://towardsdatascience.com/keystroke-dynamics-analysis-and-prediction-part-2-model-training-a13dc353b6e4

10. https://www.kaggle.com/ashusrivastava/kda-on-benchmark

11. Plank, Barbara. “Keystroke dynamics as signal for shallow syntactic parsing.” arXiv preprint arXiv:1610.03321 (2016).

12. Montalvao, Jugurta, Carlos Augusto S. Almeida, and Eduardo O. Freire. “Equalization of keystroke timing histograms for improved identification performance.” 2006 International telecommunications symposium. IEEE, 2006.

13. https://pandas.pydata.org/docs/getting_started/comparison/comparison_with_r.html

14. https://towardsdatascience.com/jypyter-notebook-shortcuts-bf0101a98330

15. https://github.com/kevinmgamboa/keystroke_dynamics/blob/master/keystroke_dynamics-Benchmark_kd.main_v1.ipynb

16. https://ipa-tys.github.io/ROCR/articles/ROCR.html#references-1

17. https://www.cs.cmu.edu/~keystroke/#sec1

18. https://appliedmachinelearning.blog/2017/07/26/user-verification-based-on-keystroke-dynamics-python-code/

19. https://appliedmachinelearning.blog/2017/07/26/user-verification-based-on-keystroke-dynamics-python-code/

20. https://github.com/RoyMaxion/RoyMaxion.github.io/blob/master/projects/keystroke-benchmark/evaluation-script.R?fbclid=IwAR3knWKKaG89_PmNF2zDlGzQZghf8ZOts_EZy9qeVA-uik0GlnrJqMXRhMI