HO CHI MINH UNIVERSITY OF TECHNOLOGY
FALCUTY OF COMPUTER SCIENCE AND ENGINERRING

Probability and Statistics

Assignment

Project 2 – Topic 5

Advisors: Phan Thị Hường				
Student's Name	**Student's ID**	**Class**	**Work**	**Percentage of work**
Huỳnh Kim Hưng	1952745	CC07	Summarizing the activities & Make report	100%
Trịnh Sơn Lâm	1852502	CC07	Activity 1	100%
Phan Tuấn Khải	1952780	CC07	Activity 2	100%
Lưu Trịnh Lâm	1952315	CC07	Activity 1	100%
Trịnh Mạnh Hùng	1952740	CC07	Activity 2	100%

Contents

1. Activity 1

1.1. Data visualization – Description statistics

Chicken farming is a multi-billion dollar industry, and any methods that increase the growth rate of young chicks can reduce consumer costs while increasing company profits, possibly by millions of dollars. An experiment was conducted to measure and compare the effectiveness of various feed supplements on the growth rate of chickens. Newly hatched chicks were randomly allocated into six groups, and each group was given a different feed supplement.

	weight	feed
1	179	horsebean
2	160	horsebean
3	136	horsebean
4	227	horsebean
5	NA	horsebean
6	168	horsebean
7	108	horsebean
8	124	horsebean
9	143	horsebean
10	140	horsebean
11	309	linseed
12	229	linseed
13	181	linseed
14	141	linseed
15	260	linseed
16	203	linseed
17	NA	linseed
18	169	linseed
19	213	linseed
20	257	linseed
21	244	linseed
22	271	linseed
23	243	soybean
24	230	soybean
25	248	soybean
26	327	soybean
27	329	soybean
28	250	soybean
29	193	soybean
30	271	soybean
31	316	soybean
32	267	soybean
33	199	soybean
34	171	soybean
35	158	soybean
36	248	soybean
37	423	sunflower
38	340	sunflower
39	392	sunflower
40	339	sunflower
41	341	sunflower
42	226	sunflower
43	320	sunflower
44	295	sunflower
45	334	sunflower
46	322	sunflower
47	297	sunflower
48	318	sunflower
49	325	meatmeal
50	257	meatmeal
51	303	meatmeal
52	315	meatmeal
53	380	meatmeal
54	153	meatmeal
55	263	meatmeal
56	242	meatmeal
57	206	meatmeal
58	344	meatmeal
59	258	meatmeal
60	368	casein
61	390	casein
62	379	casein
63	260	casein
64	404	casein
65	318	casein
66	352	casein
67	359	casein
68	216	casein
69	222	casein
70	283	casein
71	332	casein

- Using one way ANOVA: Is there differences in the mean weight of chickens that are feed different chicken feeds?
- Multiple comparison
- Kruskal- Wallis test

1.2. Analysis Methods

1.2.1. Introduction to Hypothesis testing

The Hypothesis Testing is a statistical test used to determine whether the hypothesis assumed for the sample of data stands true for the entire population or not. Simply, the hypothesis is an assumption which is tested to determine the relationship between two data sets.

In hypothesis testing, two opposing hypotheses about a population are formed. Null Hypothesis (H_0) and Alternative Hypothesis (H_1). The Null hypothesis is the statement which asserts that there is no difference between the sample statistic and population parameter and is the one which is tested, while the alternative hypothesis is the statement which stands true if the null hypothesis is rejected.

The following Hypothesis Testing Procedure is followed to test the assumption made:

Step 1: Set up a Hypothesis

The first step is to establish the hypothesis to be tested. The statistical hypothesis is an assumption about the value of some unknown parameter, and the hypothesis provides some numerical value or range of values for the parameter. Here two hypotheses about the population are constructed Null Hypothesis and Alternative Hypothesis.

The Null Hypothesis denoted by H_0 asserts that there is no true difference between the sample of data and the population parameter and that the difference is accidental which is caused due to the fluctuations in sampling. Thus, a null hypothesis states that there is no difference between the assumed and actual value of the parameter.

The alternative hypothesis denoted by H_1 is the other hypothesis about the population, which stands true if the null hypothesis is rejected. Thus, if we reject H_0 then the alternative hypothesis H_1 gets accepted.

For our frequentist statistics, our hypothesis:

- H_0: the data set comes from normal distribution.
- H_1: the data set does not come from normal distribution.

Step 2: Set up a Suitable Significance Level

Once the hypothesis about the population is constructed the researcher has to decide the level of significance, i.e. a confidence level with which the null hypothesis is accepted or rejected. The significance level is denoted by 'α' and is usually defined before the samples are drawn such that results obtained do not influence the choice. In practice, we either take 5% or 1% level of significance.

If the 5% level of significance is taken, it means that there are five chances out of 100 that we will reject the null hypothesis when it should have been accepted, i.e. we are about 95% confident that we have made the right decision. Similarly, if the 1% level of significance is taken, it means that there is only one chance out of 100 that we reject the hypothesis when it should have been accepted, and we are about 99% confident that the decision made is correct.

**NOTE: For our project, the confident level selected is 95%.*

Step 3: Determining a Suitable Test Statistic

After the hypothesis are constructed, and the significance level is decided upon, the next step is to determine a suitable test statistic and its distribution. Most of the statistic tests assume the following form:

$$Test\ statistic = \frac{Sample\ statistic - Hypothesized\ Parameter}{Standard\ Error\ of\ the\ statistic}$$

- **Determining the Critical Region:** Before the samples are drawn it must be decided that which values to the test statistic will lead to the acceptance of H_0 and which will lead to its rejection. The values that lead to rejection of H_0 is called the critical region.
- **Performing Computations:** Once the critical region is identified, we compute several values for the random sample of size 'n.' Then we will apply the formula of the test statistic as shown in step (3) to check whether the sample results falls in the acceptance region or the rejection region.
- **Decision-making:** Once all the steps are performed, the statistical conclusions can be drawn, and the management can take decisions. The decision involves either accepting the null hypothesis or rejecting it. The decision that the null hypothesis is accepted or rejected depends on whether the computed value falls in the acceptance region or the rejection region.

Thus, to test the hypothesis, it is necessary to follow these steps systematically so that the results obtained are accurate and do not suffer from either of the statistical error.

While testing the hypothesis, an individual may commit the following types of error:

- **Type-I Error:** True Null hypothesis is rejected, i.e. hypothesis is rejected when it should be accepted. The probability of committing the type-I error is denoted by α and is called as a level of significance.

  If, α = P(type-I error) = P(reject H_0|H_0 is true) Then, (1-α) = P(accept H0|H0 is true) (1-α) = corresponds to the concept of Confidence Interval.

- **Type-II Error:** A False Null hypothesis is accepted, i.e. hypothesis is accepted when it should be rejected. The probability of committing the type-II error is denoted by β.

  If, β = P(type-II error) = P(accept H_0|H_0 is false) Then, (1-β) = P(reject Ho|H0 is false) (1-β) = power of a statistical test.

Thus, hypothesis testing is the important method in the statistical inference that measures the deviations in the sample data from the population parameter. The hypothesis tests are widely used in the business and industry for making the crucial business decisions.

1.2.2. Post – Hoc tests

Only interpret post hoc tests for the significant factors from the ANOVA. If the interaction is NOT significant, interpret the post hoc tests for significant main effects but if it is significant, only interpret the interactions post hoc tests.

The interaction was significant so the main effects are not interpreted here but if your data does not have a significant interaction, interpret these in the same way as post hoc tests on the one-way ANOVA resource.

ANOVA tests the null hypothesis 'all group means are the same' so the resulting p-value only concludes whether or not there is a difference between one or more pairs of groups. If the ANOVA is significant, further 'post hoc' tests have to be carried out to confirm where those differences are. The post hoc tests are mostly t-tests with an adjustment to account for the multiple testing. Tukey's is the most commonly used post hoc test but check if your discipline uses something else. Use the command `TukeyHSD()`.

1.2.3. Analysis of Variance (ANOVA)

Introduction: Analysis of variance (ANOVA) is a collection of statistical models and their associated estimation procedures (such as the "variation" among and between groups) used to analyze the differences among group means in a sample. ANOVA was developed by statistician and evolutionary biologist Ronal Fisher.

The ANOVA is based on the law of total variance, where the observed variance in a particular variable is partitioned into components attributable to the observed variance in a particular variable is partitioned into components attributable to different sources of variation. In its simplest form, ANOVA provides a statistical test of whether two or more population means are equal, and therefore generalizes the t-test beyond two means.

Types of ANOVA: There are two types of ANOVA:

- **One-way ANOVA:** a hypothesis test in which only one categorical variable or single factor is taken into consideration. With the help of F-distribution, it enables us to compare the means of three or more samples. The null hypothesis (H_0) is the equity in all population means while an alternative hypothesis is a difference in at least one mean.
- **Two-way ANOVA:** a hypothesis test examines the effect of two independent factors on a dependent variable. It also studies the inter-relationship between independent variables influencing the values of the dependent variable, if any.

Formula:

$$F = \frac{MST}{MSE}$$

Where:

F: ANOVA coefficient

MST: Mean sum of squares due to the treatment

MSE: Mean sum of squares due to error

$$MST = \frac{SST}{p-1}$$

$$SST = \sum n(x - \bar{x})^2$$

Where:

SST: Sum of squares due to treatment

p: Total number of populations

n: The total number of samples in a population

$$MSE = \frac{SSE}{N-p}$$

$$SSE = \sum (n-1)S^2$$

Where:

SSE: Sum of squares due to errors

S: Standard deviation of the samples

N: Total number of observations

Application:

- To test the significance between the variance of two or more samples.
- To test correlation and regression.
- To study the homogeneity in case of two-way classification.
- To test the significance of the multiple correlation coefficient.
- To test the linearity of regression.
- Interpretation of the significance of means and their interactions.

1.2.4. Kruskal – Wallis test

The Kruskal Wallis test is the non-parametric alternative to the One Way ANOVA. Non parametric means that the test doesn't assume your data comes from a particular distribution. The H test is used when the assumptions for ANOVA aren't met (like the assumption of normality). It is sometimes called the one-way ANOVA on ranks, as the ranks of the data values are used in the test rather than the actual data points.

The test determines whether the medians of two or more groups are different. Like most statistical tests, you calculate a test statistic and compare it to a distribution cut-off point. The test statistic used in this test is called the H statistic**.** The hypotheses for the test are:

- H_0: population medians are equal.
- H_1: population medians are not equal.

The Kruskal Wallis test will tell you if there is a significant difference between groups. However, it won't tell you which groups are different. For that, you'll need to run a Post Hoc test.

$$H = \frac{12}{n(n+1)} \sum \frac{R_i^2}{n_i} - 3(n+1)$$

1.3. R/R-Studio Implementation

Set $\alpha=0.05$

Hypothesis:

H_0: The long – run mean weights are the same under all 6 crops

H_1: At least one of the long – run mean weights is different

1.3.1. Import data to R

We can easily open and import our data set into Rstudio, by the help of `read.csv()` function and set the variable to *chickendata*.

Let see what our *chickendata* holds

```
> chickendata<-read.csv("chicken_feed.csv",header = T,sep= ",")
> chickendata
    X weight      feed
1   1    179 horsebean
2   2    160 horsebean
3   3    136 horsebean
4   4    227 horsebean
5   5     NA horsebean
6   6    168 horsebean
7   7    108 horsebean
8   8    124 horsebean
9   9    143 horsebean
10 10    140 horsebean
11 11    309   linseed
12 12    229   linseed
13 13    181   linseed
14 14    141   linseed
15 15    260   linseed
16 16    203   linseed
17 17     NA   linseed
18 18    169   linseed
19 19    213   linseed
20 20    257   linseed
21 21    244   linseed
22 22    271   linseed
23 23    243   soybean
24 24    230   soybean
25 25    248   soybean
26 26    327   soybean
27 27    329   soybean
28 28    250   soybean
29 29    193   soybean
30 30    271   soybean
31 31    316   soybean
```

32	32	267	soybean
33	33	199	soybean
34	34	171	soybean
35	35	158	soybean
36	36	248	soybean
37	37	423	sunflower
38	38	340	sunflower
39	39	392	sunflower
40	40	339	sunflower
41	41	341	sunflower
42	42	226	sunflower
43	43	320	sunflower
44	44	295	sunflower
45	45	334	sunflower
46	46	322	sunflower
47	47	297	sunflower
48	48	318	sunflower
49	49	325	meatmeal
50	50	257	meatmeal
51	51	303	meatmeal
52	52	315	meatmeal
53	53	380	meatmeal
54	54	153	meatmeal
55	55	263	meatmeal
56	56	242	meatmeal
57	57	206	meatmeal
58	58	344	meatmeal
59	59	258	meatmeal
60	60	368	casein
61	61	390	casein
62	62	379	casein
63	63	260	casein
64	64	404	casein
65	65	318	casein
66	66	352	casein
67	67	359	casein
68	68	216	casein
69	69	222	casein
70	70	283	casein
71	71	332	casein

Quick overview our data by `summary()`

```
> summary(chickendata)
       X             weight          feed
 Min.   : 1.0   Min.   :108.0   Length:71
 1st Qu.:18.5   1st Qu.:206.0   Class :character
 Median :36.0   Median :260.0   Mode  :character
 Mean   :36.0   Mean   :263.6
 3rd Qu.:53.5   3rd Qu.:325.0
 Max.   :71.0   Max.   :423.0
                NA's   :2
```

As the 'x' and 'feed' are factors, we do not expect to see the mean of them, instead I want to cover all the observation on each treatment.

To do that we modified our read function a little bit to treat 'x', 'feed' as factors and the 'weight' as af numeric value while reading:

```
> chickendata<-read.csv("C:/Users/Admin/Downloads/chicken_feed.csv", header = T, colClasses = c('factor','numeric','factor'))
```

Then we got a better summary:

```
> chickendata<-read.csv("C:/Users/Admin/Downloads/chicken_feed.csv", header = T, colClasses = c('factor','numeric','factor'))
> summary(chickendata)
       X          weight           feed
 1      : 1   Min.   :108.0   casein   :12
 10     : 1   1st Qu.:206.0   horsebean:10
 11     : 1   Median :260.0   linseed  :12
 12     : 1   Mean   :263.6   meatmeal :11
 13     : 1   3rd Qu.:325.0   soybean  :14
 14     : 1   Max.   :423.0   sunflower:12
 (Other):65   NA's   :2
```

What we got from the summary are:

The mean of weight of all observation is 263.6

The median: 260.0

The ¼ first values ends at 206.0

The ¼ last values starts at 325.0

As we expected, this `summary()` give us the number per each level of feed, for example: there are 12 observations in the casein-type of feed, 10 in horsebean,…

Also, in the last row, we noticed that there are many observations that do not give the exact weight of the chicken, noted as NA-Not Available, in the 'weight' column. There are plenty of reasons which lead to this problem, and of course this will affect our correctness of analyzing. We must do something to cover it.

1.3.2. Handle missing value and update table

Cleaning data is a huge field in statistic. Numbers of issues may cause our data to be 'bad', such as duplicate observations or irrelevant observations. We just can't ignore missing data because many algorithms will not accept missing values. In this activity, we only deal with the NA problem.

As our data is not too big and the summary give us the information that there are only 2 missing value in our data.

We can easily track their location:

```
> is.na(chickendata)
          X weight  feed
 [1,] FALSE  FALSE FALSE
 [2,] FALSE  FALSE FALSE
 [3,] FALSE  FALSE FALSE
 [4,] FALSE  FALSE FALSE
 [5,] FALSE   TRUE FALSE
 [6,] FALSE  FALSE FALSE
 [7,] FALSE  FALSE FALSE
 [8,] FALSE  FALSE FALSE
 [9,] FALSE  FALSE FALSE
[10,] FALSE  FALSE FALSE
[11,] FALSE  FALSE FALSE
[12,] FALSE  FALSE FALSE
[13,] FALSE  FALSE FALSE
[14,] FALSE  FALSE FALSE
[15,] FALSE  FALSE FALSE
[16,] FALSE  FALSE FALSE
[17,] FALSE   TRUE FALSE
[18,] FALSE  FALSE FALSE
```

Our missing values locate at row 7 and row 17, in the 'weight' column.

```
> sum(is.na(chickendata))
[1] 2
> which(is.na(chickendata), arr.ind = T)
     row col
[1,]   5   2
[2,]  17   2
```

There are several ways to handle missing data. One of those is to replace them with the mean of the others. But, since our data set is small and also, when using this method we need to deal with terms called outliners may occur in our data. They are nothing but an extreme value that deviates from the other observations in the dataset. Moreover, tons of ways are found in the internet; however we do not able to handle this problem.

So, another approach seems to be better in this situation. We can skip the missing values by simply remove them from our data. Doing this will drop or lose information but we think we known what we are doing.

```
> chickendata<-na.omit(chickendata) #omit missing value and update chickendata
> chickendata #new table
    X weight      feed
1   1    179 horsebean
2   2    160 horsebean
3   3    136 horsebean
4   4    227 horsebean
6   6    168 horsebean
7   7    108 horsebean
8   8    124 horsebean
9   9    143 horsebean
10 10    140 horsebean
11 11    309   linseed
12 12    229   linseed
13 13    181   linseed
14 14    141   linseed
15 15    260   linseed
16 16    203   linseed
18 18    169   linseed
19 19    213   linseed
20 20    257   linseed
21 21    244   linseed
22 22    271   linseed
23 23    243   soybean
24 24    230   soybean
25 25    248   soybean
26 26    327   soybean
27 27    329   soybean
28 28    250   soybean
29 29    193   soybean
30 30    271   soybean
31 31    316   soybean
32 32    267   soybean
33 33    199   soybean
34 34    171   soybean
35 35    158   soybean
36 36    248   soybean
37 37    423 sunflower
38 38    340 sunflower
39 39    392 sunflower
40 40    339 sunflower
41 41    341 sunflower
42 42    226 sunflower
43 43    320 sunflower
44 44    295 sunflower
45 45    334 sunflower
46 46    322 sunflower
47 47    297 sunflower
48 48    318 sunflower
49 49    325  meatmeal
50 50    257  meatmeal
51 51    303  meatmeal
52 52    315  meatmeal
53 53    380  meatmeal
54 54    153  meatmeal
55 55    263  meatmeal
56 56    242  meatmeal
57 57    206  meatmeal
58 58    344  meatmeal
59 59    258  meatmeal
60 60    368    casein
61 61    390    casein
62 62    379    casein
63 63    260    casein
64 64    404    casein
65 65    318    casein
66 66    352    casein
67 67    359    casein
68 68    216    casein
69 69    222    casein
70 70    283    casein
71 71    332    casein
```

The `na.omit()` function just simply remove every row that have the NA value in any column.

Since there are only two NA value in the 'weight' column, this function works fine for us to keep track what we are doing.

The alternative function to do this task is `drop_na()` by {tidyr} which can drop rows containing missing values.

```
> chickendata<-drop_na(chickendata)
```

Checking our work so far:

```
> summary(chickendata)
       X          weight          feed
 1      : 1   Min.   :108.0   casein   :12
 10     : 1   1st Qu.:206.0   horsebean: 9
 11     : 1   Median :260.0   linseed  :11
 12     : 1   Mean   :263.6   meatmeal :11
 13     : 1   3rd Qu.:325.0   soybean  :14
 14     : 1   Max.   :423.0   sunflower:12
 (Other):63
> sum(is.na(chickendata))
[1] 0
```

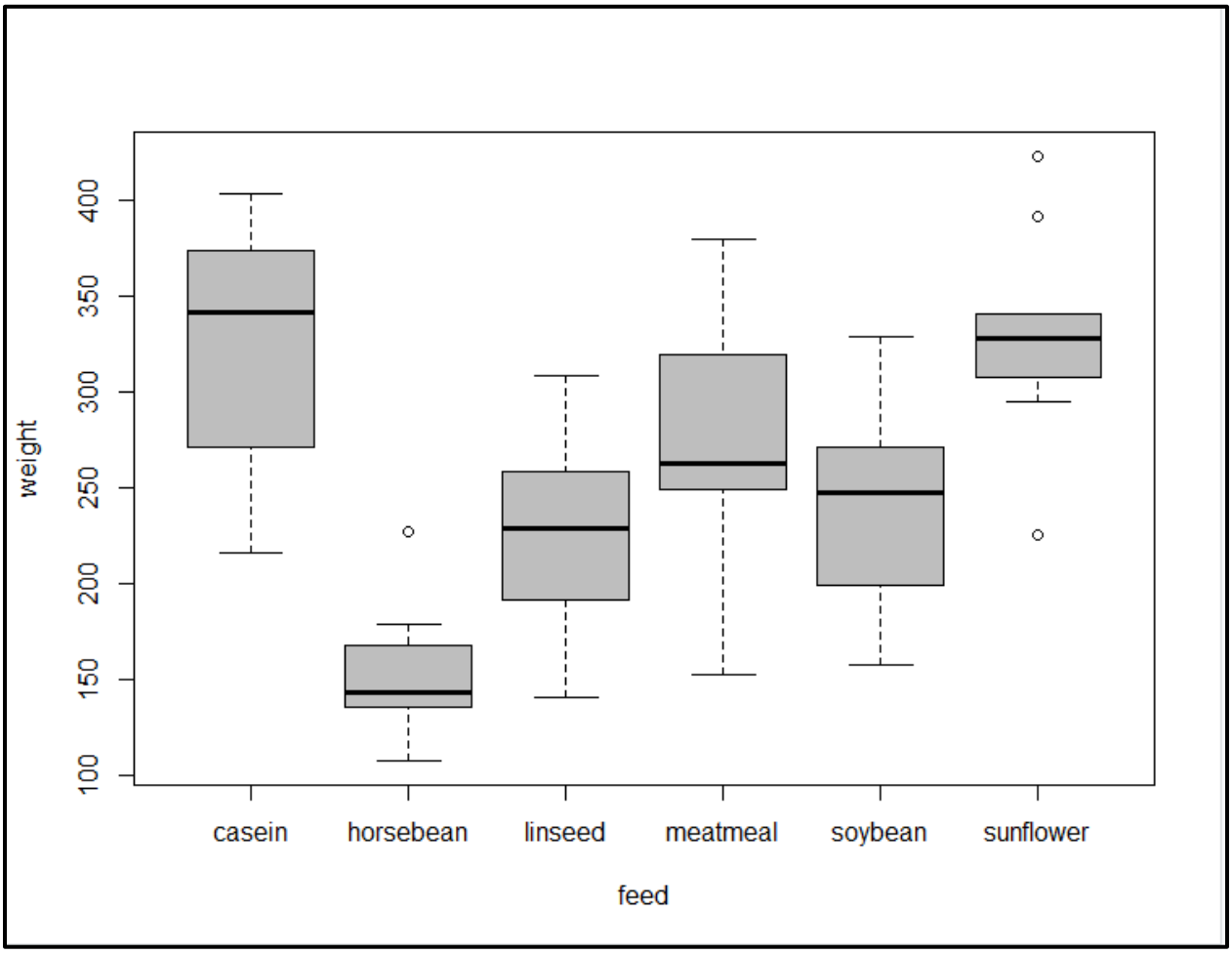
1.3.3. Data Visualization

By using function *unique()* we get that there are 6 types of food we are interested in:

```
> unique(chickendata$feed) #show number of feed
[1] "horsebean" "linseed"   "soybean"   "sunflower" "meatmeal"  "casein"
```

Draw boxplot of the mean weight of each feed

```
> boxplot(weight~feed,data=chickendata,col='gray') #show boxplot from data
```



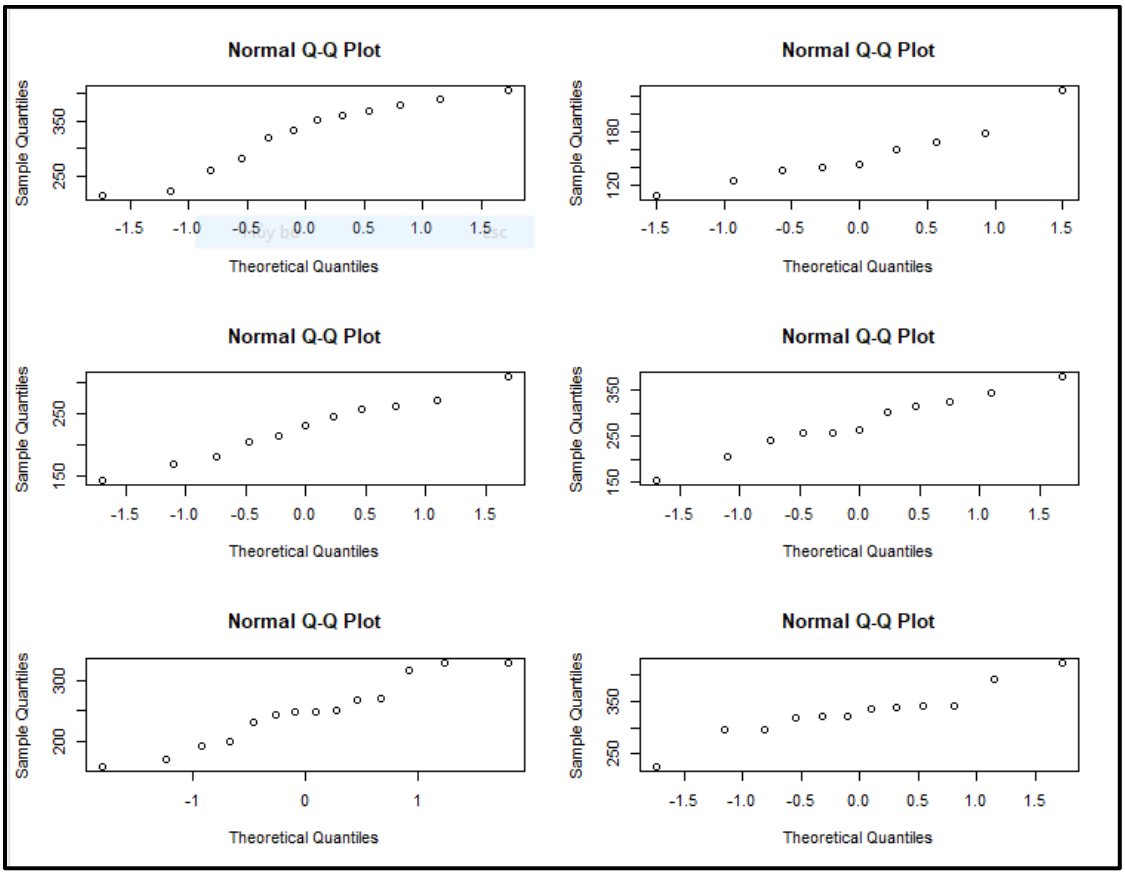
The boxplots are pretty different. Casein feed looks much better than horsebean for example. But the sample sizes in each group are very small, so it is not obvious that the differences will be statistically significant. Below is an example of how to do ANOVA with R.

1.3.4. Using ANOVA

```
> Anova_result<-aov(weight~feed,data=chickendata)#handle data
> summary(Anova_result) #show table
            Df Sum Sq Mean Sq F value   Pr(>F)
feed         5 225012   45002    15.2 8.83e-10 ***
Residuals   63 186511    2960
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> p_valweight=summary(Anova_result)[[1]][["Pr(>F)"]] #get F_value from table Anova_Result
> p_value=p_valweight[1]
> p_value #p_value that use compare
[1] 8.833496e-10
>
> #conclusion
> if(p_value<0.05){
+   print("We have evidence to prove there is a different in the mean weight of chickens that are feed different chicken feeds")
+ }else{
+   print("We don't have evidence to prove there is a different in the mean weight of chickens that are feed different chicken fe
eds")
+ }
[1] "We have evidence to prove there is a different in the mean weight of chickens that are feed different chicken feeds"
```

Based on the F_value and its p_value, we should be able to safely reject the null hypothesis and conclude that there are differences in the mean weight of chickens that are feed different chicken feeds. Before we do that, let's double check that data in each group is roughly normal. Because the groups are so small, a qq-plot is probably a better choice than a histogram.

```
> #qqnorm to check condition
> par(mfrow=c(3,2))
> qqnorm(subset(chickendata,feed=='casein')$weight)
> qqnorm(subset(chickendata,feed=='horsebean')$weight)
> qqnorm(subset(chickendata,feed=='linseed')$weight)
> qqnorm(subset(chickendata,feed=='meatmeal')$weight)
> qqnorm(subset(chickendata,feed=='soybean')$weight)
> qqnorm(subset(chickendata,feed=='sunflower')$weight)
```



These normal quantile plots are reasonably close to straight lines, so the normality assumption is probably okay. We also need to double check that the constant variance assumption isn't way off. The rule of thumb from the book was to make sure that none of the sample standard deviations are separated by a factor greater than 2. Here is one way to quickly calculate the sample standard deviation for each group.

```
> aggregate(weight~feed,data=chickendata,FUN=sd) #calculate sd weight of each feed
       feed   weight
1    casein 64.43384
2 horsebean 35.07650
3   linseed 49.55163
4  meatmeal 64.90062
5   soybean 54.12907
6 sunflower 48.83638
```

Since the ratio of the largest standard deviation (64.9) to the smallest (38.6) is less than 2, we are probably safe assuming that the data has a constant standard deviation. We also don't need to worry about the independence assumption since this was a random sample of chickens. Thus our conclusion that the differences are significant is almost certainly valid.

In a more popular way, **Levene's test is considered.**

This method tests for comparing the **variances** of two or more samples. Equal variances across samples is called **homogeneity** of **variances**.

For Levene's test the statistical hypotheses are:

- Null Hypothesis: All population variation are equal
- Alternative Hypothesis: At least two of them differ

In Rstudio, the function `leveneTest()` [in **car** package] can be used.

```
> #load the library
> library(car)
> #Levene's test
> leveneTest(weight ~ feed, data = chickendata)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  5  0.9016 0.4858
      63
```

Noted the p-value, it is more than the α level is 0.05 (0.4858>0.05). We cannot reject the null hypothesis, which lead to the fact that we don't have enough evidence to prove that the variation is different among considering samples. We decided continue analyzing ANOVA and believe it will give us some necessary knowledge about this data as well as suggest an alternative method for non-parametric data which will be discussed in a chapter below.

1.3.5. Calculate mean weight of each feed to multiple comparison

```
> aggregate(weight~feed,data=chickendata,FUN=mean) #calculate mean weight of each feed to compare
       feed   weight
1    casein 323.5833
2 horsebean 153.8889
3   linseed 225.1818
4  meatmeal 276.9091
5   soybean 246.4286
6 sunflower 328.9167
```

```
> #multiple comparison by bonferroni methods
> pairwise.t.test(chickendata$weight,chickendata$feed,p.adj='bonferroni') #Pairwise comparisons using t tests with pooled SD

        Pairwise comparisons using t tests with pooled SD

data:  chickendata$weight and chickendata$feed

          casein  horsebean linseed meatmeal soybean
horsebean 2.2e-08 -         -       -        -
linseed   0.00081 0.07376   -       -        -
meatmeal  0.66042 6.5e-05   0.44024 -        -
soybean   0.00927 0.00271   1.00000 1.00000  -
sunflower 1.00000 9.2e-09   0.00035 0.38085  0.00413

P value adjustment method: bonferroni
```

Notice the argument p.adj, which is the method for adjusting significance levels to avoid type I errors. In this case, there are 15 pairwise comparison, so R has increased all of the p-values by a factor of 15 to compensate. Notice that multiplying the p-values by 15 is pretty much the same as dividing the αα by 15. This way, however, if a p-value in the table above looks significant (i.e., below 0.05), then we can safely conclude that it is statistically significant.

It is also possible to construct confidence intervals for each difference above. The most common technique is known as Tukey's Honest Significant Differences, and the command is `TukeyHSD()`, as shown below:

```
> #multiple comparison by TurkeyHSD of Post Hoc test
> TukeyHSD(Anova_result,conf.level=0.95)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = weight ~ feed, data = chickendata)

$feed
                           diff          lwr       upr     p adj
horsebean-casein    -169.694444 -240.2146894 -99.17420 0.0000000
linseed-casein       -98.401515 -165.1579285 -31.64510 0.0007400
meatmeal-casein      -46.674242 -113.4306557  20.08217 0.3240085
soybean-casein       -77.154762 -140.0688745 -14.24065 0.0077887
sunflower-casein       5.333333  -59.9557269  70.62239 0.9998856
linseed-horsebean     71.292929   -0.5879602 143.17382 0.0531327
meatmeal-horsebean   123.020202   51.1393125 194.90109 0.0000620
soybean-horsebean     92.539683   24.2123152 160.86705 0.0023904
sunflower-horsebean  175.027778  104.5075328 245.54802 0.0000000
meatmeal-linseed      51.727273  -16.4649266 119.91947 0.2391269
soybean-linseed       21.246753  -43.1888185  85.68232 0.9259562
sunflower-linseed    103.734848   36.9784352 170.49126 0.0003279
soybean-meatmeal     -30.480519  -94.9160912  33.95505 0.7325632
sunflower-meatmeal    52.007576  -14.7488376 118.76399 0.2135819
sunflower-soybean     82.488095   19.5739826 145.40221 0.0035948
```

From this table, we can see from the bottom row that the difference between the average weights of chickens fed sunflower feed versus soybean feed will be between 19. and 145.8 (with 95%) confidence. Notice that the adjusted p-values are little lower than the ones using the Bonferroni condition. The Tukey method is a little more complicated, and it is a little less conservative.

We can see that there are some significant differences between (as the p-value is significantly small):

Horsebean - Casein

Sunflower - Horsebean

1.3.6. Kruskal – Wallis test

According to the previous introduction, this session will handle this bunch of data when the conditions for ANOVA do not match.

The Kruskal Wallis test is the non-parametric alternative to the One Way ANOVA. The H test is used when the assumptions for ANOVA aren't met (like the assumption of normality). It is sometimes called the one-way ANOVA on ranks.

There are two hypothesis in the Kruskal-Wallis test:

- Null hypothesis H_0: Population medians are equal.
- Alternative hypothesis H_1: Population medians are not equal.

Unlike one-way ANOVA, which deal with the means among level, the Kruskal-Wallis test use medians to evaluate the difference between groups. The Kruskal-Wallis test calculates H statistic $H = \frac{12}{n(n+1)} \sum \frac{R_i^2}{n_i} - 3(n+1)$ and Chi – square value.

An the Kruskal – Wallis test

```
        Kruskal-Wallis rank sum test

data:  weight by feed
Kruskal-Wallis chi-squared = 36.038, df = 5, p-value = 9.335e-07
```

We see that the p-value is less than 0.05 so we have evidence to prove there is a different in the mean weight of chickens that are feed different chicken feeds.

1.4. Conclusion

The problem is interesting, but to us, the survey is not that good, we think. It is small and narrow, which may lead to there are several approaches may be taken in different assumption of the homogeneous of the data. We tried many of them, each of which give us a closer look about data and what is the power of data analyzing.

We, therefore, have sufficient evidence to reject the very first null hypothesis. Our initial guess that a statistically significant difference existed in the means was backed by this statistical analysis. We have evidence to suggest that weight of chickens is affected by feed given.

1.5. Code implementation

```
#import data from chicken_feed.csv
#set alpha = 0.05
#H0: The long-run mean weights are the same under all six crops.
#H1: At least one of the long-run mean weights is different.

chickendata<-read.csv("chicken_feed.csv",header = T,sep= ",")
chickendata
chickendata<-read.csv("C:/Users/Admin/Downloads/chicken_feed.csv", header = T,
colClasses = c('factor','numeric','factor'))
summary(chickendata)
#handle missing value in table
is.na(chickendata) #find missing value
which(is.na(chickendata),arr.ind = T) #identify index
sum(is.na(chickendata)) #count number of missing value
chickendata<-na.omit(chickendata) #omit missing value and update chickendata
chickendata #new table

drop_na(chickendata)
sum(is.na(chickendata))# check again the number of missing value

unique(chickendata$feed) #show number of feed

boxplot(weight~feed,data=chickendata,col='gray') #show boxplot from data

Anova_result<-aov(weight~feed,data=chickendata)#handle data
```

```
summary(Anova_result) #show table


p_valWeight=summary(Anova_result)[[1]][["Pr(>F)"]] #get F_value from table
Anova_Result
p_value=p_valWeight[1]
p_value #p_value that use compare


#conclusion
if(p_value<0.05){
  print("We have evidence to prove there is a different in the mean weight of
chickens that are feed different chicken feeds")
}else{
  print("We don't have evidence to prove there is a different in the mean weight of
chickens that are feed different chicken feeds")
}


#qqnorm to check condition
par(mfrow=c(3,2))
qqnorm(subset(chickendata,feed=='casein')$weight)
qqnorm(subset(chickendata,feed=='horsebean')$weight)
qqnorm(subset(chickendata,feed=='linseed')$weight)
qqnorm(subset(chickendata,feed=='meatmeal')$weight)
qqnorm(subset(chickendata,feed=='soybean')$weight)
qqnorm(subset(chickendata,feed=='sunflower')$weight)


aggregate(weight~feed,data=chickendata,FUN=sd) #calculate sd weight of each feed


aggregate(weight~feed,data=chickendata,FUN=mean) #calculate mean weight of each feed
to compare


#load the library
library(car)
#Levene's test
leveneTest(weight ~ feed, data = chickendata)


#multiple comparison by bonferroni methods
```

```
pairwise.t.test(chickendata$weight,chickendata$feed,p.adj='bonferroni') #Pairwise
comparisons using t tests with pooled SD


#multiple comparison by TurkeyHSD of Post Hoc test
TukeyHSD(Anova_result,conf.level=0.95)


#Kruskal-Wallis test
kruskal.test(weight ~ feed, data = chickendata)
```

2. Activity 2

2.1. Choose the Dataset

We take dataset from this url

https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings

This is a dataset about video game sales. In this dataset, we only use **Global_Sales** and **Genre** column to make the dataset clearer and easier to work with.

Attribute Information

- Genre: Game's category
- Global_Sales: Total sales global (in millions of units)

2.2. R/R – Studio Implementation

2.2.1. Import data to R

We import dataset from a file named Video_Games_Sales.csv

We take the first 100 rows only.

```
#Import data
data <- read.csv(file = '/Users/khaiphan/Downloads/Video_Games_Sales.csv', sep = ",",header = T,
                 nrows = 100)
#find the head's number to get choosen data
names(data)
newdata <- data[,c(4,10)]
#eliminate line 1 to use ANOVA-One-way test
newdata <- newdata[-c(1),]
newdata
```

Below is the result:

	Genre	Global_Sales
2	Platform	40.24
3	Racing	35.52
4	Sports	32.77
5	Role-Playing	31.37
6	Puzzle	30.26
7	Platform	29.80
8	Misc	28.92
9	Platform	28.32
10	Shooter	28.31
11	Simulation	24.67
12	Racing	23.21
13	Role-Playing	23.10
14	Sports	22.70
15	Misc	21.81
16	Sports	21.79
17	Action	21.04
18	Action	20.81
19	Platform	20.61
20	Misc	20.15
21	Role-Playing	18.25
22	Platform	18.14
23	Platform	17.28
24	Action	16.27
25	Action	16.15
26	Role-Playing	15.85
27	Puzzle	15.29
28	Role-Playing	15.14
29	Racing	14.98
30	Shooter	14.73
31	Role-Playing	14.64
32	Shooter	14.63
33	Shooter	14.61
34	Role-Playing	14.60
35	Shooter	13.79
36	Shooter	13.67
37	Shooter	13.47
38	Shooter	13.32
39	Action	13.10
40	Fighting	12.84
41	Racing	12.66
42	Shooter	12.63
43	Action	12.61
44	Simulation	12.13
45	Shooter	12.12
46	Platform	11.89
47	Action	11.77
48	Role-Playing	11.68
49	Racing	11.66
50	Platform	11.35
51	Adventure	11.18
52	Action	11.01
53	Racing	10.95
54	Platform	10.81
55	Racing	10.70
56	Shooter	10.60
57	Platform	10.55
58	Action	10.50
59	Role-Playing	10.49
60	Platform	10.30
61	Shooter	10.25
62	Misc	10.12
63	Platform	9.90
64	Racing	9.87
65	Shooter	9.86
66	Role-Playing	9.72
67	Shooter	9.71
68	Racing	9.49
69	Misc	9.44
70	Shooter	9.36
71	Shooter	9.31
72	Platform	9.30
73	Misc	9.18
74	Simulation	9.16
75	Misc	8.91
76	Role-Playing	8.79
77	Racing	8.76
78	Sports	8.57
79	Shooter	8.49
80	Misc	8.38
81	Misc	8.27
82	Action	8.16
83	Shooter	8.09
84	Role-Playing	8.07
85	Role-Playing	8.05
86	Simulation	8.01
87	Sports	7.99
88	Shooter	7.98
89	Role-Playing	7.86
90	Puzzle	7.81
91	Role-Playing	7.72
92	Action	7.69
93	Shooter	7.66
94	Action	7.60
95	Sports	7.59
96	Platform	7.58
97	Fighting	7.55
98	Platform	7.51
99	Platform	7.46
100	Shooter	7.39

2.2.2. Data cleaning: NA (Not available)

We use `na.omit()` to clean data

```
#Data cleaning
newdata <- na.omit(newdata)
```

2.2.3. Data visualization

Step 1: Descriptive statistics for each of the variable

Our descriptive statistics of a variable includes

- Minimum value
- Maximum value
- Range
- Mean
- Median
- The standard deviation
- The standard variance

Here, we calculate the descriptive statistics for Global_Sales:

```
#min and max of global sales
rng <- range(newdata$Global_Sales)
#min
rng[1]
#max
rng[2]
#range of global sales
rng[2] - rng[1]

#mean
mean(newdata$Global_Sales)
#median
median(newdata$Global_Sales)
#the standard deviation
sd(newdata$Global_Sales)
#the standard variance
var(newdata$Global_Sales)
```

The value of descriptive statistic for Global_Sales:

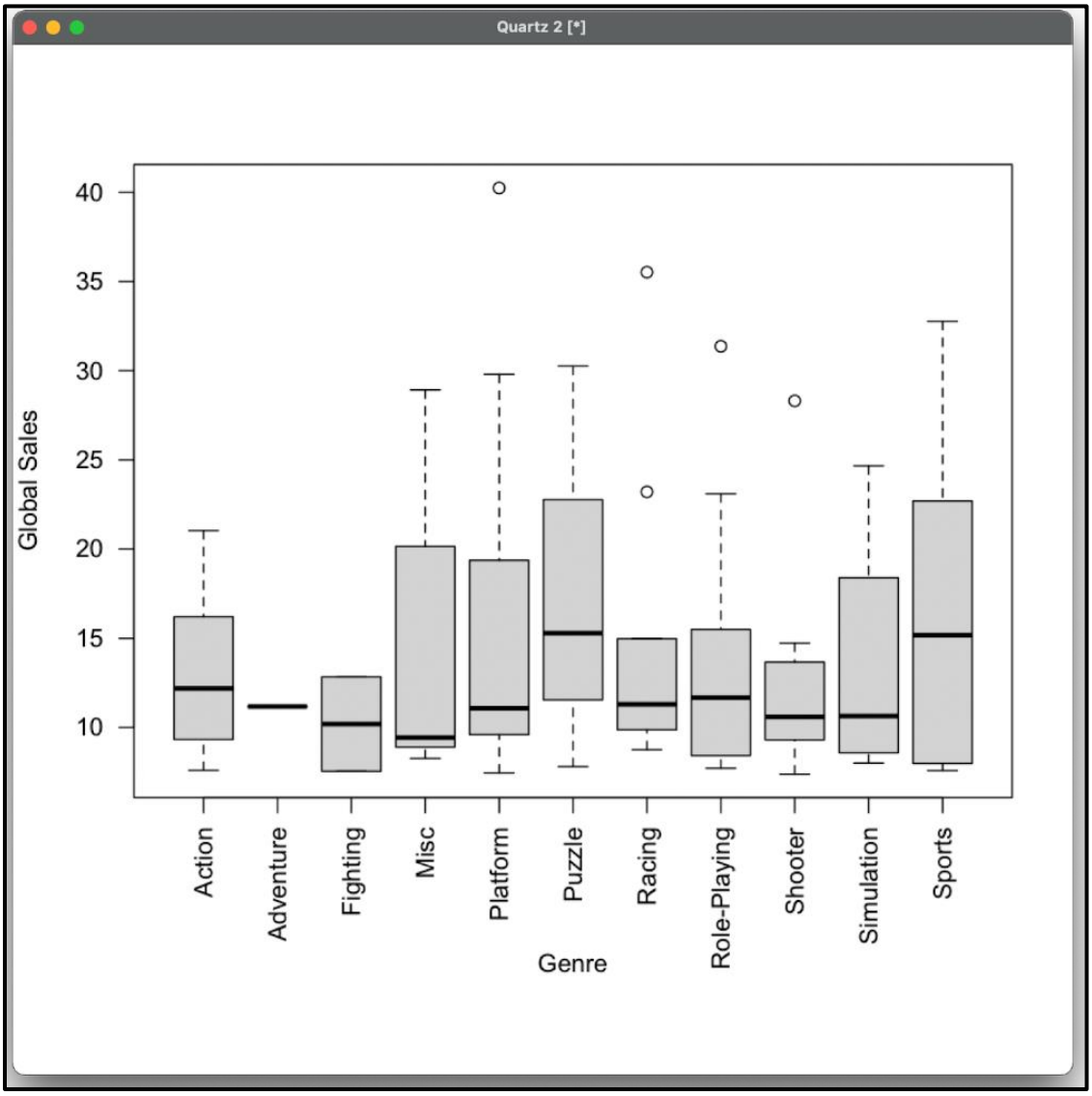
```
> #min and max of global sales
> rng <- range(newdata$Global_Sales)
> #min
> rng[1]
[1] 7.39
> #max
> rng[2]
[1] 40.24
> #range of global sales
> rng[2] - rng[1]
[1] 32.85
> #mean
> mean(newdata$Global_Sales)
[1] 13.90253
> #median
> median(newdata$Global_Sales)
[1] 11.18
> #the standard deviation
> sd(newdata$Global_Sales)
[1] 7.168831
> #the standard variance
> var(newdata$Global_Sales)
[1] 51.39214
```

Step 2: Graph: boxplot – Global Sales for each type of game (Genre)

We adjust the margins of the boxplot to make the boxplot clearer

```
#modify margines of boxplot
par(mai=c(2.0, 0.8, 0.8, 0.4))
#draw boxplot
boxplot(newdata$Global_Sales ~ newdata$Genre, las = 2, xlab = "", ylab = "Global Sales")
title(xlab = "Genre", line = 5);
```

Our result:



Conclusion from the boxplot:

The median of all kinds of games seem to be the same except for Sports and Puzzles games.

2.2.4. One – way ANOVA test

We use one-way ANOVA test to answer the questions:

At the significance level of 5%, is there a difference in average Global sales among game's genres?

Hypothesis:

- H_0: There is not a difference in average Global sales among game's genres
- H_1: There is a difference in average Global sales among game's genres

```
#One-way ANOVA test
Anova_Result = aov(Global_Sales ~ Genre, data = newdata)
summary(Anova_Result) #Show Anova result

#Get F_value from Anova_Result
F_Value_Genre <- (summary(Anova_Result))[[1]][["F value"]]
F_value <- F_Value_Genre[1]
F_value #F_value that used to compare

df <- (summary(Anova_Result))[[1]][["Df"]] #get df from Anova_Result
df_b = df[1] #df between
df_w = df[2] #df within
df_b
df_w

F_critical = qf(0.95, df_b, df_w) #search F_critical from F distribution
F_critical

#Conclusion
if(F_value >= F_critical) {
  print("At alpha = 0.05, we have enough evidence to conclude that")
  print(" there is a difference between Global_sales in different Game Genre")
}else{
  print("At alpha = 0.05, We do not have enough evidence to conclude that")
  print(" there is a difference between Global_sales in different Game Genre")
}
```

The result of One – way ANOVA test

```
> #One-way ANOVA test
> Anova_Result = aov(Global_Sales ~ Genre, data = newdata)
> summary(Anova_Result) #Show Anova result
             Df Sum Sq Mean Sq F value Pr(>F)
Genre        10    287   28.67   0.531  0.864
Residuals    88   4750   53.97
> #Get F_value from Anova_Result
> F_Value_Genre <- (summary(Anova_Result))[[1]][["F value"]]
> F_value <- F_Value_Genre[1]
> F_value #F_value that used to compare
[1] 0.5312566
> df <- (summary(Anova_Result))[[1]][["Df"]] #get df from Anova_Result
> df_b = df[1] #df between
> df_w = df[2] #df within
> df_b
[1] 10
> df_w
[1] 88
> F_critical = qf(0.95, df_b, df_w) #search F_critical from F distribution
> F_critical
[1] 1.940044
> #Conclusion
> if(F_value >= F_critical) {
+       print("At alpha = 0.05, we have enough evidence to conclude that")
+       print(" there is a difference between Global_sales in different Game Genre")
+ }else{
+       print("At alpha = 0.05, We do not have enough evidence to conclude that")
+       print(" there is a difference between Global_sales in different Game Genre")
+ }
[1] "At alpha = 0.05, We do not have enough evidence to conclude that"
[1] " there is a difference between Global_sales in different Game Genre"
```

Conclusion:

At α=0.05, we do not have enough evidence to conclude that there is a difference between Global_Sales in different game Genre

2.2.5. Multiple comparison: Tukey multiple pairwise – comparisons

As the ANOVA test is significant, we can compute **Tukey HSD** (Tukey Honest Significant Differences, R function: `TukeyHSD()`) for performing multiple pairwise-comparison between the means of groups.

```
#Multiple pair comparison
TukeyHSD(Anova_Result)
```

Result:

```
> TukeyHSD(Anova_Result)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Global_Sales ~ Genre, data = newdata)

$Genre
                              diff        lwr       upr     p adj
Adventure-Action        -1.8791667 -27.150807 23.392474 1.0000000
Fighting-Action         -2.8641667 -21.408481 15.680147 0.9999878
Misc-Action              0.8497222  -9.856842 11.556287 1.0000000
Platform-Action          2.6308333  -6.641324 11.902990 0.9971867
Puzzle-Action            4.7275000 -10.945306 20.400306 0.9953763
Racing-Action            1.7208333  -8.675330 12.116997 0.9999765
Role-Playing-Action      0.6295000  -8.774184 10.033184 1.0000000
Shooter-Action          -1.1553571  -9.941719  7.631005 0.9999974
Simulation-Action        0.4333333 -13.584850 14.451517 1.0000000
Sports-Action            3.8425000  -8.297603 15.982603 0.9932289
Fighting-Adventure      -0.9850000 -30.722058 28.752058 1.0000000
Misc-Adventure           2.7288889 -22.864696 28.322474 0.9999997
Platform-Adventure       4.5100000 -20.517464 29.537464 0.9999483
Puzzle-Adventure         6.6066667 -21.429701 34.643034 0.9994194
Racing-Adventure         3.6000000 -21.865295 29.065295 0.9999947
Role-Playing-Adventure   2.5086667 -22.567823 27.585156 0.9999998
Shooter-Adventure        0.7238095 -24.127774 25.575393 1.0000000
Simulation-Adventure     2.3125000 -24.833596 29.458596 1.0000000
Sports-Adventure         5.7216667 -20.503954 31.947287 0.9997060
Misc-Fighting            3.7138889 -15.266822 22.694599 0.9998899
```

Platform-Fighting	5.4950000	-12.715155	23.705155	0.9953619
Puzzle-Fighting	7.5916667	-14.573028	29.756361	0.9875923
Racing-Fighting	4.5850000	-14.222367	23.392367	0.9992197
Role-Playing-Fighting	3.4936667	-14.783809	21.771142	0.9999110
Shooter-Fighting	1.7088095	-16.258856	19.676475	0.9999999
Simulation-Fighting	3.2975000	-17.729776	24.324776	0.9999859
Sports-Fighting	6.7066667	-13.118039	26.531372	0.9886968
Platform-Misc	1.7811111	-8.335642	11.897864	0.9999583
Puzzle-Misc	3.8777778	-12.309027	20.064582	0.9993296
Racing-Misc	0.8711111	-10.284874	12.027096	1.0000000
Role-Playing-Misc	-0.2202222	-10.457656	10.017212	1.0000000
Shooter-Misc	-2.0050794	-11.678545	7.668386	0.9998143
Simulation-Misc	-0.4163889	-15.006977	14.174199	1.0000000
Sports-Misc	2.9927778	-9.804015	15.789570	0.9994566
Puzzle-Platform	2.0966667	-13.179295	17.372628	0.9999960
Racing-Platform	-0.9100000	-10.697664	8.877664	0.9999999
Role-Playing-Platform	-2.0013333	-10.727582	6.724916	0.9995423
Shooter-Platform	-3.7861905	-11.843380	4.270999	0.8970403
Simulation-Platform	-2.1975000	-15.770548	11.375548	0.9999809
Sports-Platform	1.2116667	-10.411597	12.834931	0.9999997
Racing-Puzzle	-3.0066667	-18.989855	12.976522	0.9999232
Role-Playing-Puzzle	-4.0980000	-19.454151	11.258151	0.9983107
Shooter-Puzzle	-5.8828571	-20.868926	9.103212	0.9669539
Simulation-Puzzle	-4.2941667	-22.838481	14.250147	0.9995019
Sports-Puzzle	-0.8850000	-18.053699	16.283699	1.0000000
Role-Playing-Racing	-1.0913333	-11.003686	8.821019	0.9999995
Shooter-Racing	-2.8761905	-12.204943	6.452562	0.9944943
Simulation-Racing	-1.2875000	-15.651864	13.076864	0.9999999
Sports-Racing	2.1216667	-10.416578	14.659911	0.9999712
Shooter-Role-Playing	-1.7848571	-9.993065	6.423351	0.9997146
Simulation-Role-Playing	-0.1961667	-13.859402	13.467069	1.0000000
Sports-Role-Playing	3.2130000	-8.515454	14.941454	0.9978962
Simulation-Shooter	1.5886905	-11.657248	14.834629	0.9999989
Sports-Shooter	4.9978571	-6.241694	16.237409	0.9258236
Sports-Simulation	3.4091667	-12.263639	19.081973	0.9997137

- **diff**: difference between means of the two groups
- **lwr**, **upr**: the lower and the upper end point of the confidence interval at 95% (default)
- **p adj**: p-value after adjustment for the multiple comparisons.

It can be seen from the output, there is no difference between the sales of game's genres.

2.2.6. Test ANOVA assumption

Step 1: Check the homogeneity of variance assumption

We used **Levene's test**, which is less sensitive to departures from normal distribution.

```
#Check validity of ANOVA test assumption
leveneTest(Global_Sales ~ Genre, data = newdata)
```

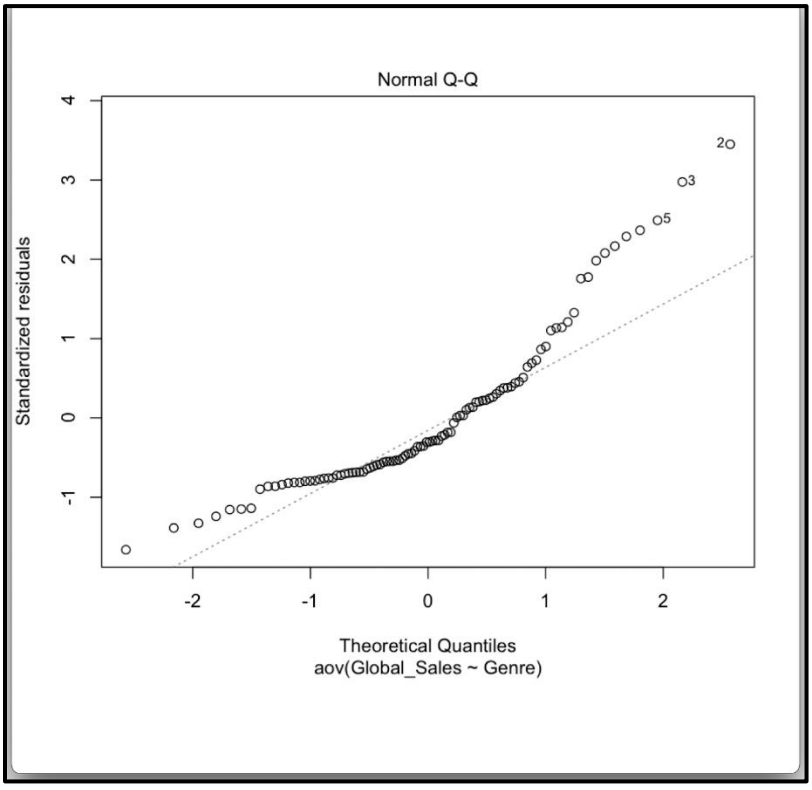
```
> leveneTest(Global_Sales ~ Genre, data = newdata)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group 10  0.8437 0.5882
      88
```

From the output above we can see that the p-value is not less than the significance level of 0.05 (0.5882>0.05). This means that there is no evidence to suggest that the variance across groups is statistically significantly different. Therefore, we can assume the homogeneity of variances in the different treatment groups.

Step 2: Check the normality assumption

```
plot(Anova_Result, 2)
```

Result:



As all the points fall approximately along this reference line, we can assume normality.

2.2.7. Kruskal – Wallis test

We want to know if there is any significant difference between the average sales of different game's genres.

Hypothesis:

- H_0: There is no significant difference between the average sales of different game's genres.
- H_1: There are significant differences between the average sales of different game's genres.

The test can be performed using the function `kruskal.test()` as follow:

```
#Kruskal-Wallis test
kruskal.test(Global_Sales ~ Genre, data = newdata)
```

Result:

```
> kruskal.test(Global_Sales ~ Genre, data = newdata)

    Kruskal-Wallis rank sum test

data:  Global_Sales by Genre
Kruskal-Wallis chi-squared = 2.4149, df = 10, p-value = 0.9921
```

As the p-value is more than the significance level (0.9921>0.05), we do not have enough evidence to conclude that there is significant difference between the average sales of different game's genres.

2.2.8. Linear regression

In order to use the linear regression, we need to use another type of dataset, so in the csv files of video games sales above, we will take the **Year_of_Release** and the **Global_Sales** column as our data in this section.

In this section, to do the linear model we use the command `lm`, and to plot the data and regression line, we use the command `plot` and `abline`.

Step 1: Prepare the data

We want to see if there is a linear relationship between Year_of_Release and the Global_Sales in the first 100 rows.

Assumption:

- H_0: The beta coefficient associated with the variables is equal to zero. (Or there does not exist a relationship between the Year_of_Release and the Global_Sales)
- H_1: The coefficient is not equal to zero. (Or there exists a relationship between the Year_of_Release and the Global_Sales)

R code:

```
###Step 1: Prepare the data
#import the data

data <- read.csv("E:\\R\\Video_Games.csv", sep = ",", header = T)
names(data) #to see what variable we want to use

linear_data = head(data[, c(3, 10)], 100)
#independent variable is Year_of_Release.
#dependent variable is Global_Sales.

#convert Year into numeric
linear_data$Year_of_Release <- as.integer(linear_data$Year_of_Release)
linear_data
```

Result:

```
>       data <- read.csv('/Users/ZEPHYRUS/Downloads/archive/Video_Games_Sales.csv', sep = ",", header =
 T)
>       names(data) #to see what variable we want to use
 [1] "Name"            "Platform"        "Year_of_Release" "Genre"           "Publisher"
 [6] "NA_Sales"        "EU_Sales"        "JP_Sales"        "Other_Sales"     "Global_Sales"
[11] "Critic_Score"    "Critic_Count"    "User_Score"      "User_Count"      "Developer"
[16] "Rating"
>       linear_data = head(data[, c(3, 10)], 100)
>       #convert Year into numeric
>       linear_data$Year_of_Release <- as.integer(linear_data$Year_of_Release)
>       linear_data
    Year_of_Release Global_Sales
1              2006        82.53
2              1985        40.24
3              2008        35.52
4              2009        32.77
5              1996        31.37
6              1989        30.26
7              2006        29.80
8              2006        28.92
9              2009        28.32
10             1984        28.31
11             2005        24.67
12             2005        23.21
13             1999        23.10
14             2007        22.70
15             2010        21.81
16             2009        21.79
17             2013        21.04
18             2004        20.81
19             1990        20.61
20             2005        20.15
21             2006        18.25
22             1989        18.14
23             1988        17.28
24             2013        16.27
25             2002        16.15
26             2002        15.85
27             2005        15.29
28             2010        15.14
29             2001        14.98
30             2011        14.73
31             1998        14.64
32             2015        14.63
33             2010        14.61
34             2013        14.60
35             2012        13.79
36             2012        13.67
37             2009        13.47
38             2011        13.32
39             2001        13.10
40             2008        12.84
41             2011        12.66
42             2010        12.63
43             2014        12.61
44             2005        12.13
```

Step 2: Do the linear regression

Plot the values using `plot(x, y).`

Then, we plot the regression line by command: `abline(lm(y ~ x))`

The command `lm(y ~ x)` shows the basic values of the regression analysis.

R code:

```
###Step 2: do the linear model
plot(linear_data$Year_of_Release, linear_data$Global_Sales)
abline(lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data))
linearMod <- lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data)
linearMod
```

Result:

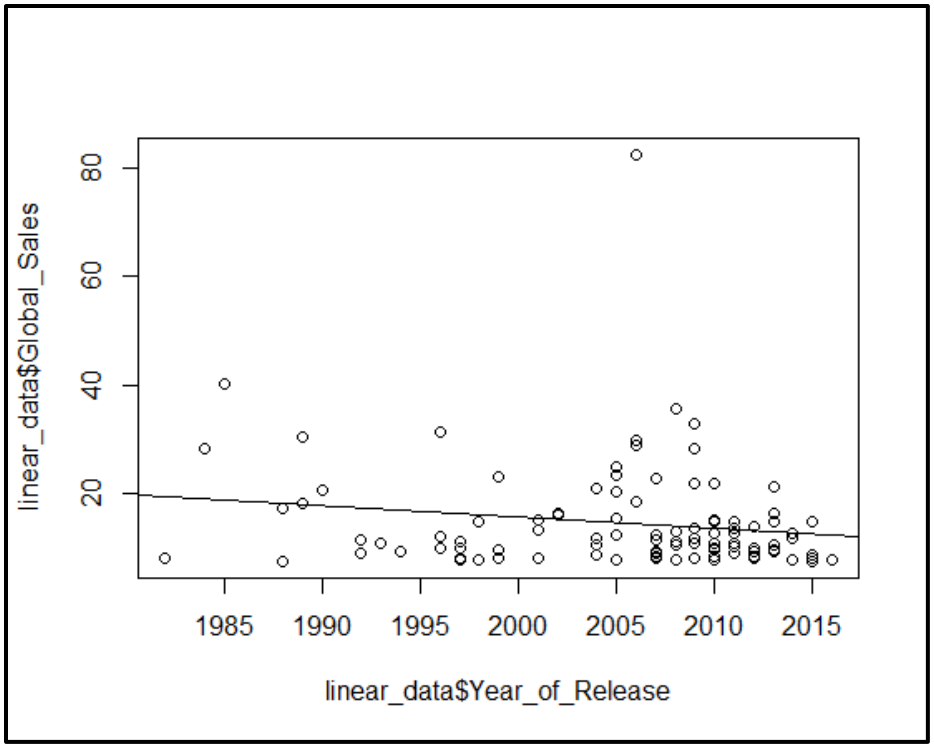
```
> ###Step 2: do the linear model
> plot(linear_data$Year_of_Release, linear_data$Global_Sales)
> abline(lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data))
> linearMod <- lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data)
> linearMod

Call:
lm(formula = linear_data$Global_Sales ~ linear_data$Year_of_Release,
    data = linear_data)

Coefficients:
                (Intercept)  linear_data$Year_of_Release
                   428.3666                      -0.2064
```

It gives us two values of a and b which is two coefficient of the regression line:

$$y = -0.2064x + 428.3666$$



Step 3: Conclusion.

Now the linear model is built and we have a formula that we can use to predict the **Global_Sales** value if a corresponding **Year_of_release** is known.

But it is not enough to use this model. Because, before using a regression model to make predictions, we need to ensure that it is statistically significant.

To ensure that, we print the summary statistic:

R code:

```
#Check the model
summary(linearMod)
```

Result:

```
Residuals:
    Min      1Q  Median      3Q     Max
-11.562  -5.183  -2.879   1.393  68.110

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)               428.3666   248.5830   1.723   0.0880
linear_data$Year_of_Release  -0.2064     0.1240  -1.665   0.0992
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.811 on 98 degrees of freedom
Multiple R-squared:  0.0275,    Adjusted R-squared:  0.01757
F-statistic: 2.771 on 1 and 98 DF,  p-value: 0.0992
```

As we can see, the model's p_value is 0.0992 and the p_value of individual predictor variables is 0.0880. Both of them is greater than the pre-determined statistical significance level of 0.05. So we does not reject H_0.

As a result, this linear model can not be considered to be statistically significant. At $\alpha = 0.05$, we do not have enough information to conclude that there exists a relationship between the Year_of_Release and the Global_Sales.

2.3. Code implementation

```
#Project 2 Exercise 2
#Resource: https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings
library(car)
#Import data
data <- read.csv(file = '/Users/ZEPHYRUS/Downloads/archive/Video_Games_Sales.csv', sep
= ",",header = T, nrows = 100)
#find the head's number to get choosen data
names(data)
newdata <- data[,c(4,10)]
#eliminate line 1 to use ANOVA-One-way test
newdata <- newdata[-c(1),]
newdata
#Data cleaning
newdata <- na.omit(newdata)
#min and max of global sales
rng <- range(newdata$Global_Sales)
#min
rng[1]
#max
rng[2]
#range of global sales
rng[2] - rng[1]
#mean
mean(newdata$Global_Sales)
#median
median(newdata$Global_Sales)
#the standard deviation
sd(newdata$Global_Sales)
#the standard variance
var(newdata$Global_Sales)
#modify margines of boxplot
par(mai=c(2.0, 0.8, 0.8, 0.4))
#draw boxplot
```

```
boxplot(newdata$Global_Sales ~ newdata$Genre, las = 2, xlab = "", ylab = "Global Sales")
title(xlab = "Genre", line = 5);
#One-way ANOVA test
Anova_Result = aov(Global_Sales ~ Genre, data = newdata)
summary(Anova_Result) #Show Anova result
#Get F_value from Anova_Result
F_Value_Genre <- (summary(Anova_Result))[[1]][["F value"]]
F_value <- F_Value_Genre[1]
F_value #F_value that used to compare
df <- (summary(Anova_Result))[[1]][["Df"]] #get df from Anova_Result
df_b = df[1] #df between
df_w = df[2] #df within
df_b
df_w
F_critical = qf(0.95, df_b, df_w) #search F_critical from F distribution
F_critical
#Conclusion
if(F_value >= F_critical) {
    print("At alpha = 0.05, we have enough evidence to conclude that")
    print(" there is a difference between Global_sales in different Game Genre")
}else{
    print("At alpha = 0.05, We do not have enough evidence to conclude that")
    print(" there is a difference between Global_sales in different Game Genre")
}
#Multiple pair comparison
TukeyHSD(Anova_Result)
#Check validity of ANOVA test assumption
leveneTest(Global_Sales ~ Genre, data = newdata)
plot(Anova_Result, 2)
#Kruskal-Wallis test
kruskal.test(Global_Sales ~ Genre, data = newdata)
```

```
#Linear regression model:
    ###Step 1: Prepare the data
    #import the data

    data <- read.csv('/Users/ZEPHYRUS/Downloads/archive/Video_Games_Sales.csv', sep = ",", header = T)
    names(data) #to see what variable we want to use

    linear_data = head(data[, c(3, 10)], 100)
    #independent variable is Year_of_Release.
    #dependent variable is Global_Sales.

    #convert Year into numeric
    linear_data$Year_of_Release <- as.integer(linear_data$Year_of_Release)
    linear_data

    ###Step 2: do the linear model
    plot(linear_data$Year_of_Release, linear_data$Global_Sales)
    abline(lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data))
    linearMod <- lm(linear_data$Global_Sales ~ linear_data$Year_of_Release, data = linear_data)
    linearMod
    #Check the model
    summary(linearMod)
```