Statistics Coursework

Rafael Castillo

06/03/2021

Importing dataset and libraries

```
setwd("D:/OneDrive/Desktop/MASTERS/Statistics CW/Datasets")
wsites = read.csv("wsites.csv", stringsAsFactors = T)
passwords = read.csv("passwords.csv", stringsAsFactors = T)
library(ggplot2)
library(summarytools)
```

```
## Warning: package 'summarytools' was built under R version 4.0.4
```

```
## Registered S3 method overwritten by 'pryr':
##   method      from
##   print.bytes Rcpp
```

```
## For best results, restart R session and update pander using devtools:: or remotes::install_github('rapporter/p
ander')
```

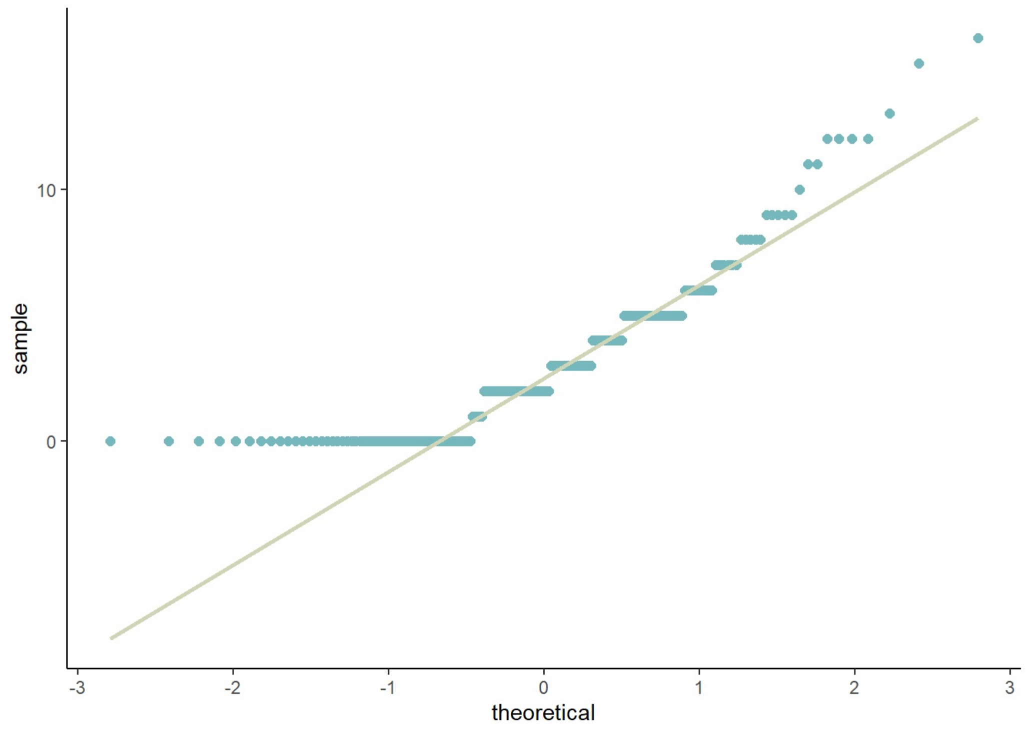
SECTION 1: VISUALISATIONS – WSITES DATASET

Generate the following charts/diagrams and, hence, enter the requested information in the spaces provided.

Q-Q Plot

1. A plot showing the distribution of values for REMOTE_IPS. Does it show a normal distribution? State the most frequent value and the 2nd most frequent value.

```
# qqplot showcasing the distribution of remote ips
RIqqplot <- ggplot(wsites, aes(sample = REMOTE_IPS)) + theme_classic()
RIqqplot <- RIqqplot + stat_qq(col="#75B9BE", size=2)
RIqqplot <- RIqqplot + stat_qq_line(size=1)
RIqqplot$layers[[2]]$aes_params$colour <- "#D0D6B5"
RIqqplot
```



The qqplot shows that the majority of remote IPS instances are not aligned with the expected distribution. Some of the instances do align with the theoretical lines but not enough to be considered normal. We can also see a right skew since most of the instances are found close to the minimum values; we can conclude that the distribution of remote IPS is not normal.

Shapiro Wilk test

```
# The Shapiro test shows if the data is normally distributed or not.
result = shapiro.test(wsites$REMOTE_IPS)
result
```

```
##
##  Shapiro-Wilk normality test
##
## data:  wsites$REMOTE_IPS
## W = 0.85411, p-value = 1.624e-12
```

```
# Transform the format of the P-Value to a decimal value
pvalue = format(result$p.value, scientific=FALSE)
paste(c('P-Value =', pvalue), collapse = " ")
```

```
## [1] "P-Value = 0.000000000001623923"
```

The p-value is > 0.05 so it is reasonable to assume that the distribution is not normal

Frequent values

```
# The following code shows the first and second most frequent values in remote IPS.
RI2MostFreqValues = tail(names(sort(table(wsites$REMOTE_IPS))), 2)
RI2MostFreqValues
```

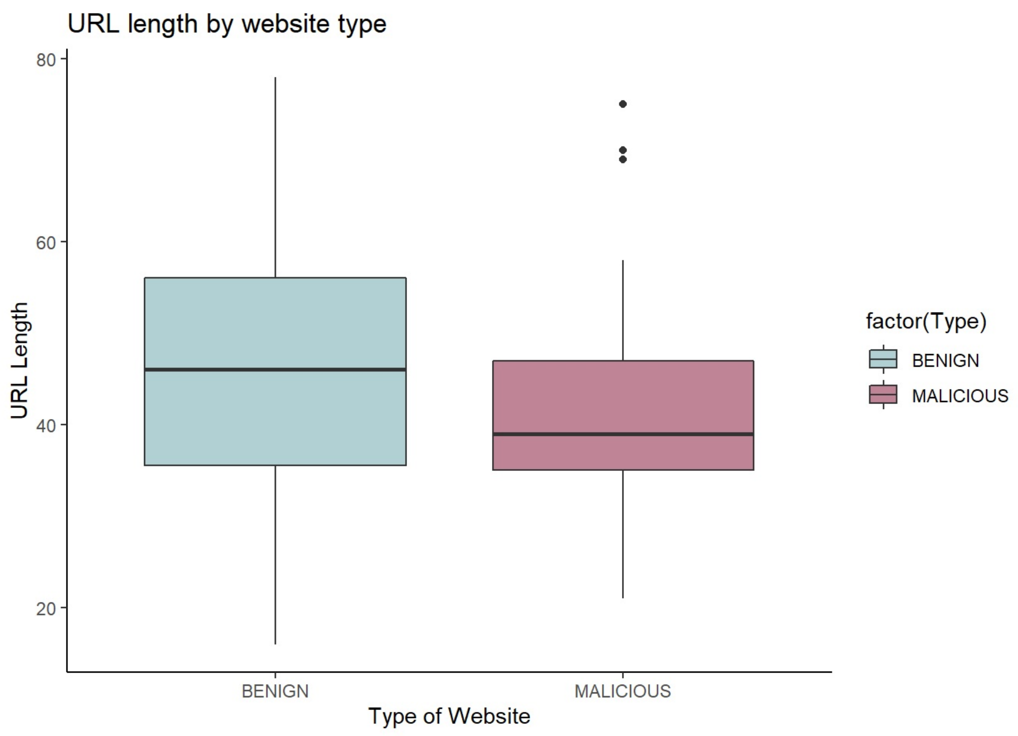
```
## [1] "2" "0"
```

The most frequent values in remote IPS is 0 and the 2nd most frequent is 2. This makes sens since most systems don’t make the use honeypots at all and if they do is usually just a few in this case we can see that the second most is 2.

Box plot for URL length and Type

2. Create a Boxplot of URL_LENGTH by Type of website. Are there any outliers? What is the approximate value of the 26th highest value for each Type?

```
# Boxplot showcasing the distribution of url length by type
barPlotColors = c("#B0D0D3", "#C08497")
ggplot(wsites, aes(factor(Type), URL_LENGTH, fill = factor(Type))) + geom_boxplot()+
ggtitle('URL length by website type') + xlab('Type of Website') + ylab('URL Length') + scale_fill_manual(values=b
arPlotColors) + theme_classic()
```



As it can be seen on our box plot “Malicious” websites seem to have a few (3) outliers when it comes to url length.

26th Highest values

```
# The following code creates a subset of two variables
typeNURLLength <- wsites[, c("URL_LENGTH", "Type")]
benignSet = subset(typeNURLLength, Type == 'BENIGN')
maliciousSet = subset(typeNURLLength, Type == 'MALICIOUS')
# The following code orders the subsets in descending order
benignSet = benignSet[order(-benignSet$URL_LENGTH),]
maliciousSet = maliciousSet[order(-maliciousSet$URL_LENGTH),]
# The following code returns the 26th value from within the subsets
paste(c('26th highest value for URLs in Benign websites is:', benignSet[26,1]), collapse = " ")
```

```
## [1] "26th highest value for URLs in Benign websites is: 61"
```

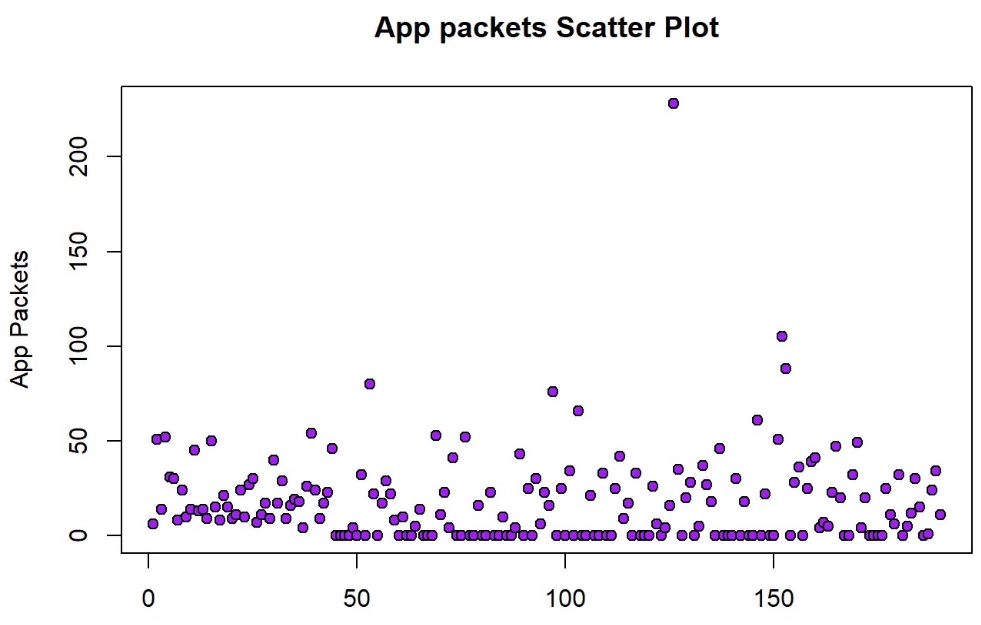
```
paste(c('26th highest value for URLs in Malicious websites is:', maliciousSet[26,1]), collapse = " ")
```

```
## [1] "26th highest value for URLs in Malicious websites is: 36"
```

Scatter plot for app packets

3. A scatterplot of number of app packets. Comment on any interesting facts.

```
# A scatterplot showing the distribution of APP_PACKETS on the Y axes
plot(wsites$APP_PACKETS, xlab = "", ylab="App Packets", main = "App packets Scatter Plot", bg = "purple", pch=21)
```



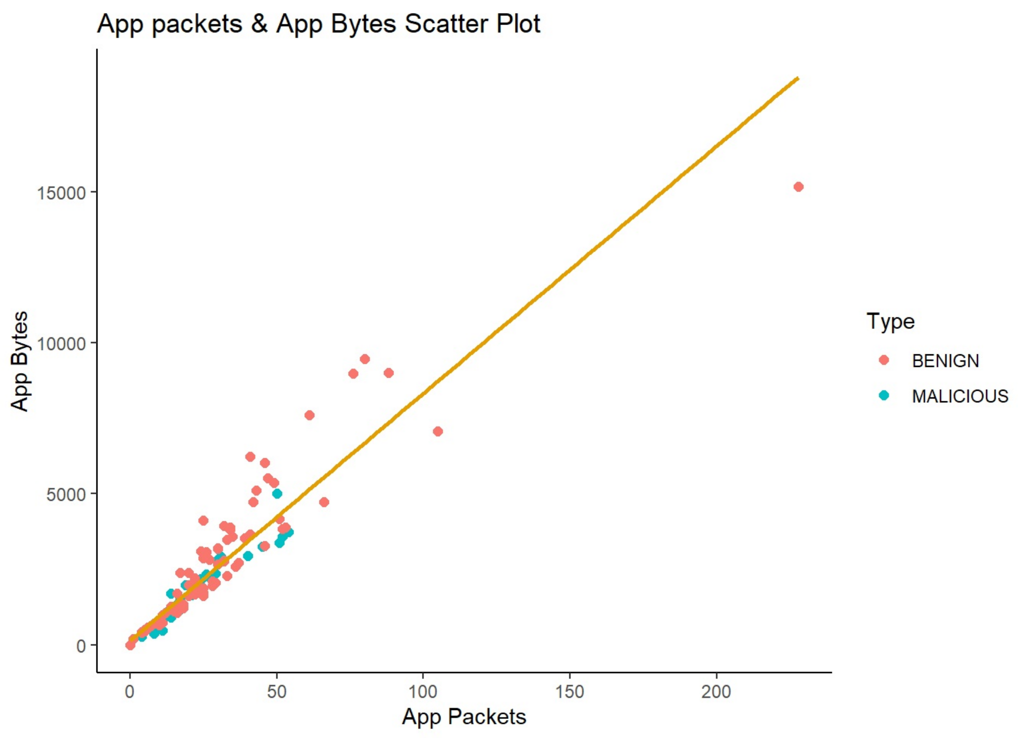
```
#Optional Code for dot plot instead of scatterplot
# p <- ggplot(wsites, aes(x= APP_PACKETS)) +
#   geom_dotplot(col="red", fill="red" , binwidth=0.05) +
#   labs(x="Use level", y="") +
#   theme_classic()
# p
```

The distribution of values does not look normal. The distribution looks positively skewed as the majority of the values are located close to the minimum value and a decrease can be seen towards the maximum value. It is safe to say that the majority of app packets are below 50.

Scatter plot for app packets and bytes

```
# qqplot showcasing the distribution of remote App Bytes vs App Packages by type
ggplot(wsites, aes(x=APP_PACKETS, y= APP_BYTES)) + geom_point(aes(col=Type), size=3, shape=20)+
ggtitle('App packets & App Bytes Scatter Plot') + xlab('App Packets') + ylab('App Bytes')+
geom_smooth(method=lm, se=FALSE, col='#E69F00') + theme_classic()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

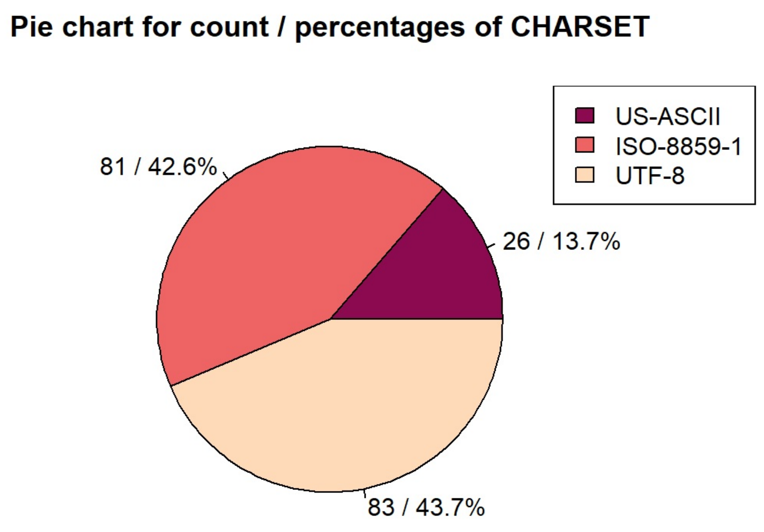


We can see that our scatter plot has a strong positive linear relationship for both types. This means that there is a correlation between both quantitative variables, it makes sense as every packet transfer bytes of data, meaning the more packages transferred the more bytes transferred as well. We can also see that Benign websites are a bit more disperse as well as extends to bigger byte and packets figures.

Pie Chart character set

5. A pie chart of CHARSET showing both counts and percentages. Comment on the plot.

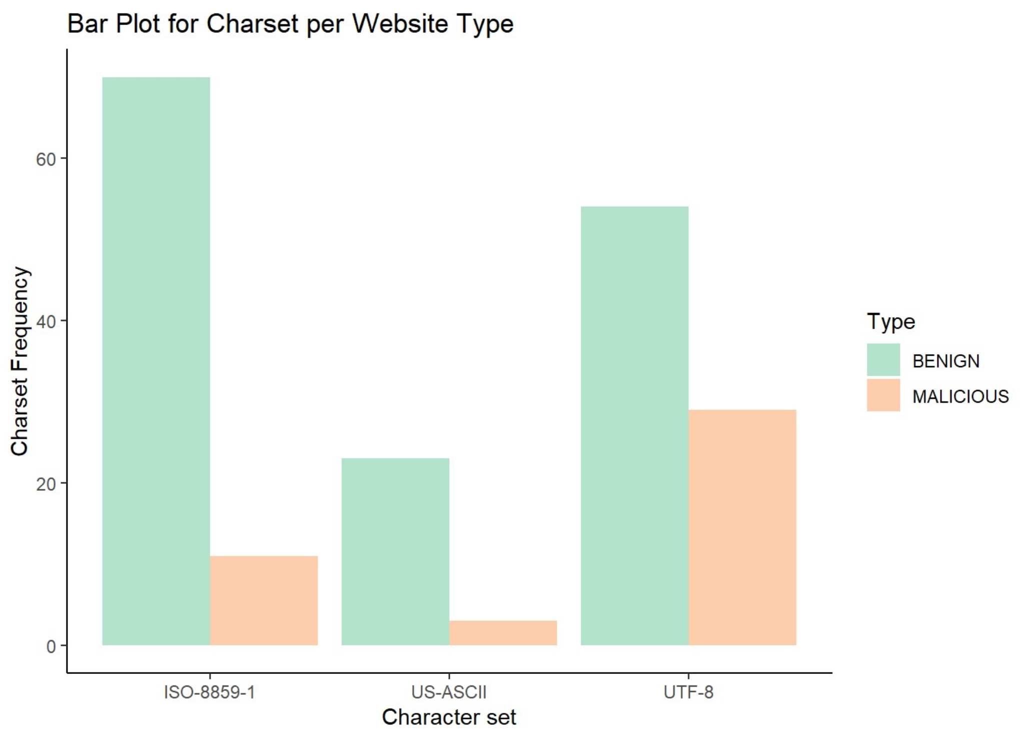
```
# The following code collects the data of CHARSET from wsites for a pie chart
charsetTable = table(wsites$CHARSET)
charsetTable = sort(charsetTable)
count = charsetTable

# prepares the format to of the variables to be introduced in the pie chart
piepercent <- round(100*charsetTable/sum(charsetTable), 1)
countNPercent = paste(count, "/", piepercent, sep=" ")
countNPercent = paste(countNPercent, "%", sep="")
pieChartColors = c("deeppink4", "indianred2", "peachpuff")

# The following code produces a pie chart
pie(charsetTable,  labels = countNPercent, main= "Pie chart for count / percentages of CHARSET", col = pieChartCo
lors)
legend("topright", c("US-ASCII","ISO-8859-1","UTF-8"), fill = pieChartColors)
```



It seems as if UTF-8 and ISO-8859-1 are the most used character sets on this websites differing only by 2 instances and with a total of 86% of the websites dataset. We can conclude that US-ASCII is the least used character set for the websites on this dataset with only 13% of the instances in the dataset.

Bar Chart for Character set

```
# qqplot showcasing the frequency of charset by type
ggplot(wsites, aes(x=CHARSET, fill=Type)) + geom_bar(stat = "count", position=position_dodge()) + labs(y="Charset Frequency", x="Character set", title="Bar Plot for Charset per Website Type") + scale_fill_brewer(palette = "Pastel2") + theme_classic()
```



We can see by the results that most websites with character set ISO-8859-1 and US-ASCII are of the benign type. In contrast UTF-8 shows the biggest increase in malicious websites. We can conclude that UTF-8 websites have a bigger chance of being malicious.

SECTION 2: TABLES & MEASURES – WSITES DATASET

Generate the following tables/descriptive statistics and, hence, enter the requested information in the spaces provided.

1. A one-way table of frequencies (counts) and cumulative frequencies for WHOIS_COUNTRY. What country appears most times?

One-Way Table of Frequencies

```
# The following binds 2 tables into one way table show casing frequency and cumulative frequency
cbind(Freq=table(wsites$WHOIS_COUNTRY), CumulFreq=cumsum(table(wsites$WHOIS_COUNTRY)))
```

```
##      Freq CumulFreq
## AU      3         3
## BE      1         4
## CA      7        11
## CZ      3        14
## ES      1        15
## FR      1        16
## GB      5        21
## IL      1        22
## None   51        73
## PA      2        75
## ru      1        76
## RU      1        77
## se      1        78
## SE      1        79
## TR      1        80
## UA      1        81
## UG      1        82
## UK      1        83
## US    107       190
```

We can see that the country that appears most times is United States (US)

Two-Way Table

2. A two-way table of Type (in rows) and CHARSET (in columns) showing frequencies. What is the number of benign websites using UTF-8?

```
# The following creates a two way table showcasing type and frequency of type and charset
ctable(x = wsites$Type, y = wsites$CHARSET, prop = "n", totals = FALSE )
```

```
## Cross-Tabulation
## Type * CHARSET
## Data Frame: wsites
##
## ----------- --------- ------------ ---------- -------
##               CHARSET   ISO-8859-1   US-ASCII   UTF-8
##        Type
##      BENIGN                     70         23      54
##   MALICIOUS                     11          3      29
## ----------- --------- ------------ ---------- -------
```

From the result we can see that the number of benign websites using UTF-8 is 54

Summary statistics

3. Obtain a summary of the data statistics for SERVER and CONTENT_LENGTH. Make at least one interesting observation on each for each of the attributes.

```
# Returns a description of the variable CONTENT_LENGTH
descr(wsites$CONTENT_LENGTH)
```

```
## Descriptive Statistics
## wsites$CONTENT_LENGTH
## N: 190
##
##                     CONTENT_LENGTH
## ----------------- ----------------
##              Mean          8897.10
##           Std.Dev         42608.12
##               Min            34.00
##                Q1           324.00
##            Median           653.50
##                Q3          3400.50
##               Max        435494.00
##               MAD           630.10
##               IQR          3039.25
##                CV             4.79
##          Skewness             9.04
##       SE.Skewness             0.23
##          Kurtosis            86.83
##           N.Valid           112.00
##         Pct.Valid            58.95
```

We can see there is a skewness in the distribution of length of websites. The median of each quartile seems to grow exponentially from 34 at min to 435494 at max.

```
# Returns a description of the variable SERVER
summary(wsites$SERVER)
```

```
##                Apache      cloudflare-nginx Microsoft-HTTPAPI/2.0
##                    58                    14                    25
##                 nginx          nginx/1.12.0                  None
##                    49                    12                    32
```

The servers don’t exceed more than 58 observations. There are 32 observation which belong to unknown servers.

SECTION 3: SIGNIFICANCE TESTS - PASSWORDS DATASET

Confidence Interval

1. Determine a 99% confidence interval for the mean value of Length (the length after training was given).

```
# Obtaining the T value.
# number of instances of data
n <- length(passwords$LengthAfter)

# Confidence and significance
confidence <- 0.99
significance <- 1- confidence
# two-tailed so half
halfSignificance <- significance/2
a <- 1 - halfSignificance

# degrees of freedom
dfLengthAfter <- n - 1

# finding t-value
# function qt returns required value for t
tValue <- qt(a, dfLengthAfter)

# Obtaining standard deviation
sdLengthAfter <- sd(passwords$LengthAfter)

# Obtaining mean value
meanLengthAfter <- mean(passwords$LengthAfter)

# Obtaining margin of error
error <- tValue * sdLengthAfter/sqrt(n)

# Lower and upper limits
lowerlimit <- meanLengthAfter - error
upperlimit <- meanLengthAfter + error

# confidence interval
CI <- paste("Confidence interval is (", lowerlimit, ", ",  upperlimit, ")")
CI
```

```
## [1] "Confidence interval is ( 8.74139549988899 ,  10.2647583462649 )"
```

Parametric test (Increase)

2. Use a parametric test for matched pairs to test for evidence in increase in length of passwords after training. Use a significance level of 0.05.

Ho: NULL hypothesis - there is no difference hence no increase of passwords after training.

H1: Alternative hypothesis – there is a increase in length of passwords after training.

```
# The following code performs a parametric t test to test for increase of x axes
t.test(x=passwords$LengthAfter, y=passwords$LengthBefore, alternative = "greater", paired=T, mu=0, conf.level=0.9
5)
```

```
##
##  Paired t-test
##
## data:  passwords$LengthAfter and passwords$LengthBefore
## t = -0.10981, df = 38, p-value = 0.5434
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.1844969        Inf
## sample estimates:
## mean of the differences
##             -0.01128205
```

The p-value(0.5434) > 0.05 so do not reject NULL hypothesis, hence there is no enough evidence of increase in password length after training.

Non-Prametric test (Difference)

3. Use a non-parametric test for matched pairs to test for evidence of difference in lengths (before and after training values).

Ho: NULL hypothesis - there is no difference between before and after training.

H1: Alternative hypothesis – there is a difference in length of passwords before and after training.

```
# The following code performs a non-parametric wilcox test to test for difference.
wilcox.test(x=passwords$LengthBefore, y = passwords$LengthAfter, alternative = "two.sided", mu = 0, paired = T, e
xact = F,correct = T, conf.int = F, conf.level = 0.95)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  passwords$LengthBefore and passwords$LengthAfter
## V = 401.5, p-value = 0.878
## alternative hypothesis: true location shift is not equal to 0
```

The p-value(0.878) > 0.05 so do not reject NULL hypothesis. This means there is no significant difference before and after training.

Non-parametric test (Increase)

4. Use a non-parametric test for matched pairs to test for evidence of increase in lengths (before and after training values).

Ho: NULL hypothesis - there is no difference hence no increase between before and after.

H1: Alternative hypothesis – there is a increase in length of passwords after training.

```
# The following code performs a non-parametric wilcox test to test for increase of x axes
wilcox.test(x=passwords$LengthAfter, y = passwords$LengthBefore, alternative = "greater", mu = 0, paired = T, exa
ct = F,correct = T, conf.int = F, conf.level = 0.95)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  passwords$LengthAfter and passwords$LengthBefore
## V = 378.5, p-value = 0.5665
## alternative hypothesis: true location shift is greater than 0
```

The p-value(0.5665) > 0.05 so do not reject NULL hypothesis, hence there is no enough evidence of increase in password length after training.

Normality Tests

5. Undertake normality tests for the 2 length attributes.

Shapiro-Wilk test of normality

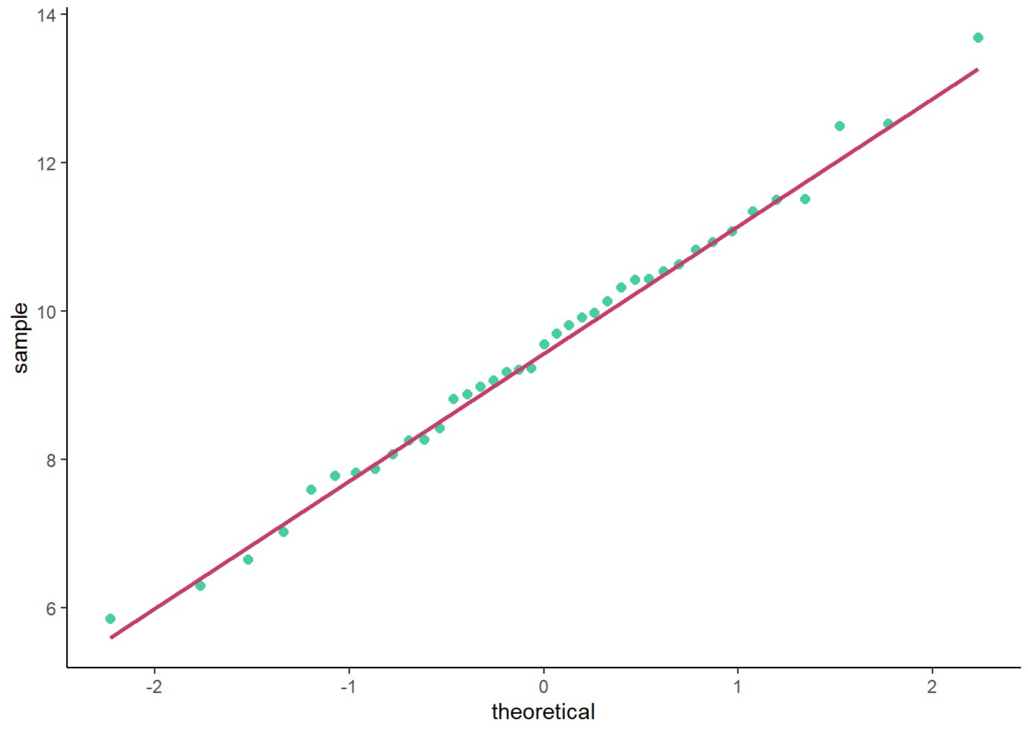
```
# The following code performs a shapiro test for normality
shapiro.test(passwords$LengthBefore)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  passwords$LengthBefore
## W = 0.98791, p-value = 0.9444
```

The p-value is > 0.05 so it is reasonable to assume that the distribution is normal.

Q-Q Plot for normality

```
# qqplot showcasing the distribution of LengthAfter
ntqqplot <- ggplot(passwords, aes(sample = LengthAfter)) + theme_classic()
ntqqplot <- ntqqplot + stat_qq(col="#43D19F", size=2)
ntqqplot <- ntqqplot + stat_qq_line(size=1)
ntqqplot$layers[[2]]$aes_params$colour <- "#CF3A6E"
ntqqplot
```



The points are close to the line. It is reasonable to assume that the distribution is normal.

ANOVA Test

6. Perform a one-way analysis of variance with Tukey's multiple comparisons to compare the mean LengthAfter of the four Department categories. (Use a significance level of 0.05 for the ANOVA test, and a 95% confidence level for Tukey's comparisons. Check the assumptions required for ANOVA and comment in the validity of the test.

Cheking assumptions

```
# The follwing code cheks for the assumptions that each departamental group is normally distributed.
depA = passwords$LengthAfter[passwords$Department == "A"]
shapiro.test(depA)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  depA
## W = 0.96516, p-value = 0.8063
```

```
depB = passwords$LengthAfter[passwords$Department == "B"]
shapiro.test(depB)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  depB
## W = 0.89931, p-value = 0.248
```

```
depC = passwords$LengthAfter[passwords$Department == "C"]
shapiro.test(depC)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  depC
## W = 0.97057, p-value = 0.8922
```

```
depD = passwords$LengthAfter[passwords$Department == "D"]
shapiro.test(depD)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  depD
## W = 0.95478, p-value = 0.7713
```

The p-value of all departments is > 0.05 so it is reasonable to assume that the distribution of all of them is normal.

Applying the one-way ANOVA test.

H0: There is no difference in mean values i.e. μA = μB = μC = μD.

H1: at least two means are different.

```
# creating mini data frames, one for each algorithm,
# each one contains one column with the name of the algorithm and one column with the time
departmentA <- data.frame(department= "A", length = depA)
departmentB <- data.frame(department= "B", length = depB)
departmentC <- data.frame(department= "C", length = depC)
departmentD <- data.frame(department= "D", length = depD)

# putting the 4 mini data frames together
departments = rbind(departmentA, departmentB, departmentC, departmentD)

# Creating a one-way anova test
anova <- aov( length ~ department, data = departments)
summary(anova)
```

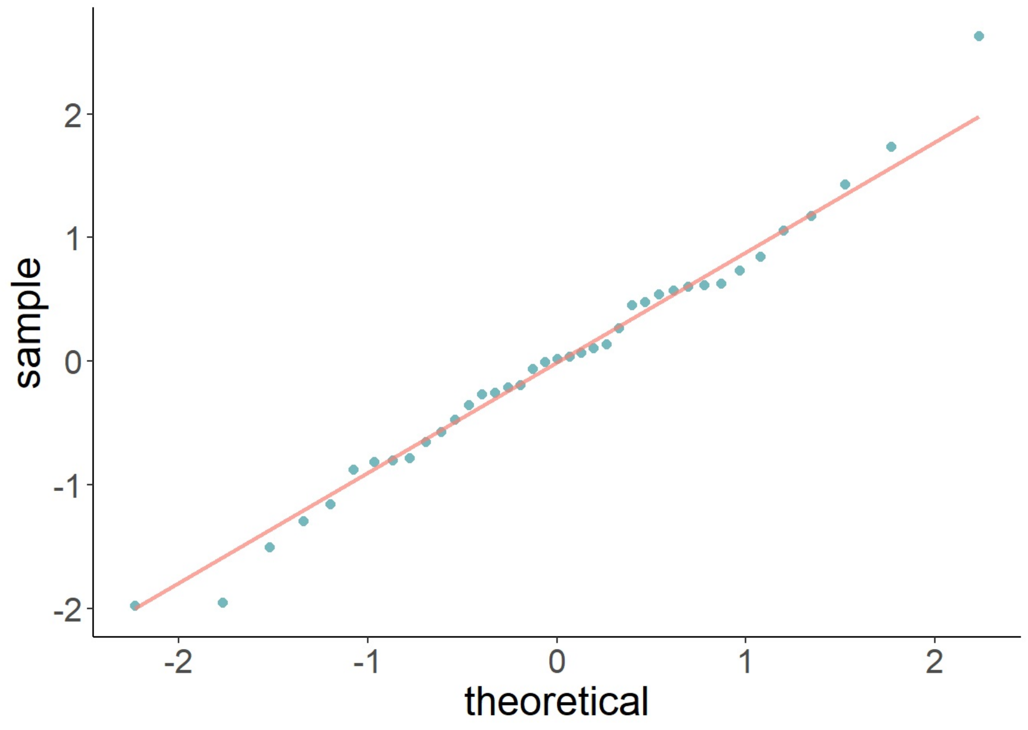
```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## department   3  81.93   27.31    27.3 2.78e-09 ***
## Residuals   35  35.01    1.00
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Tranforming P-value to decimal value
pvalue = format(2.78e-09, scientific=FALSE)
paste("The p-Value:", pvalue, "Is less than 0.05")
```

```
## [1] "The p-Value: 0.00000000278 Is less than 0.05"
```

The p-value is less than the significance level 0.05, we can conclude that there are significant differences between the groups highlighted with "***" in the model summary.

```
# Construct a data frame with the rediduals.
anovaFrame <- data.frame(residuals = anova$residuals )

# qqplot showcasing the distribution of the residuals
aqqplot <- ggplot(anovaFrame, aes(sample = residuals))
aqqplot <- aqqplot + stat_qq(size=2, col="#75B9BE") + stat_qq_line( alpha = 0.7, color = "salmon", size=1)
aqqplot <- aqqplot +  theme_classic()
aqqplot <- aqqplot + theme(text = element_text(size = 20))
aqqplot
```



The points are close to the theoretical line. It is reasonable to assume the distribution is normal.

```
# Shapiro test showcasing the distribution of the residuals
shapiro.test(anova$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova$residuals
## W = 0.98502, p-value = 0.8733
```

The p-value (0.8733) is > 0.05 so it is reasonable to assume the distribution is normal.

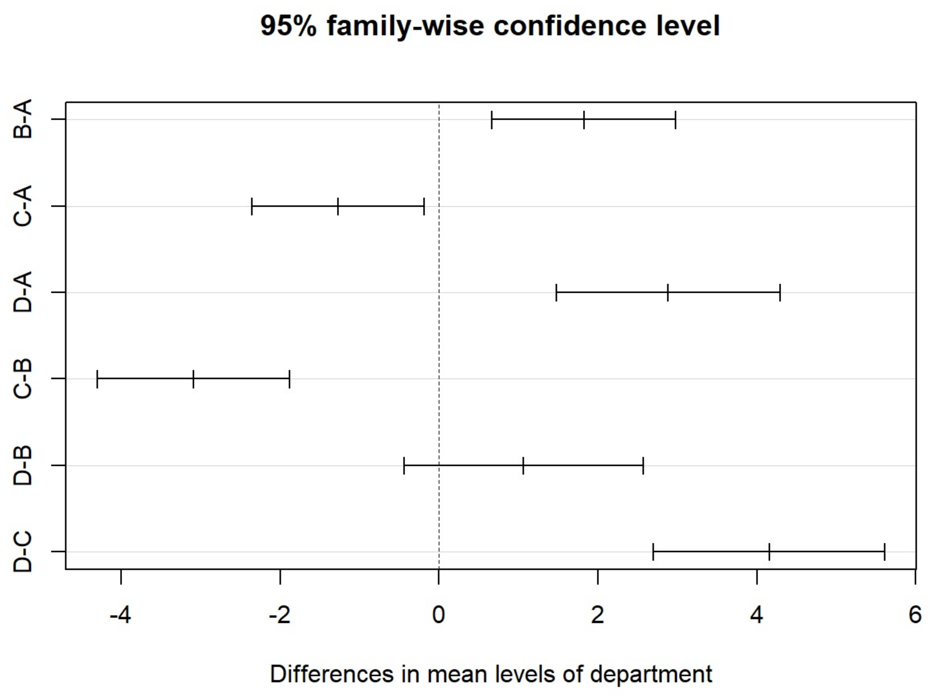
Tukey test

```
# Tukey test showcasing multiple comparisons to compare the mean LengthAfter
t <- TukeyHSD(anova)
t
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = length ~ department, data = departments)
##
## $department
##          diff        lwr        upr     p adj
## B-A  1.820794  0.6684335  2.9731538 0.0008058
## C-A -1.270519 -2.3572434 -0.1837956 0.0166496
## D-A  2.884571  1.4793740  4.2897689 0.0000182
## C-B -3.091313 -4.3036044 -1.8790219 0.0000003
## D-B  1.063778 -0.4406361  2.5681917 0.2436599
## D-C  4.155091  2.7003414  5.6098404 0.0000000
```

All pairs show a statistical significant difference in their means with the exception of group D-B.

```
# plot the tukey test
plot(t)
```



We can see that only D-B confidence interval contains zero. All other pairs do not contain 0 hence their means are significantly different.

```
# Creates dataframe with the values from the exercise
difference = c(7,7,7,10,10,10,13,13,13)
power = c(0.85,0.90,0.95,0.85,0.90,0.95,0.85,0.90,0.95)
combDiff = data.frame(difference, power)
names(combDiff) = c("Difference", "Power")

# Conducts a t test for each of the rows in combDiff
a <- power.t.test(power=combDiff$Power[1],delta=combDiff$Difference[1],sd=15,sig.level=0.01,type="two.sample")
b <- power.t.test(power=combDiff$Power[2],delta=combDiff$Difference[2],sd=15,sig.level=0.01,type="two.sample")
c <- power.t.test(power=combDiff$Power[3],delta=combDiff$Difference[3],sd=15,sig.level=0.01,type="two.sample")
d <- power.t.test(power=combDiff$Power[4],delta=combDiff$Difference[4],sd=15,sig.level=0.01,type="two.sample")
e <- power.t.test(power=combDiff$Power[5],delta=combDiff$Difference[5],sd=15,sig.level=0.01,type="two.sample")
f <- power.t.test(power=combDiff$Power[6],delta=combDiff$Difference[6],sd=15,sig.level=0.01,type="two.sample")
g <- power.t.test(power=combDiff$Power[7],delta=combDiff$Difference[7],sd=15,sig.level=0.01,type="two.sample")
h <- power.t.test(power=combDiff$Power[8],delta=combDiff$Difference[8],sd=15,sig.level=0.01,type="two.sample")
i <- power.t.test(power=combDiff$Power[9],delta=combDiff$Difference[9],sd=15,sig.level=0.01,type="two.sample")

# Prints results of t tests
cat(
paste("",
"n = ", ceiling(a$n), "units = ", a$delta, "power = ", a$power, "\n",
"n = ", ceiling(b$n), "units = ", b$delta, "power = ", b$power, "\n",
"n = ", ceiling(c$n), "units = ", c$delta, "power = ", c$power, "\n",
"n = ", ceiling(d$n), "units = ", d$delta, "power = ", d$power, "\n",
"n = ", ceiling(e$n), "units = ", e$delta, "power = ", e$power, "\n",
"n = ", ceiling(f$n), "units = ", f$delta, "power = ", f$power, "\n",
"n = ", ceiling(g$n), "units = ", g$delta, "power = ", g$power, "\n",
"n = ", ceiling(h$n), "units = ", h$delta, "power = ", h$power, "\n",
"n = ", ceiling(i$n), "units = ", i$delta, "power = ", i$power,
sep = "\t"))
```

```
##   n =     122 units =     7   power =     0.85
##   n =     139 units =     7   power =     0.9
##   n =     166 units =     7   power =     0.95
##   n =     61  units =     10  power =     0.85
##   n =     69  units =     10  power =     0.9
##   n =     82  units =     10  power =     0.95
##   n =     37  units =     13  power =     0.85
##   n =     42  units =     13  power =     0.9
##   n =     50  units =     13  power =     0.95
```

Random Instances

2. Select 20 instances at random from a set of 150, numbered from 1 to 150. Present them in ascending order. Ensure the reproducibility of your results.

```r
# Ensures reproducibility of experiment
set.seed(36)

# Creates a set from 1 to 150
instances = c(1:150)

# Selects 20 instances at random from the set
instances20 = sample(instances, 20)

#Presents the results in ascending order
sort(instances20, decreasing = FALSE)
```

```
##  [1]  21  24  25  26  43  54  58  60  61  63  69  76  77  85  95  97  99 101 104
## [20] 128
```

```r
#Optional Code seen in class
# set.seed(123)
# myRandoms <- floor(runif(20, min=1, max=156))
# sort(myRandoms, decreasing = FALSE)
```

Data Collection Procedure

1. I would have a clear objective, e.g.(Determine if the menu of a restaurant increases in profit by changing to a new menu)
2. Be mindful of the type of experimental design to apply, e.g.(Parallel, exploratory sequential or explanatory sequential)
3. List the qualitative and quantitative data needed to be collected, e.g.(Profit, number or name of plates ordered, day of the week, etc.)
4. Would select an independent variable and the dependant variables to be studied, e.g.(before menu and after menu + all other qualitative data). I would also be aware of any confounding variable that needed to be suppressed.
5. Determine the sample size needed to have significant experiments and the risk level to be tolerated, significance level, e.g.(0.05) and power fo test, e.g.(0.95)
6. Make sure the data is being collected randomly and fair to compare by like for like basis.
7. Balance between the experiment’s accuracy and the cost and time it takes to collect the data.
8. Make sure the data being collected is varied, e.g.(Not just weekends profit but all week, month and year-round)
9. Design the experimental design considering the following concepts (Replication, Randomization, Analysis method, Results interpretation) as to be more mindful of the data being collected.