APH101-Biostatistics And R

Yu.Lu23

2025-06-02

Table of contents

Final review of statistics knowledge

Three main sampling distributions in hypothesis testing

Chi-square distribution

Suppose $X_1, \cdots, X_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$.the distribution of the statistic

$$X_1^2 + \cdots + X_n^2$$

is called a chi-square distribution with n degrees of freedom, denoted by $\chi^2(n)$.

Besides, random variable $X_i^2 \sim \text{Gamma}\left(\frac{1}{2}, \frac{1}{2}\right)$ corresponds to the chi-squared distribution with 1 degree of freedom, denoted as χ_1^2.

This is derived by the MGF:

Since

$$M_{X_1^2+\cdots+X_n^2}(t) = M_{X_1^2}(t) \times \cdots \times M_{X_n^2}(t) = \begin{cases} \infty & t \geq \frac{1}{2} \\ (1-2t)^{-\frac{n}{2}} & t < \frac{1}{2} \end{cases}$$

This is the MGF of the Gamma $\left(\frac{n}{2}, \frac{1}{2}\right)$ distribution, so $X_1^2 + \cdots + X_n^2 \sim \text{Gamma}\left(\frac{n}{2}, \frac{1}{2}\right)$. This is called the chi-squared distribution with **n** degree of freedom, denoted χ_n^2.

Properties

- If $W_1, \ldots, W_n$ are independent χ^2 random variables with, respectively, $v_1, \cdots, v_n$ degrees of freedom, then the random variable $W_1 + \cdots + W_n$ follows a χ^2-distribution with $v_1 + \cdots + v_n$ degree of freedom.
- The random variable $\frac{(\bar{X}-\mu)^2}{\sigma^2/n}$ follows a χ^2-distribution with 1 degree of freedom when X follows a normal distribution with mean μ and variance σ^2.

Application

Chi-square distribution is primarily used in testing:

- Goodness-of-fit
- Independence in contingency tables

Student's t-distribution

Construction

The statistic $T = \frac{\bar{X}-\mu}{S/\sqrt{n}}$ follows a t-distribution with $v = n-1$ degrees of freedom when $X_1, \cdots, X_n$ are i.i.d. normal RVs.

$$\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right) \quad \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \sim \mathcal{N}(0,1) \quad \frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1}$$

If we know the population variance σ^2, we can easily do inference using the statistic $\frac{\bar{X}-\mu}{\sigma/\sqrt{n}}$. However, σ^2 is usually unknown in practice.

$$\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right) \quad \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \sim \mathcal{N}(0,1) \quad \frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1}$$

We can construct the t-statistic using the sample variance S^2 :

$$T = \frac{\frac{\bar{X}-\mu}{\sigma/\sqrt{n}}}{\sqrt{\frac{(n-1)s^2}{\sigma^2}/(n-1)}} = \frac{\bar{X}-\mu}{S/\sqrt{n}}$$

Notice the sample mean $\bar{X}$ and the sample variance S^2 are independent (the proof is beyond the scope of this course). So the T is now a ratio of a standard normal variable and the square root of a χ^2RV divided by its degrees of freedom. This is the definition of a t-distribution with $n-1$ degrees of freedom.

Properties

The t-distribution is primarily used in contexts where the underlying population is assumed to be normally distributed, especially when the sample size is small. Used extensively in problems that deal with inference about population mean μ when population variance σ^2 is unknown; problems where one is trying to determine if means from two samples are significantly different when population variances σ_1^2 and σ_2^2 are unknown.

F-distribution

Let U and V be two independent random variables following χ^2 distributions with ν_1 and ν_2 degrees of freedom, respectively. Then the distribution of the random variable $F = \frac{U/\nu_1}{V/\nu_2}$ is known as F-distribution.

Example

If S_1^2 and S_2^2 are the variances of independent RVs of size n_1 and n_2 taken from normal populations with variances σ_1^2 and σ_2^2 respectively, then

$$F = \frac{S_1^2/\sigma_1^2}{S_2^2/\sigma_2^2} = \frac{\sigma_2^2 S_1^2}{\sigma_1^2 S_2^2}$$

follows an F-distribution with $\nu_1 = n_1 - 1$ and $\nu_2 = n_2 - 1$ degrees of freedom.

Estimation of Population Characteristic

Point Estimation

A point estimate of a population characteristic is a single number that is based on sample data and represents a plausible value of the characteristic.

Interval Estimation—Confidence interval(CI)

An interval estimate of a parameter θ is an interval of the form $\hat{\theta}_L < \theta < \hat{\theta}_U$, where $\hat{\theta}_L$ and $\hat{\theta}_U$ depend on the value of $\hat{\theta}$ for a particular sample and also on the sampling distribution of $\hat{\Theta}$.

- If we were to construct a 95% confidence interval for some population characteristics (population proportion p or population mean μ), we would be using a method that is successful 95% of the time.
- This is also about the question relevant to "How to choose a sample size"

Definition

A $100(1-\alpha)\%$ confidence interval is an interval of the form $\hat{\theta}_L < \theta < \hat{\theta}_U$, where $\hat{\theta}_L$ and $\hat{\theta}_U$ are respectively values of $\widehat{\Theta}_L$ and $\widehat{\Theta}_U$ obtained for a particular sample, based on

$$P\left(\widehat{\Theta}_L < \theta < \widehat{\Theta}_U\right) = 1-\alpha \quad ; \quad 0 < \alpha < 1$$

in the estimation of population parameter θ.

Interpretation

For confidence level of 95% CI for any normal distribution: About 95% of the values are within 1.96 standard deviations of the mean. (Recall the concept of Z-scores)

That is, if

$$\text{Estimate} \pm (Z \times \sigma)$$

was used to generate an interval estimate over and over again with different samples, in the long run 95% of the resulting intervals would include the actual value of the characteristic being estimated.

The confidence level 95% refers to the method used to construct the interval rather than to any particular interval, such as the one we obtained.

CI on Mean

Here, $\bar{x}$ is the sample mean from a simple random sample.

μ is the population mean which we are interested in estimating.

CI on μ with σ known n ≥ 30 or the population is normal — Use z-statistics

CI: $\bar{x} \pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$, for example, 95% CI is $\bar{x} \pm 1.96\frac{\sigma}{\sqrt{n}}$

One-side Confidence Bound on μ with σ known n ≥ 30 or the population is normal — Use z-statistics

Upper one-side bound: $\mu < \bar{x} + z_{\alpha}\frac{\sigma}{\sqrt{n}}$

Lower one-side bound: $\mu > \bar{x} - z_{\alpha}\frac{\sigma}{\sqrt{n}}$

For example, 95% Confidence bound on μ is $\bar{x} \pm 1.645\frac{\sigma}{\sqrt{n}}$

CI on μ with σ unknown and the population is normal — Use t-statistics (use s as the estimate for (t-statistics with df = n-1))

CI: $\bar{x} \pm t_{\alpha/2}\frac{s}{\sqrt{n}}$, for example, 95% CI is $\bar{x} \pm t_{0.025}\frac{s}{\sqrt{n}}$ and df = n-1

Remark: The distribution of t is more spread out than the standard normal distribution but when n $\geq$ 30, t and z are very close to each other.

CI for $\mu_1 - \mu_2$, both σ_1^2 and σ_2^2 are known

CI of $\mu_1 - \mu_2$: $(\bar{x}_1 - \bar{x}_2) \pm z_{\alpha/2}\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$

CI for $\mu_1 - \mu_2$, both σ_1^2 and σ_2^2 are unknown but assumed equal

CI of $\mu_1 - \mu_2$: $(\bar{x}_1 - \bar{x}_2) \pm t_{\alpha/2} s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$ with df $= n_1 + n_2 - 2$ where $s_p = \sqrt{\frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}}$

CI for paired observations

Previous, we have two independent samples, now we have two dependent samples. We can use the difference between the two samples to construct a confidence interval.

CI of μ_d: $\bar{d} \pm t_{\alpha/2}\frac{s_d}{\sqrt{n}}$ with df = n-1 where s_d is the sample standard deviation of the differences $d_i = x_{1i} - x_{2i}$ and $\bar{d}$ is the sample mean of the differences.

CI for Estimating σ

a $100(1-\alpha)\%$ CI for σ^2 is $\left(\frac{(n-1)S^2}{\chi^2_{\alpha/2,n-1}}, \frac{(n-1)S^2}{\chi^2_{1-\alpha/2,n-1}}\right)$

where S^2 is the sample variance and $\chi^2_{\alpha/2,n-1}$ and $\chi^2_{1-\alpha/2,n-1}$ are the critical values of the chi-square distribution with $n-1$ degrees of freedom.

Estimating σ_1^2/σ_2^2

A $100(1-\alpha)\%$ CI for $\frac{\sigma_1^2}{\sigma_2^2}$ using F-statistics with $f_{1-\alpha/2}(n_1-1, n_2-1) = 1/f_{\alpha/2}(n_1-1, n_2-1)$ is

$$\frac{s_1^2}{s_2^2}\frac{1}{f_{\alpha/2}(n_1-1, n_2-1)} < \frac{\sigma_1^2}{\sigma_2^2} < \frac{s_1^2}{s_2^2} f_{\alpha/2}(n_2-1, n_1-1)$$

where $f_{\alpha/2}(v_1, v_2)$ is an F-value with v_1 and v_2 degrees of freedom, leaving an area of $\alpha/2$ to the right, and $f_{\alpha/2}(v_2, v_1)$ is a similar F-value with v_2 and v_1 degrees of freedom.

Hypothesis Testing

- z-test

Suppose $X_1, \cdots, X_n \xrightarrow[\sim]{\text{iid}} \mathcal{N}(\mu, \sigma^2)$, where μ is unknown and where $\sigma^2 = \sigma_0^2$ is known.

Suppose we wish to test $H_0 : \mu = \mu_0$ against $H_1 : \mu > \mu_0$. Then we can use the test statistic

$$Z = \frac{\bar{X} - \mu_0}{\sigma_0/\sqrt{n}}$$

If H_0 is true then $Z \sim \mathcal{N}(0,1)$. Let

$$z_{\text{obs}} = \frac{\bar{x} - \mu_0}{\sigma_0/\sqrt{n}}$$

A large value of z_{obs} casts doubt on the validity of H_0 and indicates a departure from H_0 in the direction of H_1. So the p-value for testing H_0 against H_1 is

$$\begin{aligned} p &= P\left(Z \geq Z_{\text{obs}} \mid H_0\right) \\ &= P\left(\mathcal{N}(0,1) \geq Z_{\text{obs}}\right) \\ &= 1 - \Phi\left(Z_{\text{obs}}\right) \end{aligned}$$

The z-test of $H_0 : \mu = \mu_0$ against the alternative $H_1 : \mu < \mu_0$ is similar but this time a small, i.e. very negative, value of Z_{obs} casts doubt on H_0. So the p-value is

$$\begin{aligned} p &= P\left(Z \leq Z_{\text{obs}} \mid H_0\right) \\ &= P\left(\mathcal{N}(0,1) \leq Z_{\text{obs}}\right) \\ &= \Phi\left(Z_{\text{obs}}\right) \end{aligned}$$

Finally, consider testing $H_0 : \mu = \mu_0$ against the alternative $H_1 : \mu \neq \mu_0$. Let $z_0 = |z_{\text{obs}}|$. A large value of z_0 indicates a departure from H_0, so the p-value is

$$\begin{aligned} p &= P\left(|Z| \geq z_0 \mid H_0\right) \\ &= P\left(\mathcal{N}(0,1) \geq z_0\right) + P\left(\mathcal{N}(0,1) \leq -z_0\right) \\ &= 2\left(1 - \phi\left(z_0\right)\right) \end{aligned}$$

- t-test

We can use the test statistic

$$T = \frac{\bar{X} - \mu_0}{S/\sqrt{n}}$$

If H_0 is true then $T \sim t_{n-1}$. Let $t_{\text{obs}} = t(x) = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$ and $t_0 = |t_{\text{obs}}|$. Similarly, we have $\square$ the p-value is $P\left(t_{n-1} \geq t_{\text{obs}}\right)$ - the p-value is $P\left(t_{n-1} \leq t_{\text{obs}}\right)$ the p-value is $2P\left(t_{n-1} \geq t_0\right)$

Example of one-sample t-test

A marine biologist is studying a species of fish known to have an average length of 20 cm in ocean populations. A new population in a freshwater lake is being analyzed to determine if the environmental differences have altered the fish's average length. The biologist measures the lengths of 10 randomly selected fish, yielding the following data:

22, 23, 21, 24, 22, 20, 25, 19, 23, 22

Assuming the data satisfy the assumption of normality, please address the following using a significance level of 0.1:

a

- null hypothesis: The mean length of fish is 20 cm ($H_0 : \mu = 20$).
- alternative hypothesis: The mean length of fish is not 20 cm ($H_1 : \mu \neq 20$).

b

Since the data is supposed to be normally distributed, the sampling distribution of the sample mean follows t-distribution. The t-test statistic is calculated as follows:

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

The t-test statistic is calculated as follows:

```
data2_4 <- c(22, 23, 21, 24, 22, 20, 25, 19, 23, 22)

x_bar <- mean(data2_4)

s <- sd(data2_4)

t <- (x_bar - 20) / (s / sqrt(length(data2_4)))

t
```

```
[1] 3.705882
```

The t-test statistic is 3.705882...

c

Using pt() function, the p-value is calculated as follows:

```
p_value <- 2 * pt(-t, df = 9)
p_value
```

```
[1] 0.004875954
```

The p-value is approximately 0.1. Since the p-value is less than 0.1, we reject the null hypothesis.

Therefore, there is sufficient evidence to conclude that the population mean is not equal to 20 which means the environmental differences have altered the fish's average length.

d

Using qt() function to find the critical value for a two-tailed test with 90% confidence level:

```
t_critical <- qt(0.95, df = 9)
t_critical
```

```
[1] 1.833113
```

The critical value for a two-tailed test with 90% confidence level is about 1.833113.

Since the t-test statistic 3.705882 is greater than the critical value 1.833113, which is in the critical region. Therefore, we reject the null hypothesis.

Also, we could use confidence interval to verify the result. The 90% confidence interval for the population mean is calculated as follows:

```
ci_4 <- c(x_bar - t_critical * s / sqrt(10), x_bar + t_critical * s / sqrt(10))
ci_4
```

```
[1] 21.06124 23.13876
```

So the 90% confidence interval for the population mean is about (21.1, 23.2).

The confidence interval does not contain the hypothesized value 20. Therefore, we reject the null hypothesis.

Therefore, there is sufficient evidence to conclude that the population mean is not equal to 20 which means the environmental differences have altered the fish's average length.

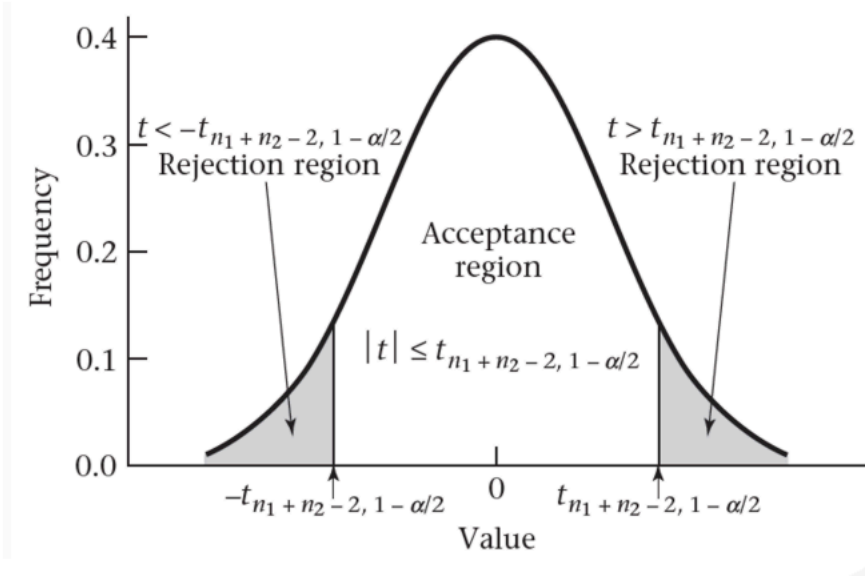
Example of unpaired t-test

Suppose σ_1^2 and σ_2^2 are unknown but assumed equal. We want to test the null hypothesis $H_0 : \mu_1 = \mu_2$ against the alternative hypothesis $H_1 : \mu_1 \neq \mu_2$. The test statistic is given by

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where s_p is the pooled sample standard deviation, given by

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$$



Example of one-sample Variance Test

χ^2-test for variance. Suppose $X_1, \cdots, X_n$ are i.i.d. normal random variables with mean μ and variance σ^2. We want to test the null hypothesis $H_0 : \sigma^2 = \sigma_0^2$ against the alternative hypothesis $H_1 : \sigma^2 \neq \sigma_0^2$. The test statistic is given by

$$\chi^2 = \frac{(n-1)s^2}{\sigma^2}$$

Example of two-sample Variance Test

H_0	Test Statistic	H_1	Rejection Region
$\sigma_1^2 = \sigma_2^2$	$f = \frac{s_1^2}{s_2^2}$	$\sigma_1^2 < \sigma_2^2$	$f < f_{\alpha}(\nu_1, \nu_2)$
		$\sigma_1^2 > \sigma_2^2$	$f > f_{1-\alpha}(\nu_1, \nu_2)$
	$\sigma_1^2 \neq \sigma_2^2$	$f < f_{\alpha/2}(\nu_1, \nu_2)$ or $f > f_{1-\alpha/2}(\nu_1, \nu_2)$	
$\nu_1 = n_1 - 1$ and $\nu_2 = n_2 - 1$ are two degree of freedom.			

$$F = \frac{\frac{(n_1-1)S_1^2}{\sigma_1^2} / (n_1 - 1)}{\frac{(n_2-1)S_2^2}{\sigma_2^2} / (n_2 - 1)} = \frac{\sigma_2^2 S_1^2}{\sigma_1^2 S_2^2}$$

If $\sigma_1^2 = \sigma_2^2$, we have

$$F = \frac{S_1^2}{S_2^2} \sim F_{n_1-1, n_2-1}$$

ANOVA– Analysis of Variance

- one-way ANOVA

we need to test the null hypothesis that the group population means are all the same against the alternative that at least one group population mean differs from the others. That is,

$H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ against $H_1 :$ at least one μ_i differs from the others.

ANOVA Table

Source	DF	Sum Sq	Mean Sq	F value	p value
Factor	m-1	11.84 (SS between)	2.9587 (MSB)	8.074 (MSB/MSW)	$5.38e-05$ (p-value)
Error	n-m	16.49 (SS Within)	0.3664 (MSW)		
Total	n-1	28.33 (SS Total)			

Source means "the source of the variation in the data." the possible sources for a one-factor study are Factor, Residuals, and Total.

Factor means "the variability due to the factor of interest." In the drug example, the factor was the different drug. In the learning example on the previous page, the factor was the method of learning. Sometimes the row heading is labeled as Between.

Error (or Residuals) means "the variability within the groups" or "unexplained random error." Sometimes the row heading is labeled as Within.

Total means "the total variation in the data from the grand mean".

DF means "the degrees of freedom in the source."

Sum Sq means "the sum of squares due to the source."

Mean Sq means "the mean sum of squares due to the source."

F value means "the F-statistic."

P value means "the P-value."

SS(Total)=SS(Between)+SS(Within), where

SS(Between) is the sum of squares between the group means and the grand mean. As the name suggests, it quantifies the variability between the groups of interest.

SS(Within) is the sum of squares between the data and the group means. It quantifies the variability within the groups of interest.

SS(Total) is the sum of squares between the n data points and the grand mean. As the name suggests, it quantifies the total variability in the observed data.

- two-way ANOVA

We can extend the idea of a one-way ANOVA, which tests the effects of one factor on a response variable, to a two-way ANOVA which tests the effects of two factors and their interaction on a response variable.

Source	DF	Sum Sq	MSW	F
Cells	$ab-1$	$\sum_{i=1}^{a}\sum_{j=1}^{b} n\left(\bar{X}_{ij}-\bar{X}_{\cdot\cdot\cdot\cdot}\right)^2$		
A	a-1	bn $\sum_{i=1}^{a}\left(\bar{X}_{i\cdot\cdot}-\bar{X}_{\cdot\cdot\cdot\cdot}\right)^2$	SS(A)	MS(Error
B	b-1	an $\sum_{j=1}^{b}\left(\bar{X}_{\cdot j\cdot}-\bar{X}_{\cdot\cdot\cdot\cdot}\right)^2$	SS(B)	MS(B)
A × B	$(a-1)(b-1)$	SS(Cells)-SS(A)-SS(B)	SS(AB)	MS(Error
			DF(A × B)	MS(Error
Error	$ab(n-1)$	$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{l=1}^{n}\left(X_{ijl}-\bar{X}_{ij\cdot}\right)^2$	SS(Error)	
Total	$abn-1$	$\sum_{i=1}^{a}\sum_{j=1}^{b} n\left(X_{ijl}-\bar{X}_{\cdot\cdot\cdot\cdot}\right)^2$		

- $F = \frac{\text{MS(A)}}{\text{MS(Error)}}$, for H_0 : no effect of factor A on response variable,
- $F = \frac{\text{MS(B)}}{\text{MS(Error)}}$, for H_0 : no effect of factor B on response variable,
- $F = \frac{\text{MS(A}\times\text{B)}}{\text{MS(Error)}}$, for H_0 : no effect of interaction on response variable.

We reject any H_0 if $F \geq F_{\text{critical}}$; otherwise, we do not reject H_0.

Example of two-way ANOVA

Two-way ANOVA. In this question, we will use the built-in R data set ToothGrowth to perform two-way ANOVA test. ToothGrowth includes information from a study on the effects of vitamin C on tooth growth in Guinea pigs. The response is the length of odontoblasts (cells responsible for tooth growth) in 60 guinea pigs. Each animal received one of three dose levels of vitamin C (0.5, 1, and 2 mg/day) by one of two delivery methods, orange juice or ascorbic acid (a form of vitamin C and coded as VC). Assuming the data satisfy the assumptions of normality and equal variance, please address the following using a significance level of 0.05

a

- The effects of vitamin C on tooth growth in guinea pigs:
- null hypothesis: H_0: mean tooth growth for all doses of vitamin C are equal
- alternative hypothesis: H_1: at least one of the means of all doses of vitamin C is different from the others
- The effects of delivery method on tooth growth in guinea pigs:
- null hypothesis: H_0: mean tooth growth for the delivery method of orange juice and ascorbic acid are equal.
- alternative hypothesis H_1: mean tooth growth for the delivery method of orange juice and ascorbic acid are different.
- The interaction effects of the dose of vitamin C and delivery method on tooth growth in guinea pigs:
- null hypothesis: H_0 : there is no interaction between the dose of vitamin C and delivery method on tooth growth in guinea pigs, meaning that the relationship between vitamin C and tooth growth is the same for both delivery methods (similarly, the relationship between delivery method and tooth growth is the same for all doses of vitamin C).

- alternative hypothesis: H_1: there is an interaction between the dose vitamin C and delivery method on tooth growth in guinea pigs, meaning that the relationship between vitamin C and tooth growth is different for both delivery methods (similarly, the relationship between delivery method and tooth growth depends on the dose of vitamin C).

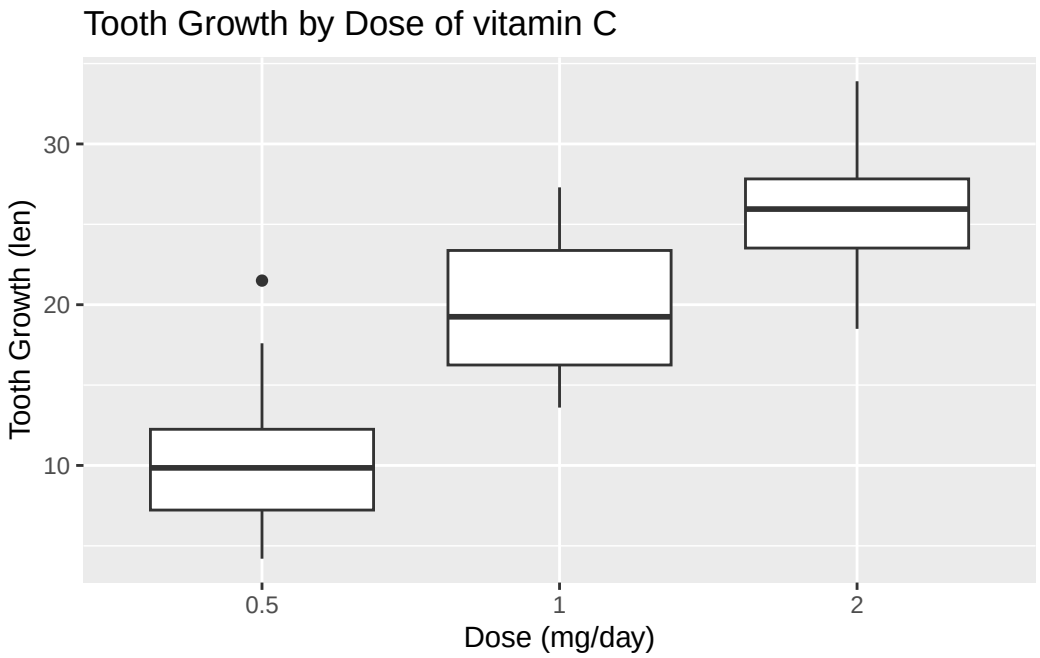
b

We can plot the relationship one by one using two plots

```
library(ggplot2)
data(ToothGrowth)
head(ToothGrowth)
```

```
   len supp dose
1  4.2   VC  0.5
2 11.5   VC  0.5
3  7.3   VC  0.5
4  5.8   VC  0.5
5  6.4   VC  0.5
6 10.0   VC  0.5
```

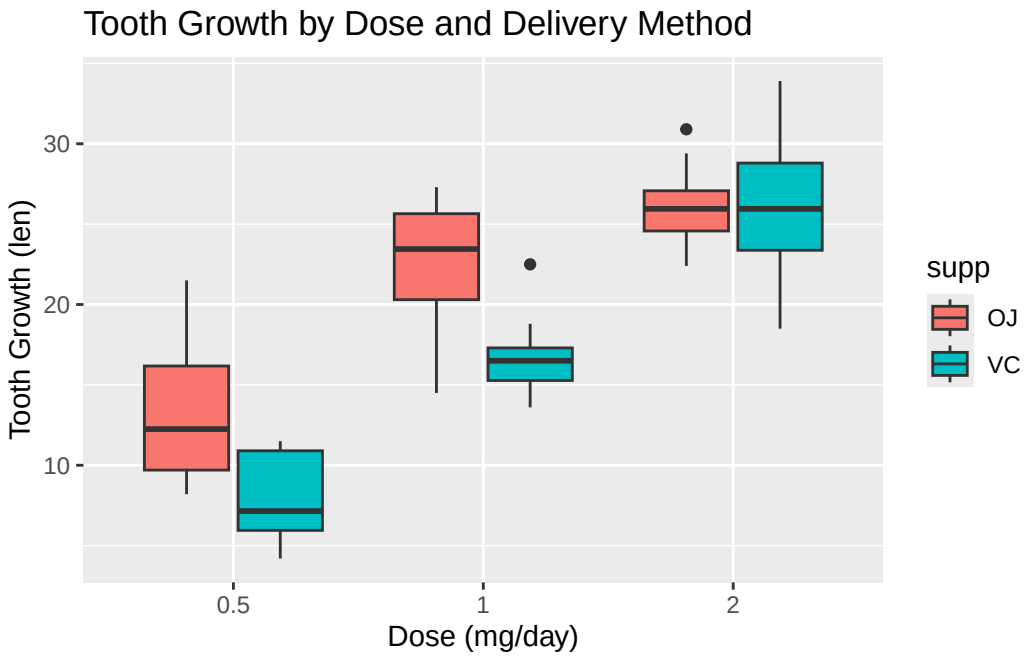
```
# potential effects of vitamin C on tooth growth.

ggplot(ToothGrowth, aes(x = factor(dose), y = len)) +
  geom_boxplot() +
  labs(x = "Dose (mg/day)", y = "Tooth Growth (len)", title = "Tooth Growth by Dose of vitam
```




```
# potential effects of delivery method on tooth growth.

ggplot(ToothGrowth, aes(x = supp, y = len)) +
  geom_boxplot() +
  labs(x = "Delivery Method", y = "Tooth Growth (len)", title = "Tooth Growth by Delivery Met
```

or just one:

```
library(ggplot2)

# potential effects of vitamin C and delivery method.

# OJ represents orange juice and VC represents ascorbic acid.

ggplot(ToothGrowth, aes(x = factor(dose), y = len, fill = supp)) +
  geom_boxplot() +
  labs(x = "Dose (mg/day)", y = "Tooth Growth (len)", title = "Tooth Growth by Dose and Deliv
```



c

```
# Perform two-way ANOVA

anova_result <- aov(len ~ supp * dose, data = ToothGrowth)

summary(anova_result)
```

```
             Df Sum Sq Mean Sq F value   Pr(>F)
supp          1  205.4   205.4  12.317 0.000894 ***
```

```
dose          1 2224.3  2224.3 133.415  < 2e-16 ***
supp:dose     1   88.9    88.9   5.333 0.024631 *
Residuals    56  933.6    16.7
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

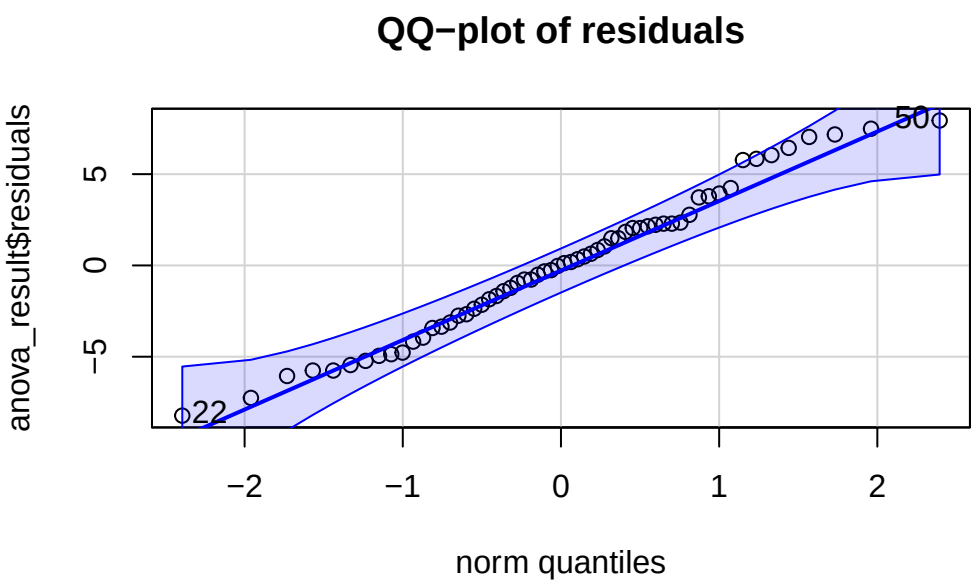
Since all p-values are less than 0.05, we reject all null hypotheses. Therefore, there is sufficient evidence to conclude that the dose of vitamin C, delivery method, and their interaction have significant effects on tooth growth in guinea pigs.

d

```
library(car)
```

```
Loading required package: carData
```

```
qqPlot(anova_result$residuals, main = "QQ-plot of residuals")
```




```
[1] 22 50
```

Non-parametric tests

Application of testing the goodness of fit

Testing whether there is a "good fit" between the observed data and the assumed probability model amounts to testing:

Construction of test statistics with an example of 2 categories

Population is 60% female and 40% male. Then, if a sample of 100 students yields 53 females and 47 males, can we conclude that the sample is (random and) representative of the population? That is, how "good" do the data "fit" the assumed probability model of 60% female and 40% male?

Here, let Y_1 denote the number of females selected, $Y_1 \sim B(n, p_1)$ and let Y_2 denote males selected, $Y_2 = (n - Y_1) \sim B(n, p_2) = B(n, 1 - p_1)$.

for samples satisfying the general rule of thumb (the expected number of successes must be at least 5 and the expected number of failures must be at least 5), we can use the normal approximation to the binomial distribution. The test statistic is given by

$$Z = \frac{Y_1 - np_1}{\sqrt{np_1(1-p_1)}} \sim N(0,1)$$

which is at least approximately normally distributed.

and

$$Z^2 = Q_1 = \frac{(Y_1 - np_1)^2}{np_1(1-p_1)} \sim \chi^2(1)$$

which is an approximate chi-square distribution with one degree of freedom.

Now we can multiply Q_1 by $1 = (1 - p_1) + p_1$ to get

$$Q_1 = \frac{(Y_1 - np_1)^2(1-p_1)}{np_1(1-p_1)} + \frac{(Y_1 - np_1)^2 p_1}{np_1(1-p_1)} \sim \chi^2(1)$$

Since $Y_1 = n - Y_2$ and $p_1 = 1 - p_2$, after simplifying, we have

$$Q_1 = \frac{(Y_1 - np_1)^2}{np_1} + \frac{(-(Y_2 - np_2))^2}{np_2} \sim \chi^2(1)$$

which is $Q_1 = \sum_{i=1}^{2} \frac{(Y_i - np_i)^2}{np_i} = \sum_{i=1}^{2} \frac{(\text{ Observed } - \text{ Expected })^2}{\text{Expected}} \sim \chi^2(1)$

Hence, it is observed that if the observed counts are very different from the expected counts, then the test statistic will be large. So we reject the null hypothesis if Q_1 is large and how large is large is determined by the critical value of the chi-square distribution with one degree of freedom.

The statistics Q_1 is called the chi-square goodness of fit statistic.

Going back to the example,

- H_0:$p_F = 0.6$
- H_1:$p_F \neq 0.6$

we can calculate the test statistic using a significant level of $\alpha = 0.05$ ($\chi^2_{0.05,1} = 3.84$)as follows:

$$Q_1 = \frac{(53-60)^2}{60} + \frac{(47-40)^2}{40} = 2.04$$

Since $Q_1 = 2.04 < 3.84$, we do not reject the null hypothesis. Therefore, we conclude that the sample is (random and) representative of the population.

This can be extended to k categories

Construction of test statistics with an example of k categories

For categories more than 2, i.e.

Categories	1	2	$\cdots$	$k-1$	k
Observed	Y_1	Y_2	$\cdots$	Y_{k-1}	$n - Y_1 - Y_2 - \cdots - Y_{k-1}$
Expected	np_1	np_2	$\cdots$	np_{k-1}	np_k

Karl Pearson showed that the chi-square statistic Q_{k-1} defined as:

$$Q_{k-1} = \sum_{i=1}^{k} \frac{(Y_i - np_i)^2}{np_i}$$

follows approximately a chi-square random variable with $k-1$ degrees of freedom. Let's try it out on an example.

- Example:

Categories	Brown	Yellow	Orange	Green	Coffee	Total
Observed y_i	224	119	130	48	59	580
Assumed $H_0\,(p_i)$	0.4	0.2	0.2	0.1	0.1	1.0
Expected np_i	232	116	116	58	58	580

$Q_4 = \frac{(224-232)^2}{232} + \frac{(119-116)^2}{116} + \frac{(130-116)^2}{116} + \frac{(48-58)^2}{58} + \frac{(59-58)^2}{58} = 3.784$

Because there are $k = 5$ categories, we have to compare our chisquare statistic Q_4 to a chi-square distribution with $k - 1 = 5 - 1 = 4$ degrees of freedom:

$$Q_4 = 3.784 < \chi^2_{4,0.05} = 9.488$$

we fail to reject the null hypothesis.

Application of testing for homogeneity

This is to look at a method for testing whether two or more multinomial distributions are equal.

- Example:

Test the hypothesis that the acceptances of males and females are ditributed equally among the four schools,

(Acceptances)	Bus	Eng	L Arts	Sci	(FIXED) Total
Male	240 (20%)	480 (40%)	120 (10%)	360 (30%)	1200
Female	240 (30%)	80 (10%)	320 (40%)	160 (20%)	800
Total	480 (24%)	560 (28%)	440 (22%)	520 (26%)	2000

Here,

$H_0 : p_{MB} = p_{FB}, p_{ME} = p_{FE}, p_{ML} = p_{FL}$, and $p_{MS} = p_{FS}$

$H_1 : p_{MB} \neq p_{FB}$ or $p_{ME} \neq p_{FE}$ or $p_{ML} \neq p_{FL}$, or $p_{MS} \neq p_{FS}$

where:

- p_{Mj} is the proportion of males accepted into school $j = B, E, L, S$.
- p_{Fj} is the proportion of females accepted into school $j = B, E, L, S$.

In conducting such a hypothesis test, we're comparing the proportions of two multinomial distributions.

#(Acc)	Bus ($j=1$)	Eng ($j=2$)	L Arts ($j=3$)	Sci ($j=4$)	(FIXED) Total
M($i=1$)	$y_{11}\,(\hat{p}_{11})$	$y_{12}\,(\hat{p}_{12})$	$y_{13}\,(\hat{p}_{13})$	$y_{14}\,(\hat{p}_{14})$	$n_1 = \sum_{j=1}^{k} y_{1j}$
F ($i=2$)	$y_{21}\,(\hat{p}_{21})$	$y_{22}\,(\hat{p}_{22})$	$y_{23}\,(\hat{p}_{23})$	$y_{24}\,(\hat{p}_{24})$	$n_2 = \sum_{j=1}^{k} y_{2j}$
Total	$y_{11}+y_{21}\,(\hat{p}_1)$	$y_{12}+y_{22}\,(\hat{p}_2)$	$y_{13}+y_{23}\,(\hat{p}_3)$	$y_{14}+y_{24}\,(\hat{p}_4)$	n_1+n_2

The chi-square test statistic for testing the equality of two multinomial distributions:

$$Q=\sum_{i=1}^{2}\sum_{j=1}^{k}\frac{\left(y_{ij}-n_i\hat{p}_j\right)^2}{n_i\hat{p}_j}$$

follows an approximate chi-square distribution with $k-1$ degrees of freedom. Reject the null hypothesis of equal proportions if Q is large (since if male and female distributed nearly equally, the expected number of each should be $n_i\hat{p}_j$):

$$Q\geq\chi^2_{\alpha,k-1}$$

(omit the derive of the above Q)

Generally,

$$Q=\sum_{i=1}^{h}\sum_{j=1}^{k}\frac{\left(y_{ij}-n_i\hat{p}_j\right)^2}{n_i\hat{p}_j}\sim\chi^2_{(h-1)(k-1)}$$

Further example

The head of a surgery department at a university medical center was concerned that surgical residents in training applied unnecessary blood transfusions at a different rate than the more experienced attending physicians. Therefore, he ordered a study of the 49 Attending Physicians and 71 Residents in Training with privileges at the hospital. For each of the 120 surgeons, the number of blood transfusions prescribed unnecessarily in a one-year period was recorded. Based on the number recorded, a surgeon was identified as either prescribing unnecessary blood transfusions Frequently, Occasionally, Rarely, or Never. Here's a summary table (or "contingency table") of the resulting data:

Physician	Frequent	Occasionally	Rarely	Never	Total
Attending	6.942	12.658	22.05	7.35	49
Resident	10.058	18.342	31.95	10.65	71
Total	17	31	54	18	120

Here,

$H_0 : p_{RF} = p_{AF}, p_{RO} = p_{AO}, p_{RR} = p_{AR}$, and $p_{RN} = p_{AN}$

$H_1 : p_{RF} \neq p_{AF}$ or $p_{RO} \neq p_{AO}$ or $p_{RR} \neq p_{AR}$, or $p_{RN} \neq p_{AN}$

We should also calculate the expected counts under the null hypothesis. The expected counts are calculated as follows:

Physician	Frequent	Occasionally	Rarely	Never	Total
Attending	6.942	12.658	22.05	7.35	49
Resident	10.058	18.342	31.95	10.65	71
Total	17	31	54	18	120

where, for example, $6.942 = \frac{17}{120} \times 49$ and $10.058 = \frac{17}{120} \times 71$. Now that we have the observed and expected counts, calculating the chisquare statistic is a straightforward exercise:

$$Q = \frac{(2 - 6.942)^2}{6.942} + \cdots + \frac{(5 - 10.65)^2}{10.65} = 31.88$$

The chi-square test tells us to reject the null hypothesis, at the 0.05 level, if Q is greater than a chi-square random variable with 3 degrees of freedom, that is, if $Q = 31.88 > 7.815$, we reject the null hypothesis.

Application of testing for independence

This is to look at whether two or more categorical variables are independent.

(previously,the the sampling scheme involves: Taking two random (and therefore independent) samples with n1 and n2 fixed in advance and observing into which of the k categories the first random samples fall, and observing into which of the k categories the second random samples fall.)

lets consider a different example to illustrate an alternative sampling scheme. Suppose 395 people are randomly selected, and are "cross-classified" into one of eight cells, depending into which age category they fall and whether or not they support legalizing marijuana:

(the sampling scheme involves: Taking one random sample of size n, with n fixed in advance, and then "cross-classifying" each subject into one and only one of the mutually exclusive and exhaustive $A_i \cap B_j$ cells.)

Marijuana Support		Variable B (Age)				
Variable A	OBSERVED	$(18-24)B_1$	(25-34) B_12	(35-49) B_3	$(50-64)B_4$	Total
	(YES) A_1	60	54	46	41	201
	(NO) A_2	40	44	53	57	194
	Total	100	98	99	98	$n = 395$

Here,

H0 : Variable A is independent of variable B, that is $P(A_i \cap B_j) = PA_i \times B_j$ for all i and j

H1: Variable A is not independent of variable B.

Generally,

Suppose we have k (column) levels of Variable B indexed by the letter j, and h (row) levels of Variable A indexed by the letter i. Then, we can summarize the data and probability model in tabular format, as follows:

Variable B					
Variable A	$B_1(j=1)$	$B_2(j=2)$	$B_3(j=3)$	$B_4(j=4)$	Total
$A_1(i=1)$	$Y_{11}\,(p_{11})$	$Y_{12}\,(p_{12})$	$Y_{13}\,(p_{13})$	$Y_{14}\,(p_{14})$	$(p_{1.})$
$A_2(i=2)$	$Y_{21}\,(p_{21})$	$Y_{22}\,(p_{22})$	$Y_{23}\,(p_{23})$	$Y_{24}\,(p_{24})$	$(p_{2.})$
Total	$(p_{.1})$	$(p_{.2})$	$(p_{.3})$	$(p_{.4})$	n

where $p_{ij} = Y_{ij}/n, p_{i\cdot} = \sum_{j=1}^{k} p_{ij}$, and $p_{\cdot j} = \sum_{i=1}^{h} p_{ij}$

$Q = \sum_{j=1}^{k} \sum_{i=1}^{h} \frac{\left(y_{ij} - \frac{y_{i\cdot} y_{\cdot j}}{n}\right)^2}{\frac{y_{i\cdot} y_{\cdot j}}{n}} \sim \chi^2_{(h-1)(k-1)}$

Are chi-square statistic for homogeneity and the chi-square statistic for independence equivalent?

Although their chi-square statistics are equivalent, the two tests are not equivalent since their sampling experiment designs are different.

Here's the table of expected counts:

Bicycle Riding Interest		Variable B (Age)				
Variable A	EXPECTED	18-24	25-34	35-49	50-64	Total
	YES	50.886	49.868	50.377	49.868	201
	NO	49.114	48.132	48.623	48.132	194
	Total	100	98	99	98	395

$$Q = \frac{(60 - 50.886)^2}{50.886} + \cdots + \frac{(57 - 48.132)^2}{48.132} = 8.006$$

The chi-square test tells us to reject the null hypothesis, at the 0.05 level, since Q is greater than a chi-square random variable with 3 degrees of freedom, that is, $Q = 8.006 > 7.815$.

Summary

Parametric tests make assumptions that aspects of the data follow some sort of theoretical probability distribution. Non-parametric tests or distribution free methods do not, and are used when the distributional assumptions for a parametric test are not met. While this is an advantage, it often comes at a cost of power (in the sense they are less likely to be able to detect a difference when a true difference exists).

Most non-parametric tests are just hypothesis tests; there is no estimation of a confidence interval.

Most non-parametric methods are based on ranking the values of a variable in ascending order and then calculating a test statistic based on the sums of these ranks.

Non-parametric tests include:

- Two-sample independent t-test - Wilcoxon rank-sum test or Mann-Whitney U test
- Paired t-test - Wilcoxon signed-rank test
- One-way ANOVA - Kruskal-Wallace Test
- Normality tests - Shapiro-Wilk test and Kolmogorov-Smirnov test

Linear Regression

Simple linear regression

Brief introduction

Some people think simple methods is bad and like complicated methods , but actually simple is very good–SLR works very well in lots of situations.

SLR is used to answer:

is there a relationship between..

How strong the relationship is

which variable contribute to this relationship

How accurate could we predict the response variable

Is the relationship linear?

Is there a synergy among independent variables?

This is a model with two random variables, X and Y, where we are trying to predict Y from X. Here are the model's assumptions:

- The distribution of X is arbitrary, possibly is even non-random;
- If X=x, then $Y = \beta_0 + \beta_1 x + \epsilon$ for some constants β_0, β_1 and some random noise variable ϵ
- ϵ has mean 0, a constant variance σ^2, and is uncorrelated with X and uncorrelated across observations $\text{Cov}(\epsilon_i, \epsilon_j) = 0$ for $i \neq j$

Using Least Squares, we can estimate $\hat{\beta}_0$ and $\hat{\beta}_1$, which are unbiased estimates of β_0 and β_1.

- Gaussian-Noise Simple Linear Regression Model

Now we further assume that the distribution of ϵ is normal, i.e. $\epsilon \sim N(0, \sigma^2)$, independent of X.

They tell us, exactly, the probability distribution for Y given X, and so will let us get exact distributions for predictions and for other inferential statistics.

Maximum Likelihood Estimation (MLE)

Introduction to MLE

Likelihood is a fundamental concept in statistics that measures how well a particular set of parameters (e.g., the mean of a distribution) explains observed data. Think of it as a "score" that tells you which parameter values make your data most plausible.

Compared to probability, which answers: "What's the chance of seeing this data if we assume specific parameters?" , likelihood answers: "Given this data, how plausible are these parameters?"

If the parameters are b_0, b_1, s^2 (reserving the Greek letters for their true values), then $Y \mid X = x \sim N\left(b_0 + b_1 x, s^2\right)$, and Y_i and Y_j are independent given X_i and X_j, so the overall likelihood is

$$\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(y_i-(b_0+b_i x_i))^2}{2s^2}}$$

As usual, we work with the log-likelihood, which gives us the same information but replaces products with sums:

$$L\left(b_0, b_1, s^2\right) = -\frac{n}{2}\ln\left(2\pi s^2\right) - \frac{1}{2s^2}\sum_{i=1}^{n}\left(y_i - \left(b_0 + b_1 x_i\right)\right)^2$$

maximize it:

$$\frac{\partial L}{\partial b_0} = -\frac{1}{2s^2}\sum_{i=1}^{n} 2\left(y_i - \left(b_0 + b_1 x_i\right)\right)(-1)$$

$$\frac{\partial L}{\partial b_1} = -\frac{1}{2s^2}\sum_{i=1}^{n} 2\left(y_i - \left(b_0 + b_1 x_i\right)\right)(-x_i)$$

Same result of MLE as least squares in linear regression

Notice that when we set these derivatives to zero, all the multiplicative constants - in particular, the prefactor of $1/2s^2$ - go away. We are left with

$$\sum_{i=1}^{n}\left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)\right) = 0$$

$$\sum_{i=1}^{n}\left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_0 x_i\right)\right) x_i = 0$$

These are, up to a factor of $1/n$, exactly the equations we got from the method of least squares. That means that the least squares solution is the maximum likelihood estimate under the Gaussian noise model.

Maximum likelihood estimates of the regression curve coincide with least-squares estimates when the noise around the curve is additive, Gaussian, of constant variance, and both independent of X and of other noise terms. If any of those assumptions fail, maximum likelihood and least squares estimates can diverge.

Hypothesis testing for estimates with unknown σ^2

Residual sum of squares (RSS) and t-statistics construction in linear regression

It can be shown that (the proof is beyond the scope of this course)

$$\frac{RSS}{\sigma^2} = \frac{(n-2)\hat{\sigma}^2}{\sigma^2} \sim \chi^2_{n-2}$$

This allows us to construct a t-value

$$t = \frac{\hat{\beta} - \beta}{s_{\hat{\beta}}} \sim t_{n-2}$$

Under the normality assumption of the error terms, the estimator of the slope coefficient will itself be normally distributed with mean β_i and variance $\mathrm{Var}[\beta_i]$ For $\hat{\beta}_1$, its mean is β_1 and its variance is $\sigma^2 / \sum (x_i - \bar{x})^2$. When σ^2 is known, we know

$$\frac{\hat{\beta}_1 - \beta_1}{\frac{\sigma}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2}}}$$

follows standard normal distribution. However, in practice, σ^2 is often unknown. We then divide this standard normal distributed term by

$$\sqrt{\frac{(n-2)\hat{\sigma}^2}{(n-2)\sigma^2}} = \frac{\hat{\sigma}}{\sigma}$$

Therefore, when we write

$$s_{\hat{\beta}_1} = \frac{\hat{\sigma}}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2}}$$

we construct a t-statistic for $\hat{\beta}_1$ with degrees of freedom $n-2$. This then allows us to construct a $100(1-\alpha)\%$ confidence interval for β_1 :

$$\hat{\beta}_1 \pm t_{n-2,\alpha/2} \times s_{\hat{\beta}_1}$$

We can also do similar calculation to get the t-statistic and confidence interval for β_0.

Hyphothesis Testing

$$t = \hat{\beta} - \beta / s_{\hat{\beta}_1} \sim t_{n-2}$$

$H_0 : \beta_i = 0$

$H_1 : \beta_i \neq 0$

Codes of linear regression with CI

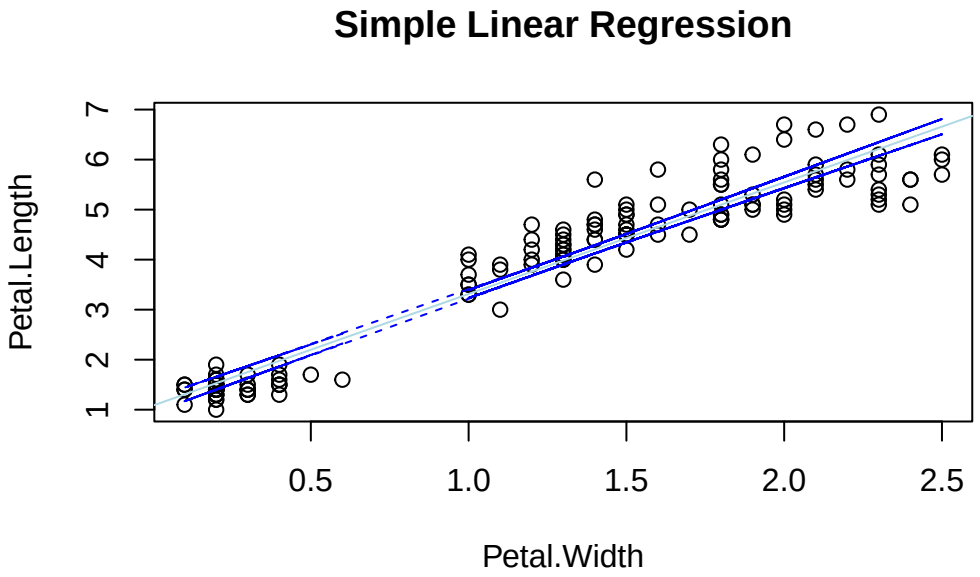
```
lmodel <- lm(Petal.Length ~ Petal.Width, data=iris)
summary(lmodel)
```

```
Call:
lm(formula = Petal.Length ~ Petal.Width, data = iris)

Residuals:
     Min       1Q   Median       3Q      Max
-1.33542 -0.30347 -0.02955  0.25776  1.39453

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.08356    0.07297   14.85   <2e-16 ***
Petal.Width  2.22994    0.05140   43.39   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4782 on 148 degrees of freedom
Multiple R-squared:  0.9271,    Adjusted R-squared:  0.9266
F-statistic:  1882 on 1 and 148 DF,  p-value: < 2.2e-16
```

```
confint(lmodel)
```

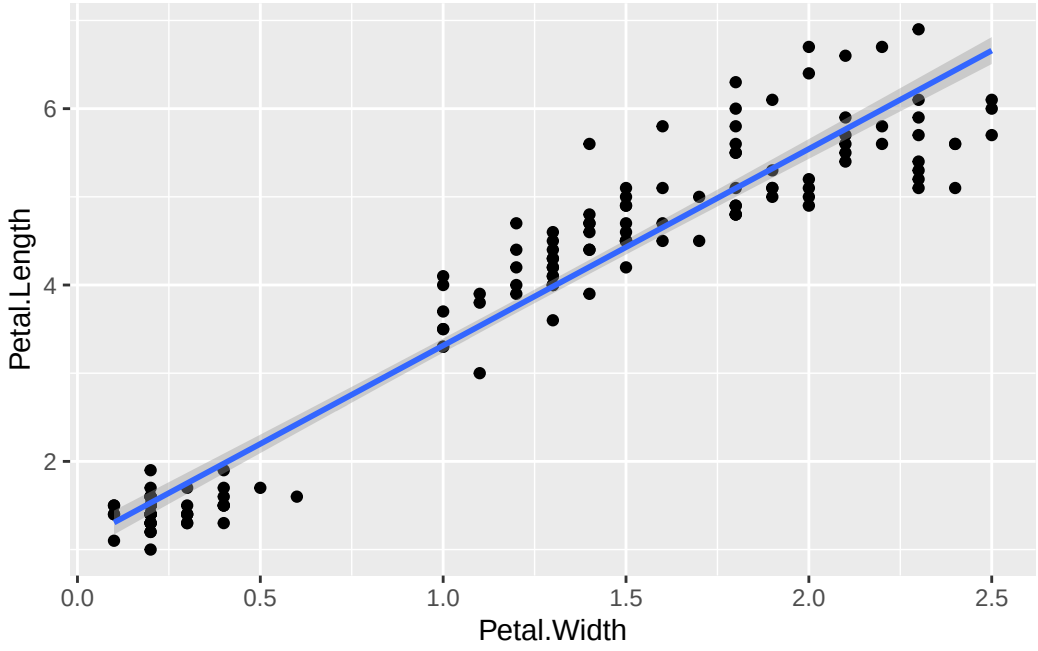
```
                2.5 %   97.5 %
(Intercept) 0.9393664 1.227750
Petal.Width 2.1283752 2.331506
```

```r
conf_interval <- predict(lmodel, data=iris, interval='confidence', level=0.95)
plot(iris$Petal.Width, iris$Petal.Length,
     xlab='Petal.Width', ylab='Petal.Length',
     main='Simple Linear Regression')
abline(lmodel, col='lightblue')
matlines(iris$Petal.Width, conf_interval[,2:3], col='blue', lty=2)
```




```r
# Using ggplot2
library(ggplot2)
ggplot(iris, aes(x=Petal.Width, y=Petal.Length)) +
  geom_point() +
  geom_smooth(method=stats::lm, se=T, level=0.95)
```

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



Linear Regression and ANOVA

```
fev_dat <- read.table('fev_dat.txt', header=T)
fev_dat_subset <- fev_dat[fev_dat$age >= 6 & fev_dat$age <= 10,]
ggplot(fev_dat_subset, aes(x=age, y=FEV)) +
  geom_point() +
  geom_smooth(method=stats::lm, se=T, level=0.95)
summary(aov(FEV ~ age, data=fev_dat_subset))
summary(lm(FEV ~ age, data=fev_dat_subset))
anova(lm(FEV ~ age, data=fev_dat_subset))
```

R^2–the fraction of variability explained by the regression

$$R^2 = 1 - \frac{SSR}{SSTO}$$

Multiple linear regression (MLR)

$y = X\beta + \epsilon$, where y is a n $\times$ 1 row vector, X is a n $\times$ (k + 1) matrix, and is a (k + 1) $\times$ 1 column vector for all n observations.

A potential problem in practice –multicollinearity

When multicollinearity exists, any of the following pitfalls can be exacerbated:

- The estimated regression coefficient of any one variable depends on which other predictors are included in the model
- The precision of the estimated regression coefficients decreases as more predictors are added to the model
- The marginal contribution of any one predictor variable in reducing the error sum of squares depends on which other predictors are already in the model
- Typothesis tests for $\mathrm{j} = 0$ may yield different conclusions depending on which predictors are in the model

Perfect multicollinearity

Perfect multicollinearity refers to a situation where the predictive variables have an exact linear relationship. When there is perfect collinearity, the design matrix X has less than full rank, and therefore the moment matrix X X cannot be inverted. In this situation, the parameter estimates of the regression are not well-defined, as the system of equations has infinitely many solutions.

Imperfect multicollinearity

Imperfect multicollinearity refers to a situation where the predictive variables have a nearly exact linear relationship.

$R^2 = r^2$ where r is the Pearson correlation coefficient.

Adjusted R-squared

Adjusted R-squared is a modified version of R-squared that adjusts for the number of predictors in a model. It provides a more accurate measure of the model's explanatory power, penalizing for the addition of irrelevant predictors. This helps in comparing models with different numbers of predictors.

Logistic Regression

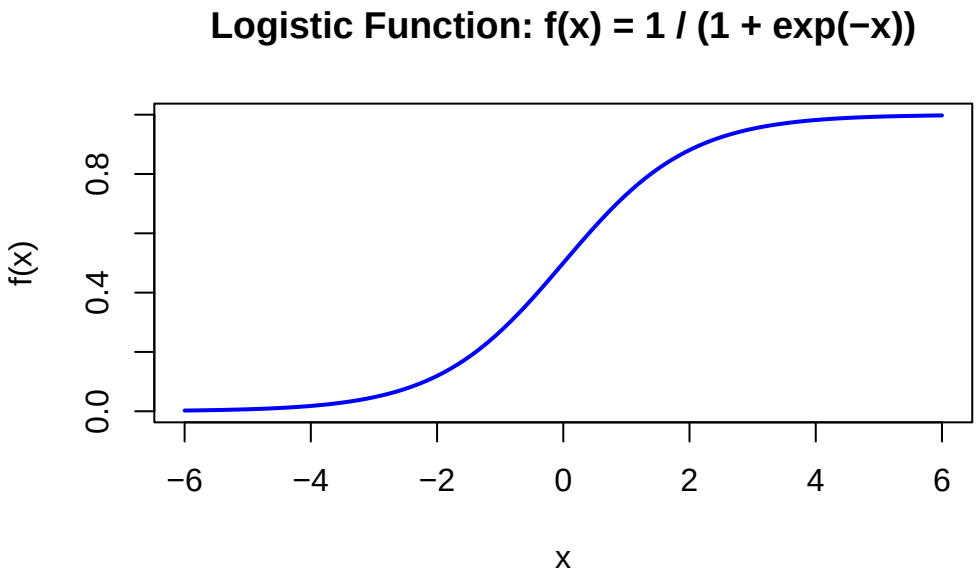
This is a regression of categorical outcome.

In regression analysis with a categorical outcome, such as predicting a binary variable (yes or no), simple linear regression is not ideal. This is because:

- The predicted values may fall outside the range of 0 to 1, which is not meaningful for probabilities.
- Small changes in the predictors can lead to relatively small fluctuations in the predicted probabilities near the 0.5 mark (natural threshold), which is actually where decision-making is most critical.

So we need S-curve to satisfy above things.

```
curve(1 / (1 + exp(-x)), from = -6, to = 6, xlab = "x", ylab = "f(x)",
main = "Logistic Function: f(x) = 1 / (1 + exp(-x))", col = "blue", lwd = 2)
```



An example of a not well-predicted logistic model (since stock is not easy to predict)

```
library(ISLR2)
attach(Smarket)
glm.fits <- glm(
    Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
    data = Smarket, family = binomial
  )
summary(glm.fits)
```

```
Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = Smarket)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.126000   0.240736  -0.523    0.601
Lag1        -0.073074   0.050167  -1.457    0.145
Lag2        -0.042301   0.050086  -0.845    0.398
Lag3         0.011085   0.049939   0.222    0.824
Lag4         0.009359   0.049974   0.187    0.851
Lag5         0.010313   0.049511   0.208    0.835
Volume       0.135441   0.158360   0.855    0.392

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1731.2  on 1249  degrees of freedom
Residual deviance: 1727.6  on 1243  degrees of freedom
AIC: 1741.6

Number of Fisher Scoring iterations: 3
```

```
glm.probs <- predict(glm.fits, type = "response")
glm.probs[1:10]
```

```
        1         2         3         4         5         6         7         8
0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.5092292
        9        10
0.5176135 0.4888378
```

```
glm.pred <- rep("Down", 1250)
glm.pred[glm.probs > .5] = "Up"
table(glm.pred, Direction)
```

```
        Direction
glm.pred Down  Up
    Down  145 141
    Up    457 507
```

```
(507 + 145) / 1250
```

```
[1] 0.5216
```

```
mean(glm.pred == Direction)
```

```
[1] 0.5216
```

Odds

Odds are another way of quantifying the probability of an event, commonly used in gambling (and logistic regression).

For some event E,

$$\text{odds}(E) = \frac{P(E)}{P\left(E^c\right)} = \frac{P(E)}{1 - P(E)}$$

The odds ratio (OR) is the ratio of the odds of an event occurring in one group to the odds of it occuring in another group. If the event in each of the groups are p_1 (first group) and p_2 (second group), then the odds ratio is:

$$\text{OR} = \frac{p_1 / \left(1 - p_1\right)}{p_2 / \left(1 - p_2\right)}$$

Generalized Linear Models

Under the hood, we're still using a linear model ($\beta_0 + \beta_1 x$), but now it's embedded in a function that ensures valid probabilities. This is the essence of logistic regression - a generalized linear model (GLM) designed for binary outcomes.

Given predictor X and d an outcom Y, a GLM is defined by three components: - A random component, that specifies a distribution for $Y \mid X$ - A systematic component, that relates a parameter η to the predictor X

$$\eta = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$$

- A link function, that connects the random and systematic component

Random Component

The random component specifies a distribution for the outcome variable (conditional on X). In the case of linear regression, we assume that $Y \mid X \sim \mathcal{N}(\mu, \sigma^2)$, for some mean μ and variance σ^2. In the case of logistic regression, we assume that $Y \mid X \sim \text{Bern}(p)$ for some probability p.

In a generalized model, we are allowed to assume that $Y \mid X$ has a probability density function or probability mass function of the form

$$f(y; \theta, \phi) = \exp\left(\frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)\right)$$

Here θ, ϕ are parameters, and a, b, c are functions. Any density of the above form is called an exponential family density. The parameter θ is called the natural parameter, and the parameter ϕ the dispersion parameter.

Exponential Family

Exponential families include many of the most common distributions. For example: - Exponential

$$f(y; \lambda) = \lambda e^{-\lambda y} = \exp(-y\lambda + \ln \lambda)$$

where $\theta = -\lambda, \phi = 1, b(\theta) = \ln \lambda, a(\phi) = 1$, and $c(y, \phi) = 0$ - Poisson

$$f(y; \lambda) = \frac{e^{-\lambda}\lambda^y}{y!} = \exp(y \ln \lambda - \lambda - \ln(y!))$$

where $\theta = \ln \lambda, \phi = 1, b(\theta) = e^\theta = \lambda, a(\phi) = 1$, and $c(y, \phi) = -\lambda - \ln(y!)$

Systematic Component and Link Component

The systematic component relates a parameter η to the predictors X. In a GLM, this is always done via

$$\eta = X\beta = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$$

We will denote the expectation of the distribution in random component as μ, i.e., $\mathbb{E}[Y \mid X] = \mu$. It will be our goal to estimate μ. Finally, the link component connects the random and systematic components, via a link function g. In particular, this link function provides a connection between μ and η, as in

$$g(\mu) = \eta \quad \text{or} \quad \mu = g^{-1}(\eta)$$

Example

Gaussian-noise Linear Regression

- Random Component: $Y \mid X \sim \mathcal{N}(\mu, \sigma^2)$ and $\mathbb{E}[Y \mid X] = \mu$
- Systematic Component: $\eta = X\beta$
- Link Component: $g(\mu) = \mu$, so that $\mu = \eta = X\beta$

Bernoulli

Suppose that $Y \in \{0, 1\}$, and we model the distribution of $Y \mid X$ as Bernoulli with success probability p. Then the probability mass function (not a density, since Y is discrete) is

$$f(y) = p^y (1-p)^{1-y}$$

We can rewrite to fit the exponential family form as

$$\begin{aligned} f(y) &= \exp(y \log p + (1-y)\log(1-p)) \\ &= \exp(y \log(p/(1-p)) + \log(1-p)) \end{aligned}$$

$$f(y; \theta, \phi) = \exp\left(\frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)\right)$$

Here we would identify $\theta = \log(p/(1-p))$ as the natural parameter. Note that the mean here is $\mu = p$, and using the inverse of the above relationship, we can directly write the mean p as a function of θ, as in $p = e^\theta / (1 + e^\theta)$. Hence $b(\theta) = \log(1-p) = -\log(1 + e^\theta)$. There is no dispersion parameter, so we can set $a(\phi) = 1$. Also, $c(y, \phi) = 0$.

Link Function

$$g(\mu) = \Phi^{-1}(\mu)$$

where Φ is the standard normal CDF.

Logistic

The three GLM criteria give us:

- $y_i \sim \text{Bern}(p_i)$
- $\eta = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$
- $\text{logit}(p) = \log \frac{p}{1-p} = \eta$

From which we know,

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k X_{ik})}{1 + \exp(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k X_{ik})}$$

Survival Analysis

Survival analysis is used to analyze data in which the time until the event is of interest. The response is often referred to as a failure time, survival time, or event time.

Some definations

Hazard ratios; ratios of hazard functions between different groups (e.g., exposed vs. unexposed) while adjusting for confounders.

censoring - which occurs when the survival time is only partially known

- Fixed type I censoring occurs when a study is designed to end after C years of follow-up. In this case, everyone who does not have an event observed during the course of the study is censored at C years.
- In random type I censoring, the study is designed to end after C years, but censored subjects do not all have the same censoring time. This is the main type of right-censoring we will be concerned with.
- In type II censoring, a study ends when there is a pre-specified number of events.

Kaplan-Meier estimate

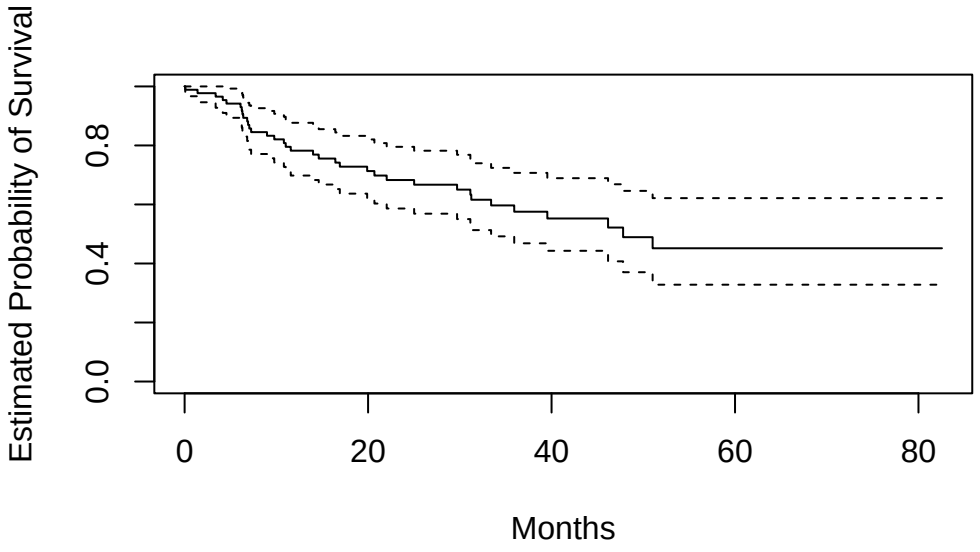
```
library(ISLR2)
names(BrainCancer)
```

```
[1] "sex"       "diagnosis" "loc"       "ki"        "gtv"       "stereo"
[7] "status"    "time"
```

```
attach(BrainCancer)
table(status)
```

```
status
 0  1
53 35
```

```
library(survival)
fit.surv <- survfit(Surv(time, status) ~ 1)
plot(fit.surv, xlab = "Months",
    ylab = "Estimated Probability of Survival")
```



Cox-proportional harzards model

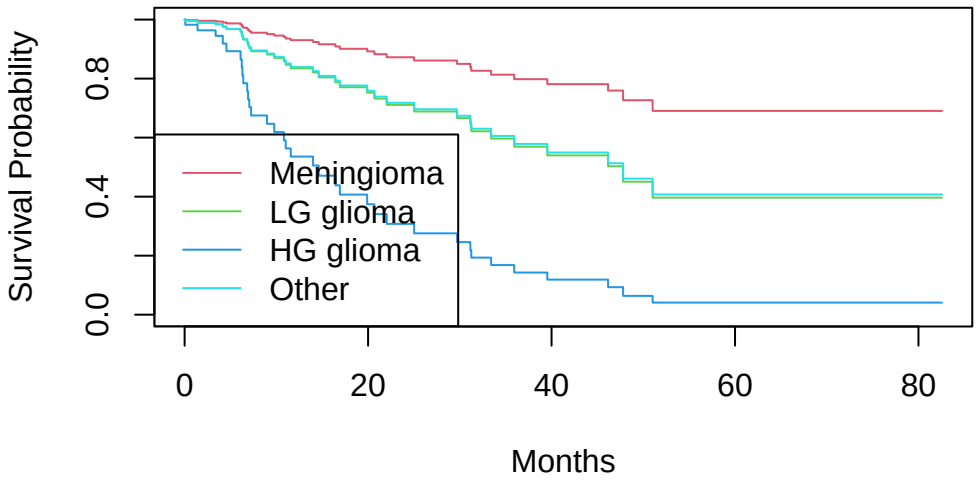
$S(t) = P(T>t)$=1-$F(t)$

$h(t) = \lim_{\Delta t \to 0} \frac{P(t<T \leq t+\Delta t | T>t)}{\Delta t}$

```
fit.all <- coxph(
Surv(time, status) ~ sex + diagnosis + loc + ki + gtv +
   stereo)
fit.all
```

```
Call:
coxph(formula = Surv(time, status) ~ sex + diagnosis + loc +
    ki + gtv + stereo)

                       coef exp(coef) se(coef)      z        p
sexMale             0.18375   1.20171  0.36036  0.510  0.61012
diagnosisLG glioma  0.91502   2.49683  0.63816  1.434  0.15161
diagnosisHG glioma  2.15457   8.62414  0.45052  4.782 1.73e-06
diagnosisOther      0.88570   2.42467  0.65787  1.346  0.17821
locSupratentorial   0.44119   1.55456  0.70367  0.627  0.53066
ki                 -0.05496   0.94653  0.01831 -3.001  0.00269
gtv                 0.03429   1.03489  0.02233  1.536  0.12466
stereoSRT           0.17778   1.19456  0.60158  0.296  0.76760

Likelihood ratio test=41.37  on 8 df, p=1.776e-06
n= 87, number of events= 35
   (1 observation deleted due to missingness)
```

```
modaldata <- data.frame(
     diagnosis = levels(diagnosis),
     sex = rep("Female", 4),
     loc = rep("Supratentorial", 4),
     ki = rep(mean(ki), 4),
     gtv = rep(mean(gtv), 4),
     stereo = rep("SRT", 4)
     )
survplots <- survfit(fit.all, newdata = modaldata)
plot(survplots, xlab = "Months",
    ylab = "Survival Probability", col = 2:5)
legend("bottomleft", levels(diagnosis), col = 2:5, lty = 1)
```



Final review of R codes

Calculation

```
log(exp(1)) #  base `e` is the default (log(e) is not defined)
```

```
[1] 1
```

Vectors

```
x <- c(1,2,3,4)
class(x)
```

```
[1] "numeric"
```

```
x %*%x # scalar ("inner") product (but default in R as an 1*1 matrix)
```

```
     [,1]
[1,]   30
```

```r
rep(c('a','b'),3)
```

```
[1] "a" "b" "a" "b" "a" "b"
```

```r
rep(c(2, 4, 8), each = 3)
```

```
[1] 2 2 2 4 4 4 8 8 8
```

```r
y <- seq(from = 1, to = 4, by =1)
class(x)
```

```
[1] "numeric"
```

```r
str(x)
```

```
 num [1:4] 1 2 3 4
```

```r
x <- c(-5:5)
str(x)
```

```
 int [1:11] -5 -4 -3 -2 -1 0 1 2 3 4 ...
```

```r
1:4 # c(1,2,3,4) and 1:4 are the same in.R
```

```
[1] 1 2 3 4
```

```r
seq(1,5, length.out=11)
```

```
 [1] 1.0 1.4 1.8 2.2 2.6 3.0 3.4 3.8 4.2 4.6 5.0
```

```r
# vac<-c((1,2,3),(3,4,5)) is wrong, but the below is true
vec1 <- c(1,2,3)
vec2 <- c(4,5,6)
vec3 <- c(vec1, vec2)
vec3[1] == vec1[1]
```

```
[1] TRUE
```

```r
vec3[3:5];vec3[c(2,3)]
```

```
[1] 3 4 5
```

```
[1] 2 3
```

```r
vec3[-1] # everything but the first element
```

```
[1] 2 3 4 5 6
```

```r
vec3[-2*c(1,2)]
```

```
[1] 1 3 5 6
```

```r
x <- -5:5

abs(-5:5)
```

```
 [1] 5 4 3 2 1 0 1 2 3 4 5
```

```r
x <- c(1, 2, 3, 4, 5,6)
y <- c(10,11)
result <- x + y
print(result)
```

```
[1] 11 13 13 15 15 17
```

```r
# x * y
c(1,2)*c(2,3)
```

```
[1] 2 6
```

```r
c(1,2)%*%c(2,3)
```

```
     [,1]
[1,]    8
```

```r
#####Application
# Calculate the sample(var(x)) and population variance
x <- c(1, 2, 3, 4, 5)
n <- length(x)
# Calculate the sample variance using R's var() function
sample_variance <-var(x)
sample_variance
```

```
[1] 2.5
```

```r
population_variance <- sample_variance * (n - 1) / n
population_variance
```

```
[1] 2
```

Rmd knowledge

{r, echo = FALSE} –Hidden Code

{r, eval = FALSE} –Do not run this code

{r, message = FALSE} –Do not show the message

{r, warning = FALSE} –Do not show the warning

{r,results='hide'} –Do not show the results

Probability in R

```r
dnorm(0) # density at 0
```

```
[1] 0.3989423
```

```r
pnorm(-1) # cumulative probability at -1
```

```
[1] 0.1586553
```

```r
pnorm(-1,lower.tail = F) # cumulative probability at -1, upper tail
```

```
[1] 0.8413447
```

```r
pnorm(0)
```

```
[1] 0.5
```

```r
qnorm(0) # quantile at 0 (with the cumulative probability of 0)
```

```
[1] -Inf
```

```r
pnorm(1.645)
```

```
[1] 0.9500151
```

```r
qnorm(0.95) # norm quantile at 0.95 (with the cumulative probability of 0.95)
```

```
[1] 1.644854
```

```r
pnorm(1.96)
```

```
[1] 0.9750021
```

```r
qnorm(0.975)
```

```
[1] 1.959964
```

```r
library(mosaic)
```

```
Registered S3 method overwritten by 'mosaic':
  method                           from
  fortify.SpatialPolygonsDataFrame ggplot2
```

```
The 'mosaic' package masks several functions from core packages in order to add
additional features.  The original behavior of these functions should not be affected by this


Attaching package: 'mosaic'

The following objects are masked from 'package:dplyr':

    count, do, tally

The following object is masked from 'package:Matrix':

    mean

The following objects are masked from 'package:car':

    deltaMethod, logit

The following object is masked from 'package:ggplot2':

    stat

The following objects are masked from 'package:stats':

    binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
    quantile, sd, t.test, var

The following objects are masked from 'package:base':

    max, mean, min, prod, range, sample, sum
```

```
ppois(3, lambda = 2)
```

```
[1] 0.8571235
```

```
sum(dpois(0:3, lambda = 2))
```

```
[1] 0.8571235
```

```r
ppois(3, lambda = 2) == sum(dpois(0:3, lambda = 2)) # this inequivalance is because of the fl
```

```
[1] FALSE
```

R basic

```r
paste("Good", "afternoon", "ladies", "and", "gentlemen")
```

```
[1] "Good afternoon ladies and gentlemen"
```

```r
paste0("Good", "afternoon", "ladies", "and", "gentlemen")
```

```
[1] "Goodafternoonladiesandgentlemen"
```

```r
x <- -10:10
which(x>0)
```

```
 [1] 12 13 14 15 16 17 18 19 20 21
```

```r
x[which(x>0)]
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```r
vowels <- c('a','e','i','o','u')
which(is.element(letters, vowels))
```

```
[1]  1  5  9 15 21
```

Data class

Numeric

Integer

```r
y <- 42L
class(y)
```

```
[1] "integer"
```

Character

```r
x <- "123"
class(x)
```

```
[1] "character"
```

```r
# [1] "character"

x <- as.numeric(x)
class(x)
```

```
[1] "numeric"
```

```r
# [1] "numeric"
```

Logical

```r
# Note that logical elements are NOT in quotes.
z = c("TRUE", "FALSE", "TRUE", "FALSE")
class(z)
```

```
[1] "character"
```

```r
as.logical(z)
```

```
[1]  TRUE FALSE  TRUE FALSE
```

```
# TRUE = 1 and FALSE = 0. sum() and mean() work on logical vectors

# remember:
TRUE & TRUE
```

```
[1] TRUE
```

```
TRUE & FALSE
```

```
[1] FALSE
```

```
TRUE | FALSE
```

```
[1] TRUE
```

Factor

```
y <- c('B','B','A','A','C')
z <- factor(y)
str(z)
```

```
 Factor w/ 3 levels "A","B","C": 2 2 1 1 3
```

```
as.numeric(z)
```

```
[1] 2 2 1 1 3
```

```
levels(z)
```

```
[1] "A" "B" "C"
```

```
z <- factor(z,                          # vector of data levels to convert
            levels=c('B','A','C'),      # Order of the levels
            labels=c("B Group", "A Group", "C Group")) # Pretty labels to use
z
```

```
[1] B Group B Group A Group A Group C Group
Levels: B Group A Group C Group
```

```
#####Application
### eg of use in the plot's x-axis name lable

iris$Species <- factor(iris$Species,
                       levels = c('versicolor','setosa','virginica'),
                       labels = c('Versicolor','Setosa','Virginica'))
#boxplot(Sepal.Length ~ Species, data=iris)

### another eg
#age_category <- ifelse(ages >= 18, "Adult", "Minor")
#age_factor <- factor(age_category, levels = c("Minor", "Adult"))
#age_factor

### transform a continuous numerical vector into a factor

x <- 1:10
cut(x, breaks = c(0, 2.5, 5.0, 7.5, 10))
```

```
 [1] (0,2.5]  (0,2.5]  (2.5,5]  (2.5,5]  (2.5,5]  (5,7.5]  (5,7.5]  (7.5,10]
 [9] (7.5,10] (7.5,10]
Levels: (0,2.5] (2.5,5] (5,7.5] (7.5,10]
```

```
x<-cut(x, breaks=3, labels=c('Low','Medium','High'))
str(x)
```

```
 Factor w/ 3 levels "Low","Medium",..: 1 1 1 1 2 2 2 3 3 3
```

Date and time

The following symbols can be used with the `format()` function to print dates.

- %d day as a number (0-31) 01-31
- %a abbreviated weekday Mon
- %A unabbreviated weekday Monday
- %m month (00-12) 00-12
- %b abbreviated month Jan
- %B unabbreviated month January
- %y 2-digit year 07
- %Y 4-digit year 2007

```r
mydates <- as.Date(c("2023-04-07", "2023-01-01"))
mydates
```

```
[1] "2023-04-07" "2023-01-01"
```

```r
days <- mydates[1] - mydates[2]; days
```

```
Time difference of 96 days
```

```r
today <- Sys.Date()
format(today, format="%B %d %Y")
```

```
[1] "June 07 2025"
```

Data frame

```r
Data_Frame <- data.frame(Tr =c("1","2","3"),
                         Pu =c(11,21,32),
                         Dur=c(22,222,1))
Data_Frame
```

```
  Tr Pu Dur
1  1 11  22
2  2 21 222
3  3 32   1
```

```r
summary(Data_Frame)
```

```
      Tr                  Pu             Dur
 Length:3           Min.   :11.00   Min.   :  1.00
 Class :character   1st Qu.:16.00   1st Qu.: 11.50
 Mode  :character   Median :21.00   Median : 22.00
                    Mean   :21.33   Mean   : 81.67
                    3rd Qu.:26.50   3rd Qu.:122.00
                    Max.   :32.00   Max.   :222.00
```

```r
### table and data frame
table(mpg$class)
```

```
   2seater    compact    midsize    minivan     pickup subcompact        suv
         5         47         41         11         33         35         62
```

```r
df <- as.data.frame(table(mpg$class))
df
```

```
        Var1 Freq
1    2seater    5
2    compact   47
3    midsize   41
4    minivan   11
5     pickup   33
6 subcompact   35
7        suv   62
```

List

```r
# List ---Can hold vectors, strings, matrices, models, list of other list, lists upon lists!

mylist <- list(letters=c("a","b","c"),
               numbers=1:3,matrix(1:25,ncol=5))
head(mylist)
```

```
$letters
[1] "a" "b" "c"

$numbers
[1] 1 2 3

[[3]]
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
```

```r
# Can reference data using $ (if the elements are named), or using [], or [[]]

mylist[1] # list
```

```
$letters
[1] "a" "b" "c"
```

```r
mylist["letters"] # list
```

```
$letters
[1] "a" "b" "c"
```

```r
mylist[[1]] # vector
```

```
[1] "a" "b" "c"
```

```r
mylist$letters == mylist[["letters"]]
```

```
[1] TRUE TRUE TRUE
```

```r
mylist[[3]][1:2,1:2]
```

```
     [,1] [,2]
[1,]    1    6
[2,]    2    7
```

```r
class(mylist[[3]][1:2,1:2])
```

```
[1] "matrix" "array"
```

```r
x = c(0, 2, 2, 3, 4); 2 %in% x
```

```
[1] TRUE
```

```r
# eg of using list
x <- c(5.1, 4.9, 5.6, 4.2, 4.8, 4.5, 5.3, 5.2)   # some toy data
results <- t.test(x, alternative='less', mu=5)   # do a t-test
str(results)
```

```
List of 10
 $ statistic  : Named num -0.314
  ..- attr(*, "names")= chr "t"
 $ parameter  : Named num 7
  ..- attr(*, "names")= chr "df"
 $ p.value    : num 0.381
 $ conf.int   : num [1:2] -Inf 5.25
  ..- attr(*, "conf.level")= num 0.95
 $ estimate   : Named num 4.95
  ..- attr(*, "names")= chr "mean of x"
 $ null.value : Named num 5
  ..- attr(*, "names")= chr "mean"
 $ stderr     : num 0.159
 $ alternative: chr "less"
 $ method     : chr "One Sample t-test"
 $ data.name  : chr "x"
 - attr(*, "class")= chr "htest"
```

```r
results$p.value
```

```
[1] 0.3813385
```

Matrix

```r
# Matrices

n=1:9
mat = matrix(n,nrow=3)
mat
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```r
y <- diag(n)

# use %*% as the product of matrices

## Eigenvalue and Eigenvector

A <- matrix(c(13, -4, 2, -4, 11, -2, 2, -2, 8), 3, 3, byrow=TRUE)
ev <- eigen(A)

(values <- ev$values)
```

```
[1] 17  8  7
```

```r
(vectors <- ev$vectors)
```

```
           [,1]       [,2]      [,3]
[1,]  0.7453560  0.6666667 0.0000000
[2,] -0.5962848  0.6666667 0.4472136
[3,]  0.2981424 -0.3333333 0.8944272
```

```r
## Data selection --row then column

mat[1, 1]
```

```
[1] 1
```

```r
mat[1,]
```

```
[1] 1 4 7
```

```r
mat[,1]
```

```
[1] 1 2 3
```

```r
class(mat[1, ]) # Note that the class of the returned object is no longer a matrix
```

```
[1] "integer"
```

Example 1

```
db_data <- list(
  drugs = list(
    general_information = data.frame(
      drugbank_id = c("DB001", "DB002", "DB003", "DB004", "DB005"),
      name = c("Aspirin", "Ibuprofen", "Paracetamol", "Insulin", "Morphine"),
      type = c("small molecule", "small molecule", "small molecule", "biotech", "small molecu
      created = as.Date(c("2020-01-01", "2020-02-01", "2020-03-01", "2020-04-01", "2020-05-0
      stringsAsFactors = FALSE
    ),
    drug_classification = data.frame(
      drugbank_id = c("DB001", "DB002", "DB003", "DB004", "DB005"),
      classification = c("Analgesic", "Anti-inflammatory", "Analgesic", "Hormone", "Analgesi
      stringsAsFactors = FALSE
    ),
    experimental_properties = data.frame(
      drugbank_id = c("DB001", "DB002", "DB003", "DB004", "DB005", "DB001", "DB002", "DB003"
      kind = c("logP", "logP", "logP", "logP", "logP", "Molecular Weight", "Molecular Weight
      value = c("1.2", "1.5", "0.8", "2.1", "1.8", "180.1", "206.3", "151.2", "5800.0", "281
      stringsAsFactors = FALSE
    )
  )
)
db_data
```

```
$drugs
$drugs$general_information
  drugbank_id        name           type    created
1       DB001     Aspirin small molecule 2020-01-01
2       DB002   Ibuprofen small molecule 2020-02-01
3       DB003 Paracetamol small molecule 2020-03-01
4       DB004     Insulin        biotech 2020-04-01
5       DB005    Morphine small molecule 2020-05-01

$drugs$drug_classification
  drugbank_id    classification
1       DB001         Analgesic
2       DB002 Anti-inflammatory
3       DB003         Analgesic
4       DB004           Hormone
```

```
5       DB005         Analgesic

$drugs$experimental_properties
   drugbank_id             kind  value
1         DB001             logP    1.2
2         DB002             logP    1.5
3         DB003             logP    0.8
4         DB004             logP    2.1
5         DB005             logP    1.8
6         DB001 Molecular Weight  180.1
7         DB002 Molecular Weight  206.3
8         DB003 Molecular Weight  151.2
9         DB004 Molecular Weight 5800.0
10        DB005 Molecular Weight  281.5
```

```r
general_information <- db_data$drugs$general_information

print(general_information)
```

```
  drugbank_id        name           type    created
1       DB001     Aspirin small molecule 2020-01-01
2       DB002   Ibuprofen small molecule 2020-02-01
3       DB003 Paracetamol small molecule 2020-03-01
4       DB004     Insulin        biotech 2020-04-01
5       DB005    Morphine small molecule 2020-05-01
```

```r
# 20. Number of drugs in the general_information dataframe
general_information <- db_data$drugs$general_information
nrow(general_information)
```

```
[1] 5
```

```r
# 21. Filter drugs of type "biotech"
general_information[general_information$type == 'biotech',]
```

```
  drugbank_id    name    type    created
4       DB004 Insulin biotech 2020-04-01
```

```r
# 22. Sort by the created column and display the first 5 rows
general_information$created <- as.Date(general_information$created)
sorted_df <- general_information[order(general_information$created), ]
head(sorted_df, 5)
```

```
  drugbank_id        name           type    created
1       DB001     Aspirin small molecule 2020-01-01
2       DB002   Ibuprofen small molecule 2020-02-01
3       DB003 Paracetamol small molecule 2020-03-01
4       DB004     Insulin        biotech 2020-04-01
5       DB005    Morphine small molecule 2020-05-01
```

```r
# 23. Subset with specific columns and display the first 5 rows
subset_df <- general_information[, c("drugbank_id", "name")]
head(subset_df, 5)
```

```
  drugbank_id        name
1       DB001     Aspirin
2       DB002   Ibuprofen
3       DB003 Paracetamol
4       DB004     Insulin
5       DB005    Morphine
```

```r
# 24. Merge dataframes and count rows
drug_classification <- db_data$drugs$drug_classification
merged_df <- merge(general_information, drug_classification, by = "drugbank_id")
nrow(merged_df)
```

```
[1] 5
```

```r
# 25. Count unique experimental properties (kind)
experimental_properties <- db_data$drugs$experimental_properties
unique_kinds <- unique(experimental_properties$kind)
length(unique_kinds)
```

```
[1] 2
```

```r
# 26. Filter for kind "logP" and count rows
logP_df <- experimental_properties[experimental_properties$kind == "logP", ]
nrow(logP_df)
```

```
[1] 5
```

```r
# 27. Convert value column to numeric and calculate mean
logP_df$value <- as.numeric(logP_df$value)
mean(logP_df$value, na.rm = TRUE)
```

```
[1] 1.48
```

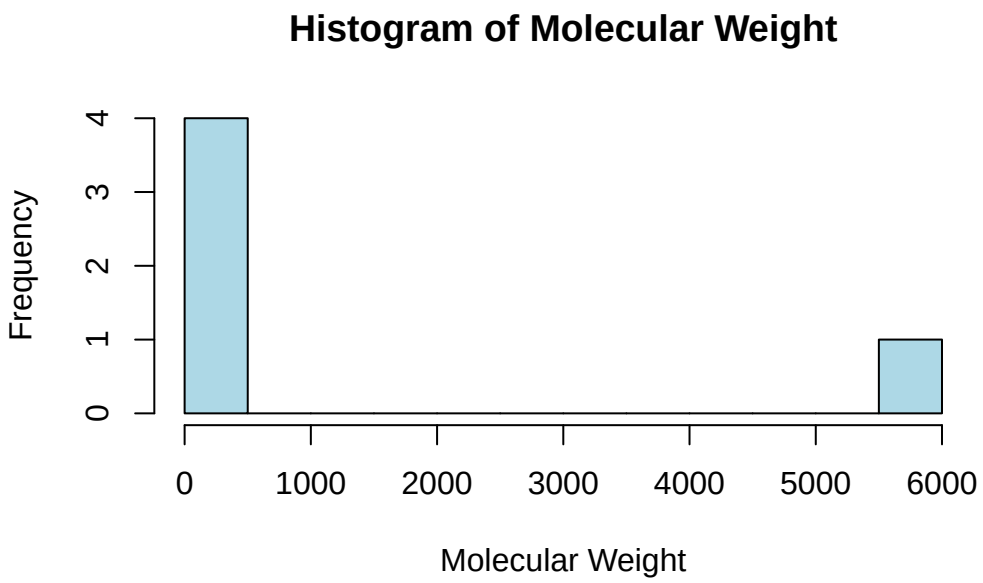
```r
# 28. Calculate summary statistics for logP values
summary(logP_df$value)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.80    1.20    1.50    1.48    1.80    2.10
```

```r
sd(logP_df$value, na.rm = TRUE)
```

```
[1] 0.5069517
```

```r
# 29. Create a histogram of molecular weight values
molecular_weight <- experimental_properties[experimental_properties$kind == "Molecular Weight
molecular_weight$value <- as.numeric(molecular_weight$value)
# clean based on 3 sigma rule
molecular_weight_clean <- molecular_weight[
  abs(molecular_weight$value - mean(molecular_weight$value, na.rm = TRUE)) <= 3 * sd(molecula
hist(molecular_weight_clean$value, main = "Histogram of Molecular Weight", xlab = "Molecular
```



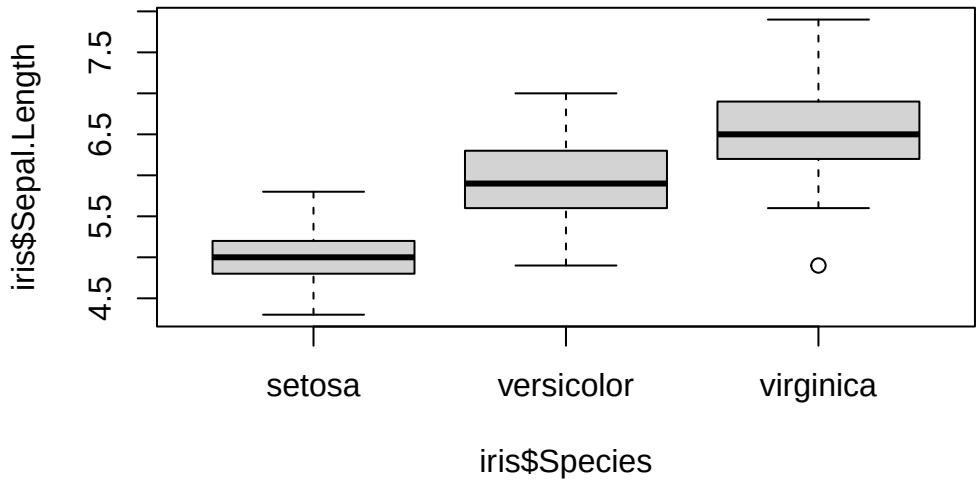

```
# 30. Filter for kind "Water Solubility" and count unique values
water_solubility_df <- experimental_properties[experimental_properties$kind == "Water Solubil
length(unique(water_solubility_df$value))
```

```
[1] 0
```

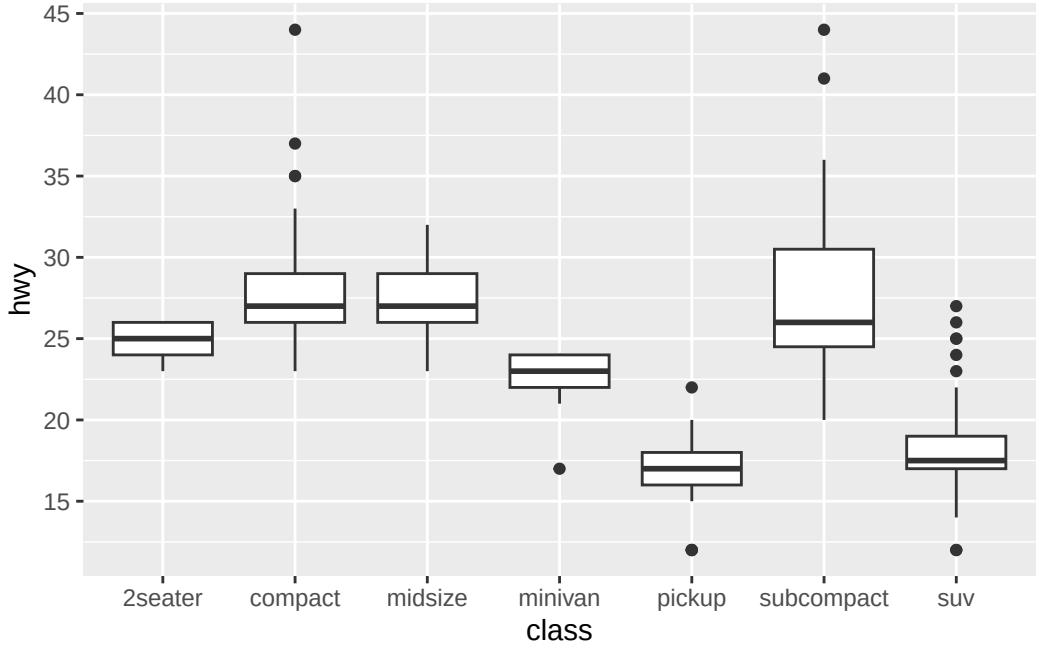
Data visualization (mainly: ggplot2)

Box plots

```
library(ggplot2)
data(iris)
boxplot(iris$Sepal.Length ~ iris$Species)
```



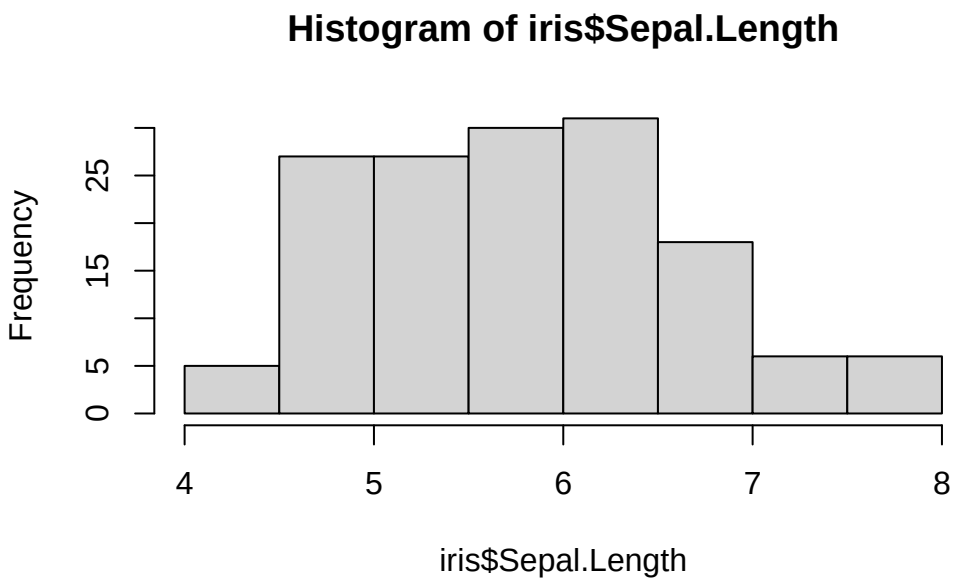

```
ggplot(mpg, aes(x=class, y=hwy)) +
  geom_boxplot() +
  scale_y_continuous(breaks = seq(10, 45, by=5))  #---diy scale in y-axis
```



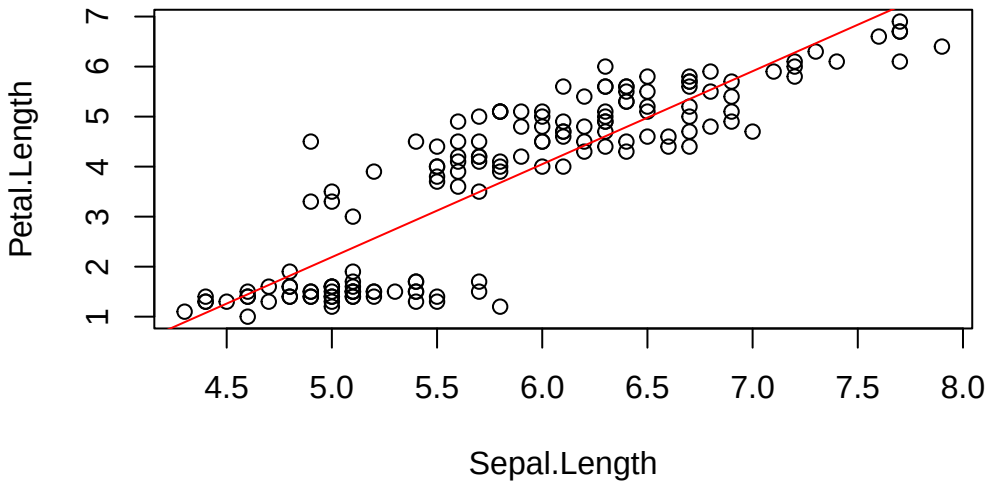

```
# ggplot(mpg, aes(x=class, y=hwy)) + geom_boxplot() +scale_y_continuous(breaks = seq(10, 45,
```

Histogram plots

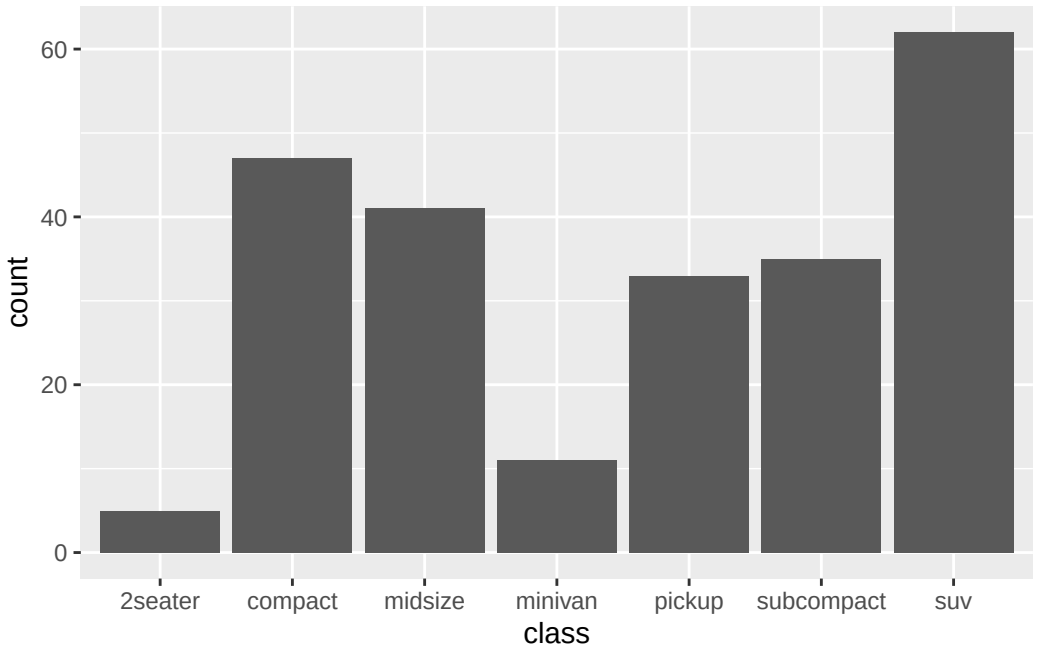
```
hist(iris$Sepal.Length)
```




```
plot(Petal.Length ~ Sepal.Length, data=iris)
abline(lm(Petal.Length ~ Sepal.Length, data=iris), col="red")
```




```
data(mpg, package='ggplot2')
ggplot(data=mpg, aes(x=class)) +
  geom_bar()
```



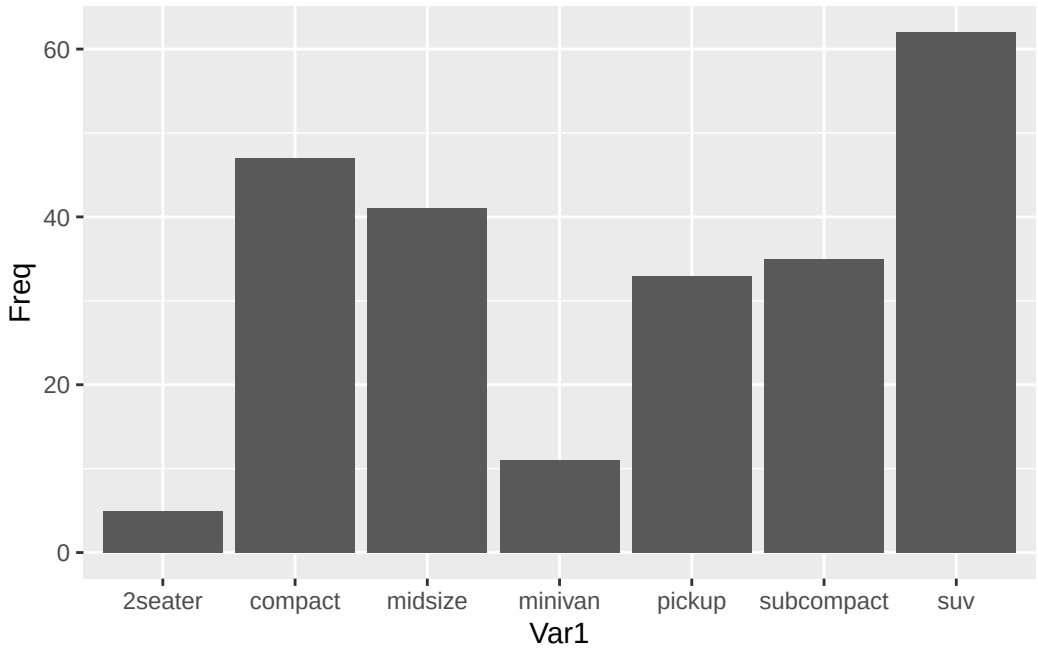

```
# By default, the geom_bar() just counts the number of cases and displays how many observatio
table(mpg$class)
```

```
   2seater    compact    midsize    minivan     pickup subcompact        suv
         5         47         41         11         33         35         62
```

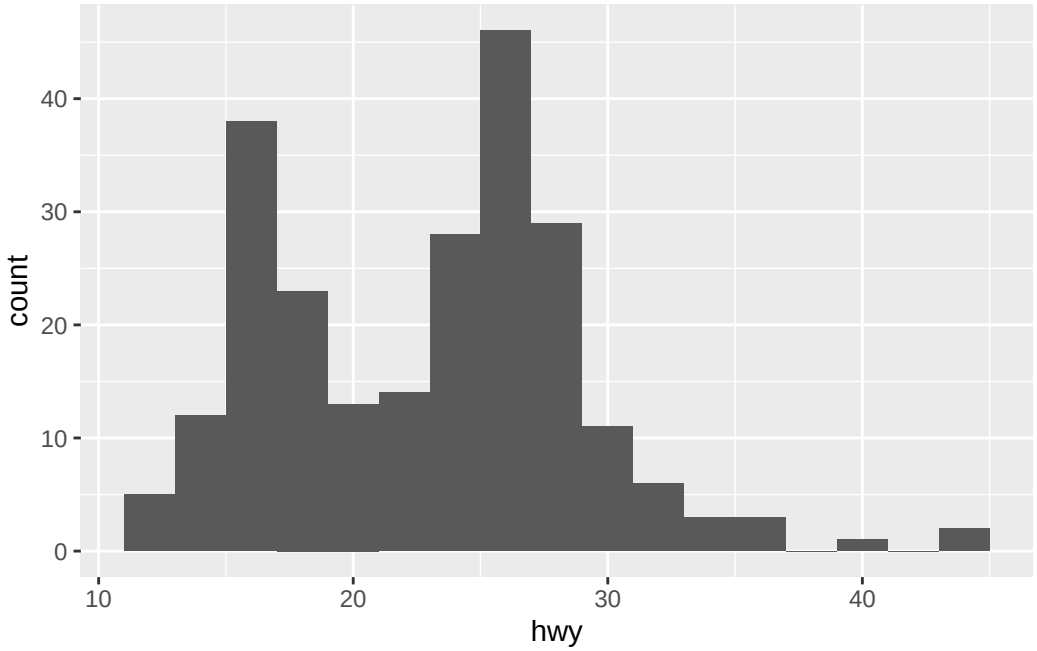
```
df <- as.data.frame(table(mpg$class))
df
```

```
        Var1 Freq
1    2seater    5
2    compact   47
3    midsize   41
4    minivan   11
5     pickup   33
6 subcompact   35
7        suv   62
```

```
ggplot(df, aes(Var1, Freq)) +
  geom_col()
```




```
ggplot(mpg,aes(x=hwy)) +geom_histogram(binwidth = 2)
```

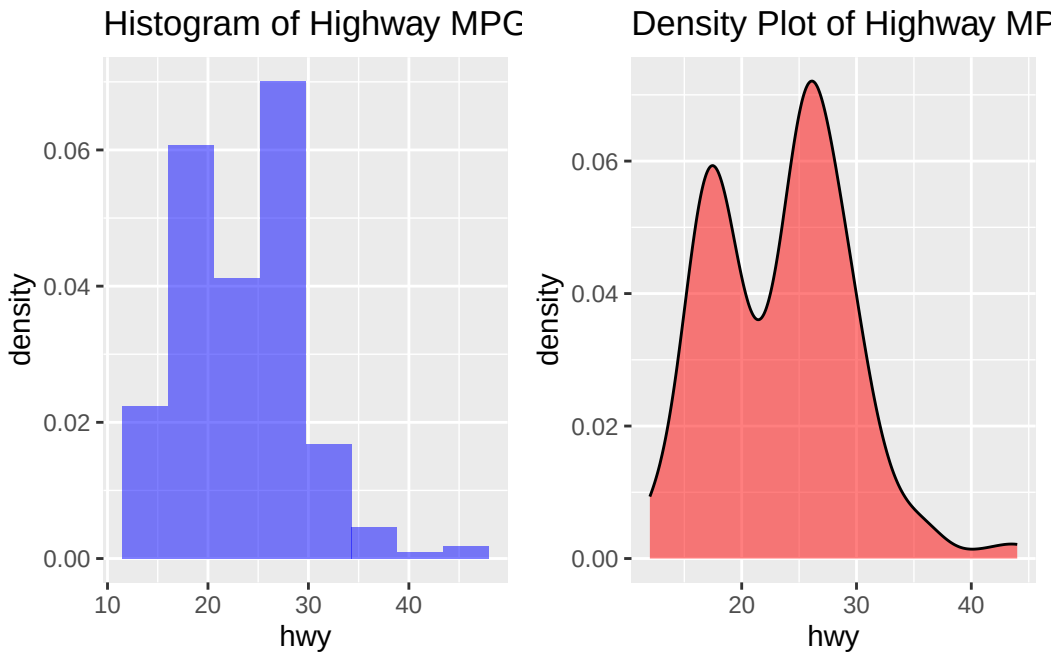


Density plots

```
p1 <- ggplot(mpg, aes(x=hwy, y=after_stat(density))) +
  geom_histogram(bins=8, fill="blue", alpha=0.5) +
  labs(title="Histogram of Highway MPG density")
p2 <- ggplot(mpg, aes(x=hwy)) +
  geom_density(fill='red', alpha=0.5) +
  labs(title="Density Plot of Highway MPG")
library(gridExtra)
```

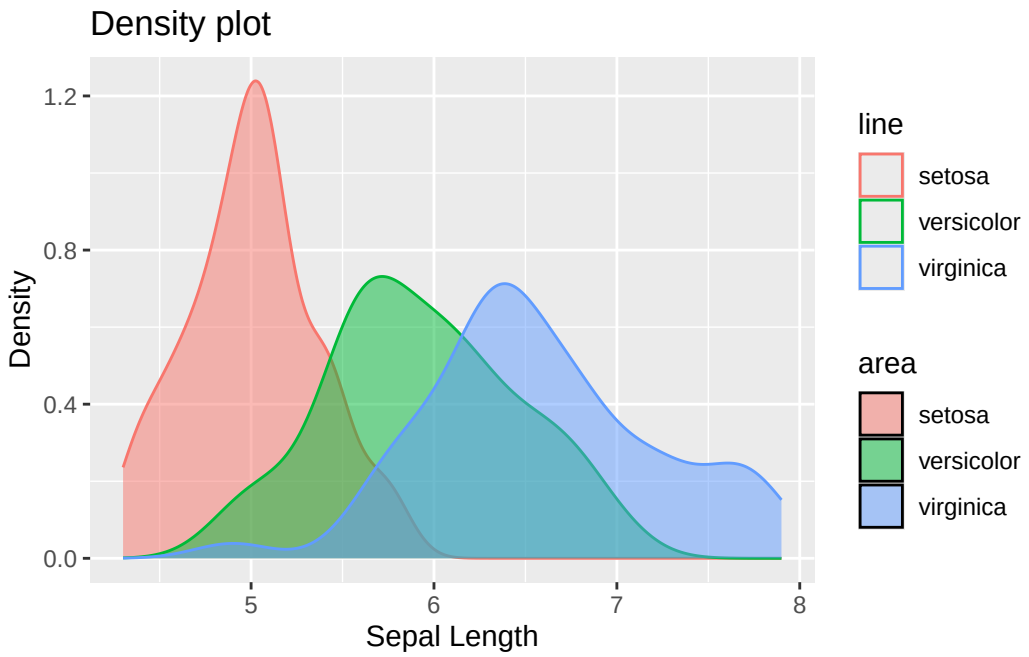
```
Attaching package: 'gridExtra'

The following object is masked from 'package:dplyr':

    combine
```

```
grid.arrange(p1, p2, ncol = 2)
```



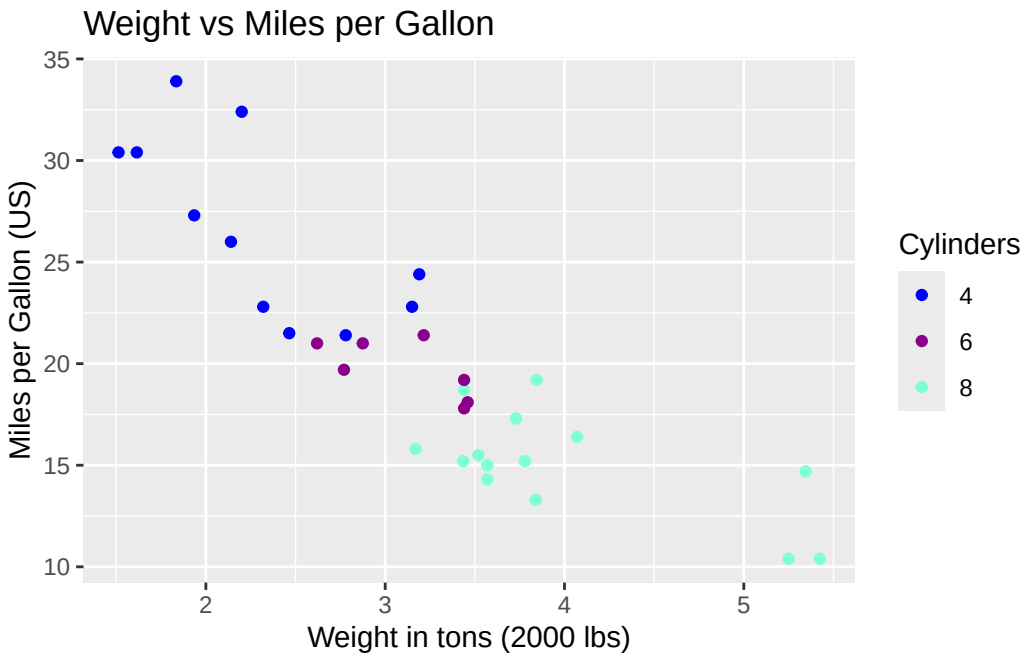
- mutiple plots in one figure

```
ggplot(iris, aes(x = Sepal.Length)) +
  geom_density(aes(fill = Species,color=Species), alpha = 0.5) +
  labs(title = "Density plot") +
  labs(x = "Sepal Length", y = "Density") +
  labs(fill = "area",color="line")  # fill is the area,color is the line or dot
```



Scatter plots

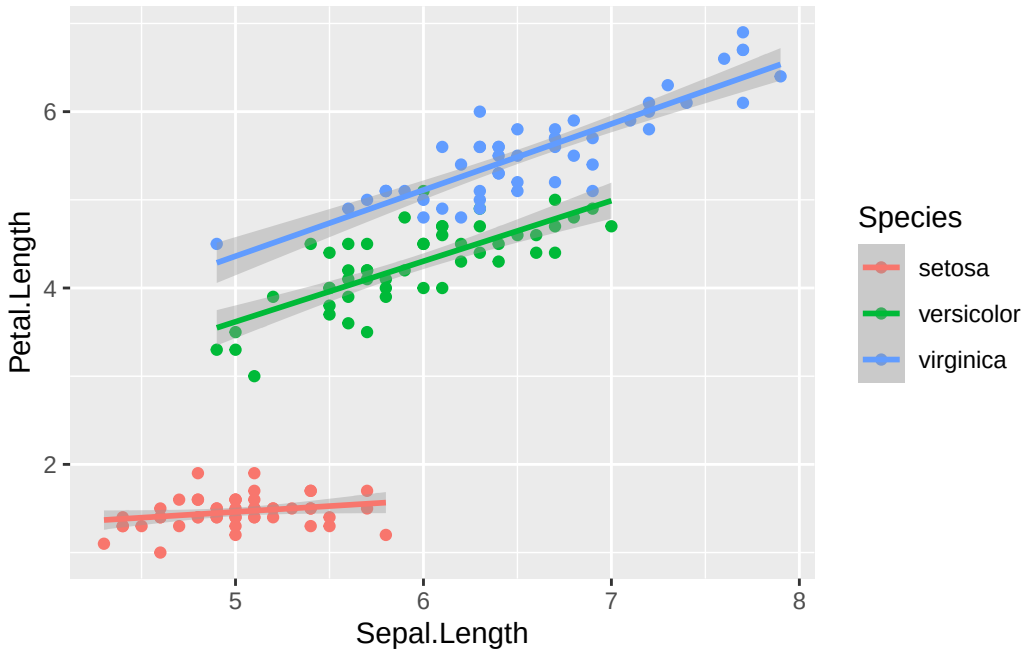
```
mtcars$cyl <- factor(mtcars$cyl)

ggplot(mtcars, aes(x=wt, y=mpg, col=cyl)) + geom_point() +
  labs(title='Weight vs Miles per Gallon') +
  labs(x="Weight in tons (2000 lbs)", y="Miles per Gallon (US)" ) +
  labs(color="Cylinders") + # color is dot or line
  scale_color_manual(values=c('blue', 'darkmagenta', 'aquamarine')) # diy color
```



Scatter plots with regression line

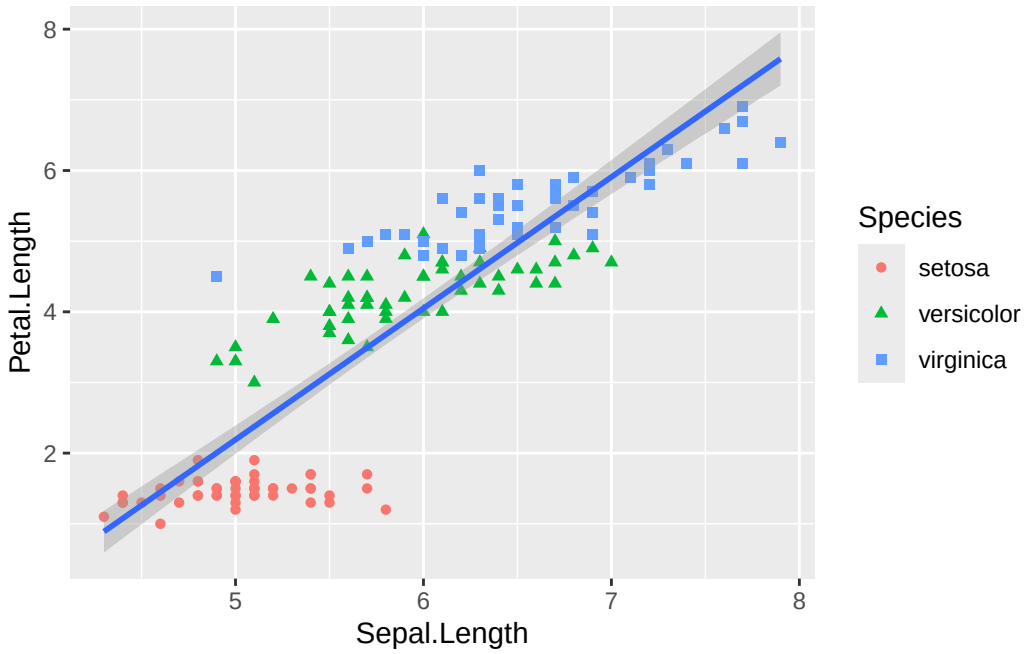
```
ggplot(data=iris, aes(x=Sepal.Length, y=Petal.Length,color=Species))+
  geom_point()+# Anything set inside an aes() command will be of the form attribute=Column_Na
   geom_smooth(method="lm") #By default, geom_smooth(method="lm") fits a linear regression li
```

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




```
ggplot(data=iris, aes(x=Sepal.Length, y=Petal.Length)) +
  geom_point(aes(color=Species,shape=Species))+
  geom_smooth(method="lm")
```

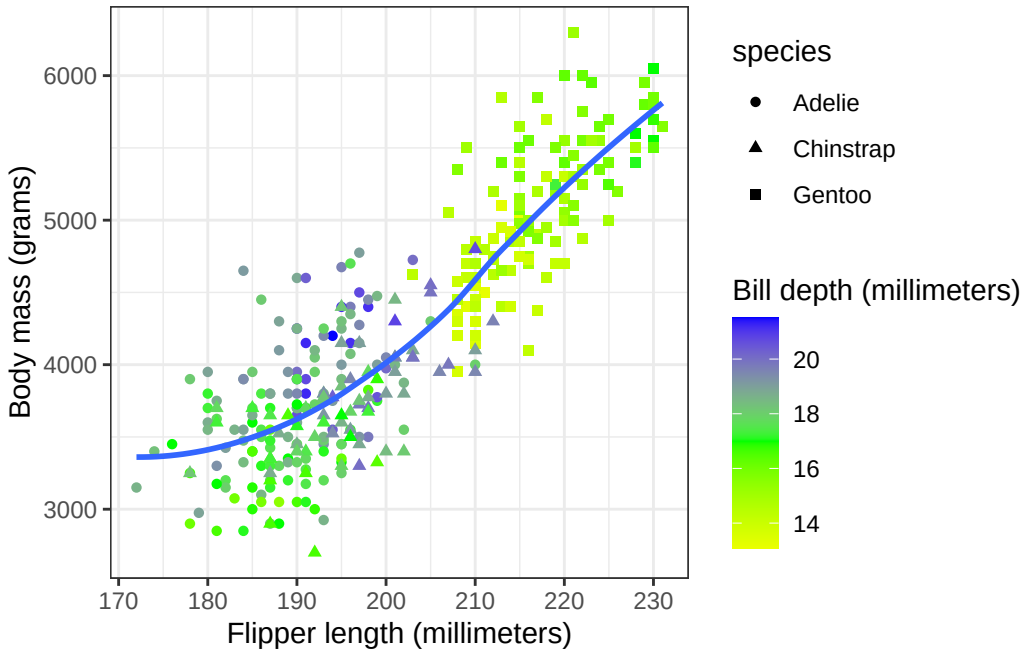
```
`geom_smooth()` using formula = 'y ~ x'
```




```
# Zooming in/out--Danger!  This removes the data points first!
#ggplot(trees, aes(x=Girth, y=Volume)) +
  #geom_point() +
  #geom_smooth(method='lm') +
  #xlim( 8, 19 ) + ylim(0, 60)
```

```
library(palmerpenguins)
ggplot(penguins, aes(x=flipper_length_mm, y=body_mass_g)) + geom_point(aes(color=bill_depth_m
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



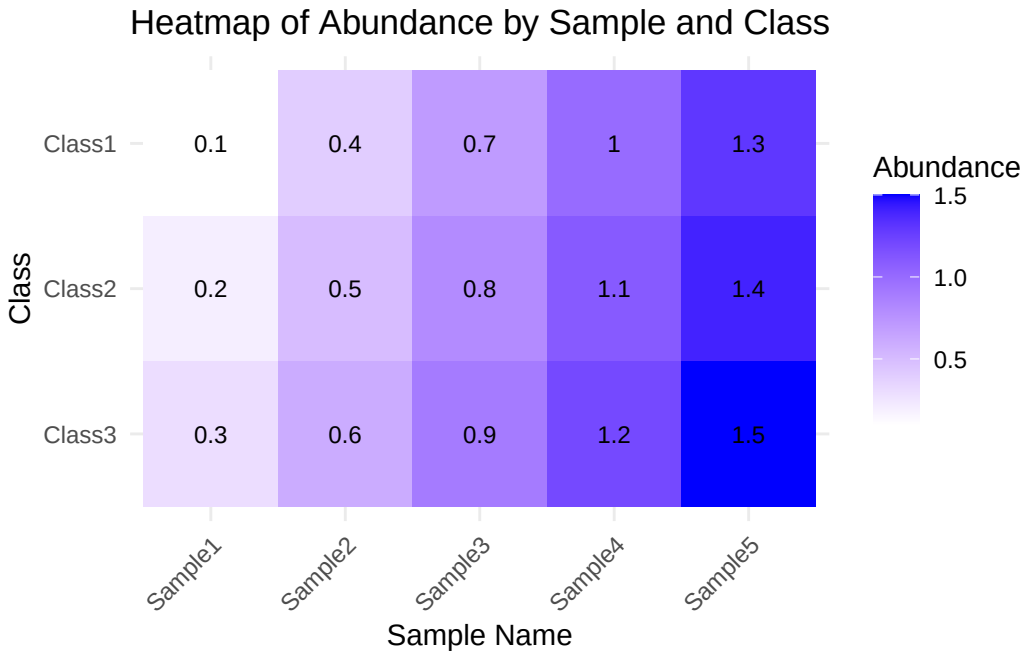
Heat map

```
# data
mine.table <- data.frame(
  Sample.name = rep(paste0("Sample", 1:5), each = 3),
  Class = rep(c("Class1", "Class2", "Class3"), times = 5),
  Abundance = c(0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5),
  Depth = c(0.5,0.5,0.5, 0.6,0.6,0.6, 0.7,0.7,0.7, 0.8,0.8,0.8, 0.9,0.9,0.9)
)
print(mine.table)
```

```
   Sample.name  Class Abundance Depth
1      Sample1 Class1       0.1   0.5
2      Sample1 Class2       0.2   0.5
3      Sample1 Class3       0.3   0.5
4      Sample2 Class1       0.4   0.6
5      Sample2 Class2       0.5   0.6
6      Sample2 Class3       0.6   0.6
7      Sample3 Class1       0.7   0.7
8      Sample3 Class2       0.8   0.7
9      Sample3 Class3       0.9   0.7
```

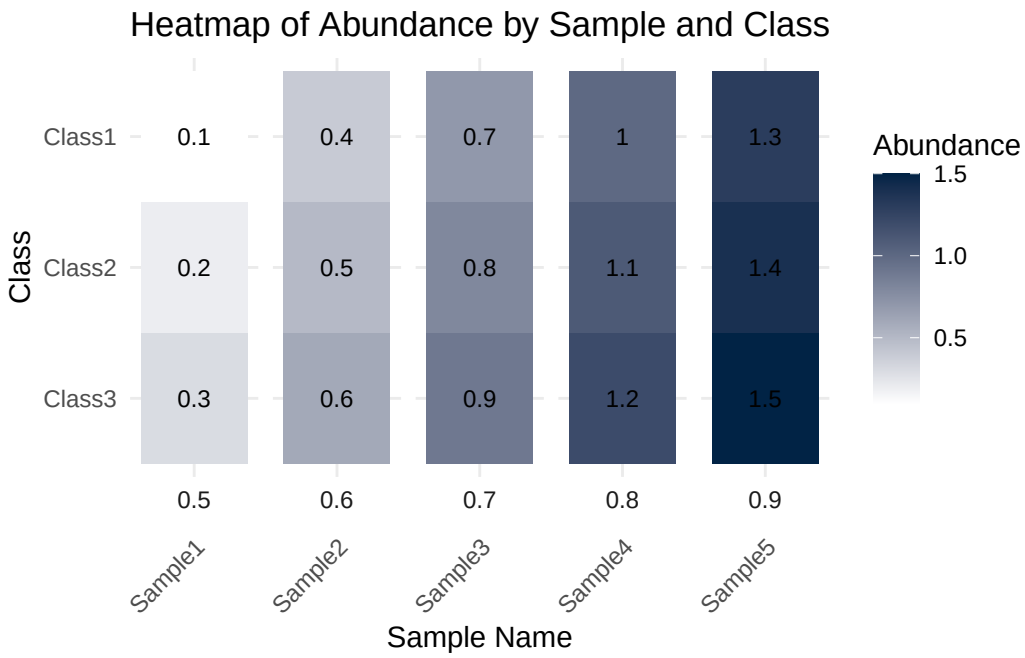
```
10     Sample4 Class1       1.0   0.8
11     Sample4 Class2       1.1   0.8
12     Sample4 Class3       1.2   0.8
13     Sample5 Class1       1.3   0.9
14     Sample5 Class2       1.4   0.9
15     Sample5 Class3       1.5   0.9
```

```r
mine.heatmap <- ggplot(data = mine.table, mapping = aes(x = Sample.name, y = Class, fill = Al
  geom_tile() + # create the heatmap with tiles+
  scale_y_discrete(limits = rev(levels(factor(mine.table$Class)))) +  # reverse the order of
  scale_fill_gradient(low = "white", high = "blue") +  # color
  theme_minimal() +  # control the theme of the plot
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +  # Rotate the X-axis label for l
  labs(x = "Sample Name",  # x-axis label
       y = "Class",  # y-axis label
       fill = "Abundance")+ # fill legend label by "Abundance"
       ggtitle("Heatmap of Abundance by Sample and Class")+ # add title
       geom_text(aes(label = round(Abundance, 2)), color = "black", size = 3) # add text labe


print(mine.heatmap)
```



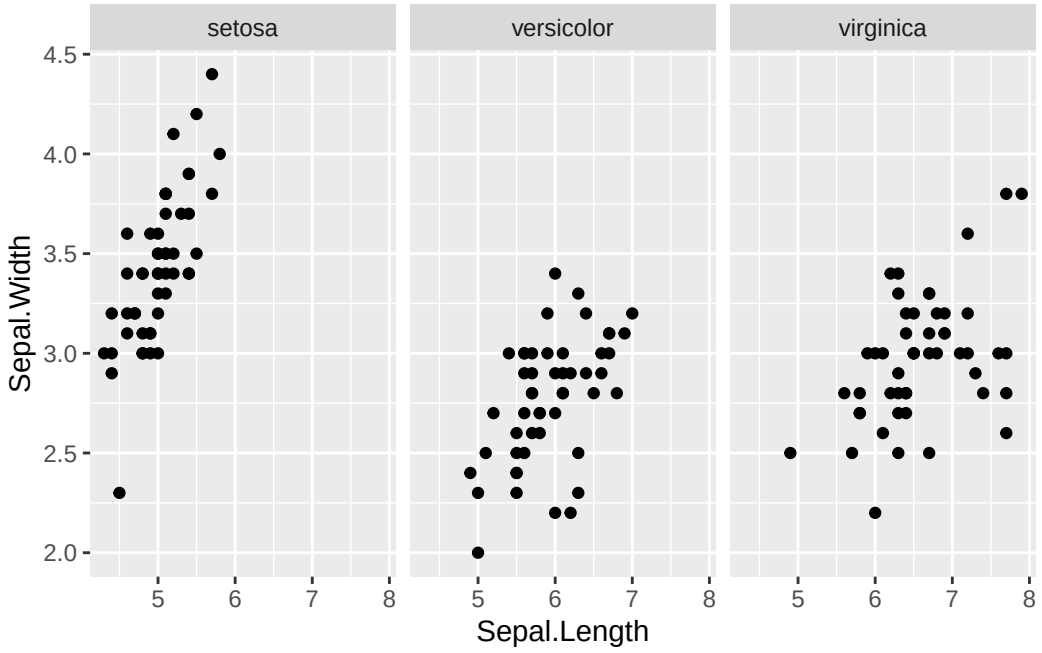
Create heat map using facet_grid to show the data in different panels by depth

```r
mine.heatmap <- ggplot(data = mine.table, mapping = aes(x = Sample.name, y = Class, fill = Al
  geom_tile() + # create the heatmap with tiles+
  facet_grid(~ Depth,switch = 'x', scales='free', space='free')+ # facet_grid to show the dat
  scale_y_discrete(limits = rev(levels(factor(mine.table$Class)))) +  # reverse the order of
  scale_fill_gradient(low="#FFFFFF", high="#012345")+  # color
  theme_minimal() +  # control the theme of the plot
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +  # Rotate the X-axis label for l
  labs(x = "Sample Name",  # x-axis label
       y = "Class",  # y-axis label
       fill = "Abundance")+ # fill legend label by "Abundance"
       ggtitle("Heatmap of Abundance by Sample and Class")+ # add title
       geom_text(aes(label = round(Abundance, 2)), color = "black", size = 3) # add text labe



print(mine.heatmap)
```



Faceting (make many panels of graphics where each panel represents the same relationship between variables, but something changes between each pane)

```
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point() +
  facet_grid(.~Species) #or facet_grid(Species~.)--Categorical variables of species will be v
```



- Another example

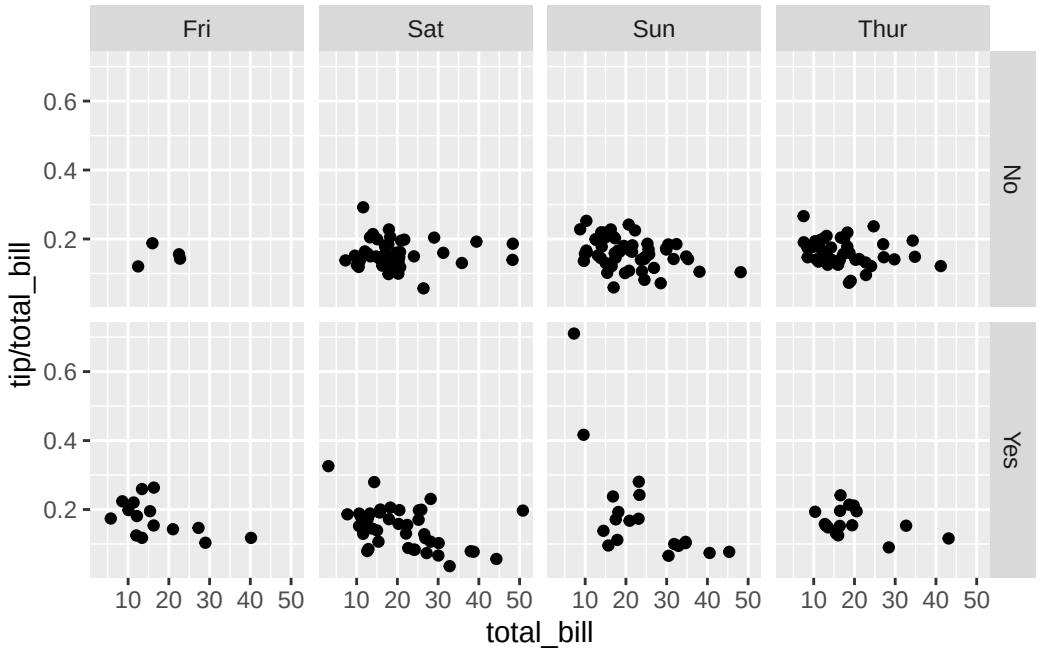
```
library(reshape)
```

```
Attaching package: 'reshape'

The following object is masked from 'package:dplyr':

    rename

The following object is masked from 'package:Matrix':

    expand
```

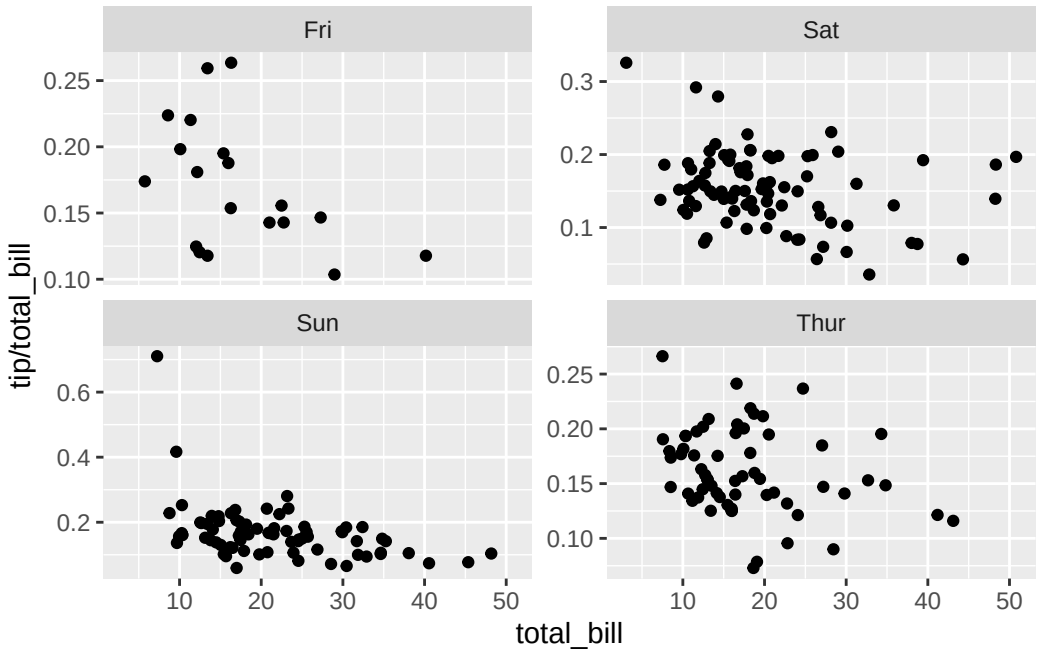
```
data(tips, package='reshape')
head(tips, 3)
```

```
  total_bill  tip    sex smoker day   time size
1      16.99 1.01 Female     No Sun Dinner    2
2      10.34 1.66   Male     No Sun Dinner    3
3      21.01 3.50   Male     No Sun Dinner    3
```

```
ggplot(tips, aes(x = total_bill, y = tip / total_bill)) +
  geom_point() +
  facet_grid( smoker ~ day )
```




```
# 'free_y' means the scale of different panels are adjusted by themselves
ggplot(tips, aes(x = total_bill, y = tip / total_bill)) +
  geom_point() +
  facet_wrap( ~ day, scales='free_y')
```




```
# log scales ---a wrapper of  scale_y_continuous() function , trans_new() function
# ggplot(ACS, aes(x=Age, y=Income)) + geom_point() +
# scale_y_log10(breaks=c(1, 10, 100),
#            minor=c(1:10,
#                 seq(10, 100, by=10 ),
#                seq(100, 1000, by=100))) +
#  ylab('Income (1000s of dollars)')
```

- Multi-plot

```
p1 <- ggplot(ChickWeight, aes(x=Time, y=weight, colour=Diet, group=Chick)) +
    geom_line() +
    ggtitle("Growth curve for individual chicks")
# Second plot
p2 <- ggplot(ChickWeight, aes(x=Time, y=weight, colour=Diet)) +
    geom_point(alpha=.3) +
    geom_smooth(alpha=.2, linewidth=1) +
    ggtitle("Fitted growth curve per diet")
# Third plot
p3 <- ggplot(subset(ChickWeight, Time==21), aes(x=weight, colour=Diet)) +
    geom_density() +
    ggtitle("Final weight, by diet")
```

```
# to realize:
# plot1 plot2 plot2
# plot1 plot2 plot2
# plot1 plot3 plot3

my.layout = cbind( c(1,1,1), c(2,2,3), c(2,2,3) ) # each c represents a column in a matrix ar
library(Rmisc)
```

```
Loading required package: plyr
```

```
------------------------------------------------------------------------------
```

```
You have loaded plyr after dplyr - this is likely to cause problems.
If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
library(plyr); library(dplyr)
```

```
------------------------------------------------------------------------------
```

```

Attaching package: 'plyr'
```

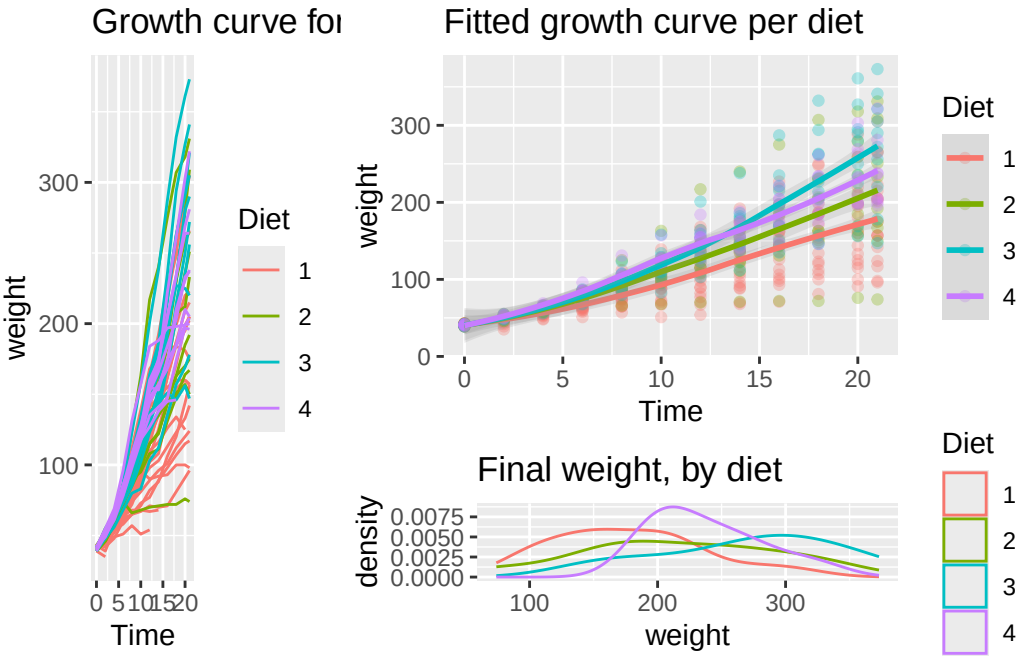
```
The following objects are masked from 'package:reshape':

    rename, round_any
```

```
The following object is masked from 'package:mosaic':

    count
```

```
The following objects are masked from 'package:dplyr':

    arrange, count, desc, failwith, id, mutate, rename, summarise,
    summarize
```

```
Rmisc::multiplot( p1, p2, p3, layout=my.layout)
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

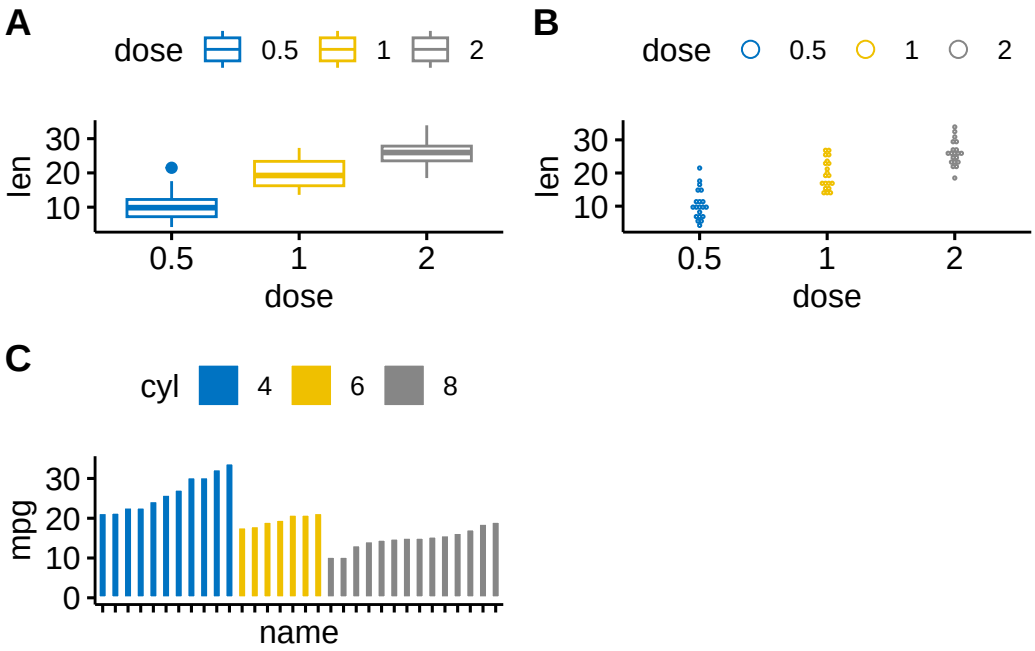



```
# OR library(ggpubr) https://rpkgs.datanovia.com/ggpubr/).

library(ggpubr)
```

```
Attaching package: 'ggpubr'

The following object is masked from 'package:plyr':

    mutate
```

```
# Box plot (bp)
bxp <- ggboxplot(ToothGrowth, x = "dose", y = "len",
                 color = "dose", palette = "jco")
# Dot plot (dp)
dp <- ggdotplot(ToothGrowth, x = "dose", y = "len",
                 color = "dose", palette = "jco", binwidth = 1)
mtcars$name <- rownames(mtcars)
mtcars$cyl <- as.factor(mtcars$cyl)
bp <- ggbarplot(mtcars, x = "name", y = "mpg",
          fill = "cyl",               # change fill color by cyl
          color = "white",            # Set bar border colors to white
```

```
          palette = "jco",            # jco journal color palett. see ?ggpar
          sort.val = "asc",           # Sort the value in ascending order
          sort.by.groups = TRUE,      # Sort inside each group
          x.text.angle = 90           # Rotate vertically x axis texts
          ) + font("x.text", size = 8)
# Scatter plots (sp)
sp <- ggscatter(mtcars, x = "wt", y = "mpg",
                add = "reg.line",               # Add regression line
                conf.int = TRUE,                # Add confidence interval
                color = "cyl", palette = "jco", # Color by groups "cyl"
                shape = "cyl"                   # Change point shape by groups "cyl"
                ) +
  stat_cor(aes(color = cyl), label.x = 3)       # Add correlation coefficient

ggarrange(bxp, dp, bp + rremove("x.text"),
          labels = c("A", "B", "C"),
          ncol = 2, nrow = 2)
```




```
# Themes
# Rmisc::multiplot( p1 + theme_bw(),          # Black and white
#                   p1 + theme_minimal(),
#                   p1 + theme_dark(),
#                   p1 + theme_light(),
```

```
#                   cols=2 )

#ggsave('p1.png', width=6, height=3, dpi=350)
```

Data manipulation

```
library(dplyr)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v stringr   1.5.1
v lubridate 1.9.4     v tibble    3.2.1
v purrr     1.0.2     v tidyr     1.3.1
v readr     2.1.5
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x plyr::arrange()      masks dplyr::arrange()
x gridExtra::combine() masks dplyr::combine()
x purrr::compact()     masks plyr::compact()
x plyr::count()        masks mosaic::count(), dplyr::count()
x purrr::cross()       masks mosaic::cross()
x plyr::desc()         masks dplyr::desc()
x mosaic::do()         masks dplyr::do()
x tidyr::expand()      masks reshape::expand(), Matrix::expand()
x plyr::failwith()     masks dplyr::failwith()
x dplyr::filter()      masks stats::filter()
x plyr::id()           masks dplyr::id()
x dplyr::lag()         masks stats::lag()
x ggpubr::mutate()     masks plyr::mutate(), dplyr::mutate()
x tidyr::pack()        masks Matrix::pack()
x dplyr::recode()      masks car::recode()
x plyr::rename()       masks reshape::rename(), dplyr::rename()
x purrr::some()        masks car::some()
x lubridate::stamp()   masks reshape::stamp()
x mosaic::stat()       masks ggplot2::stat()
x plyr::summarise()    masks dplyr::summarise()
x plyr::summarize()    masks dplyr::summarize()
x mosaic::tally()      masks dplyr::tally()
x tidyr::unpack()      masks Matrix::unpack()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

apply

```
# apply
# Summarize each column by calculating the mean.
apply(iris[,-5],        # what object do we want to apply the function to
      MARGIN=1,    # rows = 1, columns = 2, (same order as [rows, cols]
      FUN=mean     # what function do we want to apply
     ) %>% head(10)
```

```
 [1] 2.550 2.375 2.350 2.350 2.550 2.850 2.425 2.525 2.225 2.400
```

```
average <- apply(
  iris[,-5],        # what object do we want to apply the function to
  MARGIN=2,    # rows = 1, columns = 2, (same order as [rows, cols]
  FUN=mean     # what function do we want to apply
)
iris <- rbind(iris[,-5], average)
iris %>% head(3)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
```

There are several variants of the apply() function, and the most frequently used ones are lapply() and sapply(). These two functions apply a given function to each element of a list or vector and returns a corresponding list or vector of results.

```
#lapply
x <- list(a = 1:10, beta = exp(-3:3), logic = c(TRUE,FALSE,FALSE,TRUE))
x
```

```
$a
 [1]  1  2  3  4  5  6  7  8  9 10

$beta
[1]  0.04978707  0.13533528  0.36787944  1.00000000  2.71828183  7.38905610
[7] 20.08553692

$logic
[1]  TRUE FALSE FALSE  TRUE
```

```r
lapply(x, quantile, probs = 1:3/4) # list
```

```
$a
 25%  50%  75%
3.25 5.50 7.75

$beta
      25%       50%       75%
0.2516074 1.0000000 5.0536690

$logic
25% 50% 75%
0.0 0.5 1.0
```

```r
sapply(x, quantile, probs = 1:3/4) # matrix
```

```
       a      beta logic
25% 3.25 0.2516074   0.0
50% 5.50 1.0000000   0.5
75% 7.75 5.0536690   1.0
```

Tibbles

A tibble, or tbl_df, is a modern reimagining of the data.frame, keeping what time has proven to be effective, and throwing out what is not. Tibbles are data.frames that are lazy and surly: they do less (i.e. they don t change variable names or types, and don t do partial matching) and complain more (e.g. when a variable does not exist). This forces you to confront problems earlier, typically leading to cleaner, more expressive code. Tibbles also have an enhanced print() method which makes them easier to use with large datasets containing complex objects.

```r
data <- data.frame(a = 1:3, b = letters[1:3], c = Sys.Date() - 1:3)
data
```

```
  a b          c
1 1 a 2025-06-06
2 2 b 2025-06-05
3 3 c 2025-06-04
```

```
as_tibble(data)
```

```
# A tibble: 3 x 3
      a b     c
  <int> <chr> <date>
1     1 a     2025-06-06
2     2 b     2025-06-05
3     3 c     2025-06-04
```

%>%

The pipe operator %>% is used to pass the result of one function to the next function in a chain, making the code more readable and concise. For example, if we wanted to start with x, and first apply function f(), then g(), and then h(), the usual R command would be h(g(f(x))) which is hard to read because you have to start reading at the innermost set of parentheses. Using the pipe command %>%, this sequence of operations becomes x %>% f() %>% g() %>% h().

select

```
# Correct usage of select() within a pipeline
starwars %>% select(-ends_with('color'))
```

filter

```
library(dplyr)

# Filter rows where species is "Droid" and mass is greater than or equal to 100
filtered_data <- starwars %>% filter(species == "Droid", mass < 100)
print(filtered_data)
```

```
# A tibble: 3 x 14
  name  height  mass hair_color skin_color  eye_color birth_year sex   gender
  <chr>  <int> <dbl> <chr>      <chr>       <chr>          <dbl> <chr> <chr>
1 C-3PO    167    75 <NA>       gold        yellow           112 none  masculine
2 R2-D2     96    32 <NA>       white, blue red               33 none  masculine
```

```
3 R5-D4     97    32 <NA>       white, red  red               NA none  masculine
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

slice

This function is used to select rows by their position in the data frame. It can be used to select specific rows or a range of rows.

filter rows based on row number:

```
starwars %>% slice(2:4)
```

```
# A tibble: 3 x 14
  name      height  mass hair_color skin_color eye_color birth_year sex   gender
  <chr>      <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
1 C-3PO        167    75 <NA>       gold       yellow         112   none  mascu~
2 R2-D2         96    32 <NA>       white, bl~ red             33   none  mascu~
3 Darth Va~    202   136 none       white      yellow          41.9 male  mascu~
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

arrange

This function is used to sort the rows of a data frame by one or more columns. The default sorting of the number in the dataset is in ascending order, but you can use the `desc()` function to sort in descending order.

```
starwars %>% arrange(desc(name)) #The default sorting is in ascending order
```

```
# A tibble: 87 x 14
   name     height  mass hair_color skin_color eye_color birth_year sex   gender
   <chr>     <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
 1 Zam Wes~    168    55 blonde     fair, gre~ yellow            NA fema~ femin~
 2 Yoda         66    17 white      green      brown            896 male  mascu~
 3 Yarael ~    264    NA none       white      yellow            NA male  mascu~
 4 Wilhuff~    180    NA auburn, g~ fair       blue              64 male  mascu~
 5 Wicket ~     88    20 brown      brown      brown              8 male  mascu~
 6 Wedge A~    170    77 brown      fair       hazel             21 male  mascu~
 7 Watto       137    NA black      blue, grey yellow            NA male  mascu~
```

```
 8 Wat Tam~    193    48 none       green, gr~ unknown          NA male  mascu~
 9 Tion Me~    206    80 none       grey       black            NA male  mascu~
10 Taun We     213    NA none       grey       black            NA fema~ femin~
# i 77 more rows
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

```
starwars %>% arrange(desc(height)) %>% head(3)
```

```
# A tibble: 3 x 14
  name      height  mass hair_color skin_color eye_color birth_year sex   gender
  <chr>      <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
1 Yarael P~    264    NA none       white      yellow            NA male  mascu~
2 Tarfful      234   136 brown      brown      blue              NA male  mascu~
3 Lama Su      229    88 none       grey       black             NA male  mascu~
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

```
dd <- data.frame(
  Trt = factor(c("High", "Med", "High", "Low"),
               levels = c("Low", "Med", "High")), # level
  y = c(8, 3, 9, 9),
  z = c(1, 1, 1, 2))
dd %>% arrange(Trt, desc(y))
```

```
   Trt y z
1  Low 9 2
2  Med 3 1
3 High 9 1
4 High 8 1
```

mutate

This function is used to create new columns or modify existing columns in a data frame. It allows you to perform calculations and transformations on the data.

```
# select using the old columns
starwars$bmi = starwars$mass / ((starwars$height / 100) ^ 2)
starwars %>% select(name, bmi) %>% head(3)
```

```r
# mutate avoids all the starwars$

starwars$bmi <- NULL
starwars %>%
  mutate(bmi = mass / ((height / 100) ^ 2)) %>%
  select(name, bmi) %>% head(3)

# mutate_at() and mutate_if() allow us to apply a function to a particular column and save th

subset <- starwars %>%
  mutate(square_height = (height / 100) ^ 2,
         bmi = mass / square_height) %>%
  select(name, square_height, bmi)
subset %>% head(3)

subset %>% mutate_if(is.numeric, round, digits=0) # here, is.numeric is the condition

subset %>% mutate_at(2:3, round, digits=0) %>% head() # column 2 3

# Apply the transformation to columns 2 and 3 for rows 1 to 3
result <- subset %>%
  mutate_at(2:3, ~ifelse(row_number() %in% 1:3, round(., digits = 0), .))

subset %>% mutate(avg.example = select(., square_height:bmi) %>% rowMeans())
```

summarise

This function is used to create a summary table. It reduces the data frame to a single row containing summary statistics.

```r
starwars %>% summarise(mean.height=mean(height, na.rm=T), sd.height=sd(height, na.rm=T))

# apply the same statistic to each column
starwars %>% select(height:mass) %>% summarise_all(list(min=min, max=max), na.rm=T)
starwars %>% summarise_if(is.numeric, list(min=min, max=max), na.rm = T)
```

group_by

This function is used to group the data frame by one or more columns. It is often used in combination with `summarise()` to calculate summary statistics for each group.

```r
library(dplyr)
library(palmerpenguins)
table(penguins$sex, penguins$species)
penguins %>%
  filter(!is.na(sex)) %>%
  group_by(sex, species) %>%
  summarise(n = n(),
            mean.flipper = mean(flipper_length_mm),
            sd.flipper = sd(flipper_length_mm),
            .groups='keep') %>%
  head(3)
```

examples

Find the flight with the longest departure delay among flights from the same origin and destination (use `filter()`). Relocate the origin, destination, and departure delay to the first three columns and sort by origin and dest.

```r
flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(origin, dest) %>%
  filter(dep_delay == max(dep_delay)) %>%
  relocate(origin, dest, dep_delay) %>%
  arrange(origin, dest)
```

Find the flight with the longest departure delay among flights from the same origin and destination (use `top_n()` or `slice_max()`). Relocate the origin, destination, and departure delay to the first three columns and sort by origin and dest.

```r
flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(origin, dest) %>%
  top_n(1, dep_delay) %>%  # or using slice_max(dep_delay) %>%
  relocate(origin, dest, dep_delay) %>%
  arrange(origin, dest)
```

How do departure delays vary at different times of the day? Summarize the averaged departure delays by hours and create an new column named as `dep_delay_level` which `cut()` the averaged departure delays into three levels (low, median, and high).

```r
flights %>%
  group_by(hour) %>%
  summarize(avg_dep_delay = mean(dep_delay, na.rm = TRUE)) %>%
  mutate(dep_delay_level = cut(avg_dep_delay, breaks=3, c('low', 'median', 'high')))
```

How do departure delays vary at different times of the day? Illustrate your answer with a `geom_smooth()` plot.

```r
flights %>%
  group_by(hour) %>%
  summarize(avg_dep_delay = mean(dep_delay, na.rm = TRUE)) %>%
  ggplot(aes(x = hour, y = avg_dep_delay)) + geom_smooth()
```

```r
# ways to deleter the blanks
students %>%
  rename(
    student_id = `Student ID`,
    full_name = `Full Name`
  ) %>% head(3)
```

```r
read_csv(
  "# A comment I want to skip
  x,y,z
  1,2,3",
  comment = "#"
)
```

```r
library(tidyr)
grade.book <- rbind(
  data.frame(name='Alison',  HW.1=8, HW.2=5, HW.3=8, HW.4=4),
  data.frame(name='Brandon', HW.1=5, HW.2=3, HW.3=6, HW.4=9),
  data.frame(name='Charles', HW.1=9, HW.2=7, HW.3=9, HW.4=10))
grade.book
```

```
     name HW.1 HW.2 HW.3 HW.4
1  Alison    8    5    8    4
2 Brandon    5    3    6    9
3 Charles    9    7    9   10
```

```r
tidy.scores <- grade.book %>%
  pivot_longer(
    cols = starts_with("HW"),
    names_to = "Homework",
    values_to = "Score"
  )
tidy.scores
```

```
# A tibble: 12 x 3
   name    Homework Score
   <chr>   <chr>    <dbl>
 1 Alison  HW.1          8
 2 Alison  HW.2          5
 3 Alison  HW.3          8
 4 Alison  HW.4          4
 5 Brandon HW.1          5
 6 Brandon HW.2          3
 7 Brandon HW.3          6
 8 Brandon HW.4          9
 9 Charles HW.1          9
10 Charles HW.2          7
11 Charles HW.3          9
12 Charles HW.4         10
```

```r
tidy.scores %>% pivot_wider(names_from=Homework, values_from=Score)
```

```
# A tibble: 3 x 5
  name     HW.1  HW.2  HW.3  HW.4
  <chr>   <dbl> <dbl> <dbl> <dbl>
1 Alison      8     5     8     4
2 Brandon     5     3     6     9
3 Charles     9     7     9    10
```

```r
# table joins

Fish.Data <- tibble(
  Lake_ID = c('A','A','B','B','C','C'),
  Fish.Weight=rnorm(6, mean=260, sd=25) ) # make up some data
Fish.Data
```

```
# A tibble: 6 x 2
  Lake_ID Fish.Weight
  <chr>         <dbl>
1 A              279.
2 A              268.
3 B              269.
4 B              259.
5 C              261.
6 C              253.
```

```
Lake.Data <- tibble(
  Lake_ID = c('B','C','D'),
  Lake_Name = c('Lake Elaine', 'Mormon Lake', 'Lake Mary'),
  pH=c(6.5, 6.3, 6.1),
  area = c(40, 210, 240),
  avg_depth = c(8, 10, 38))
Lake.Data
```

```
# A tibble: 3 x 5
  Lake_ID Lake_Name      pH  area avg_depth
  <chr>   <chr>       <dbl> <dbl>     <dbl>
1 B       Lake Elaine   6.5    40         8
2 C       Mormon Lake   6.3   210        10
3 D       Lake Mary     6.1   240        38
```

```
full_join(Fish.Data, Lake.Data)
```

```
Joining with `by = join_by(Lake_ID)`
```

```
# A tibble: 7 x 6
  Lake_ID Fish.Weight Lake_Name      pH  area avg_depth
  <chr>         <dbl> <chr>       <dbl> <dbl>     <dbl>
1 A              279. <NA>         NA      NA        NA
2 A              268. <NA>         NA      NA        NA
3 B              269. Lake Elaine   6.5    40         8
4 B              259. Lake Elaine   6.5    40         8
5 C              261. Mormon Lake   6.3   210        10
6 C              253. Mormon Lake   6.3   210        10
7 D               NA  Lake Mary     6.1   240        38
```

```
left_join(Fish.Data, Lake.Data)
```

```
Joining with `by = join_by(Lake_ID)`
```

```
# A tibble: 6 x 6
  Lake_ID Fish.Weight Lake_Name      pH  area avg_depth
  <chr>         <dbl> <chr>       <dbl> <dbl>     <dbl>
1 A              279. <NA>         NA      NA        NA
2 A              268. <NA>         NA      NA        NA
3 B              269. Lake Elaine   6.5    40         8
4 B              259. Lake Elaine   6.5    40         8
5 C              261. Mormon Lake   6.3   210        10
6 C              253. Mormon Lake   6.3   210        10
```

```
inner_join(Fish.Data, Lake.Data)
```

```
Joining with `by = join_by(Lake_ID)`
```

```
# A tibble: 4 x 6
  Lake_ID Fish.Weight Lake_Name      pH  area avg_depth
  <chr>         <dbl> <chr>       <dbl> <dbl>     <dbl>
1 B              269. Lake Elaine   6.5    40         8
2 B              259. Lake Elaine   6.5    40         8
3 C              261. Mormon Lake   6.3   210        10
4 C              253. Mormon Lake   6.3   210        10
```

```
mutate(
    week = parse_number(week)
  )
```

how many data points are in the data set

```
gender_year <- Survey %>%
  filter(!is.na(Year)) %>%
  group_by(Sex, Year) %>%
  count() %>%
  rename(nu=n)
gender_year
gender_year %>% pivot_wider(names_from = Year, values_from = nu)
```

```r
who2 %>%
  head(3)
who2 <- who2 %>%
  pivot_longer(
    cols = !(country:year),
    names_to = c("diagnosis", "gender", "age"),
    names_sep = "_",
    values_to = "count"
  ) %>%
  filter(!is.na(count))
who2
```

```r
left_join(feb14_VX, airports, by=c('dest'='faa'))
```

```r
library(psych)
drug_prop <- drug_prop %>%
  filter(class == 'Carboxylic acids and derivatives')
drug_prop %>%
  select(logP, logS, water_solubility) %>%
  pairs.panels()
```

ps:the comparison between whether to use %>% or not

```r
# %>%
penguins %>%
  filter(!is.na(sex)) %>%
  group_by(sex, species) %>%
  mutate(Sum.Sq.Cells = (flipper_length_mm - mean(flipper_length_mm))^2)  %>%
  select(sex, species, flipper_length_mm, Sum.Sq.Cells) %>% head()

# not use %>%
head(
  select(mutate(group_by(filter(penguins, !is.na(sex)), sex, species),
                Sum.Sq.Cells = (flipper_length_mm - mean(flipper_length_mm))^2),
       sex, species, flipper_length_mm, Sum.Sq.Cells))
```

```r
library(nycflights13)
str(nycflights13::flights)
# the order of group_by and summarize matters
```

```r
flights %>%
  group_by(carrier) %>%
  summarize(avg_dep_delay = mean(dep_delay, na.rm = TRUE)) %>%
  arrange(desc(avg_dep_delay))
```

Control flow

while loop

```r
x <- 1
while (x < 10) {
  print(x)
  x <- x + 1
}
```

for loop

Fibonacci sequence

```r
F <- rep(0, 10)
F[1] <- 0
F[2] <- 1
cat('F = ', F, '\n')
```

```
F =  0 1 0 0 0 0 0 0 0 0
```

```r
for( n in 3:10 ){
  F[n] <- F[n-1] + F[n-2]
  cat('F = ', F, '\n')
}
```

```
F =  0 1 1 0 0 0 0 0 0 0
F =  0 1 1 2 0 0 0 0 0 0
F =  0 1 1 2 3 0 0 0 0 0
F =  0 1 1 2 3 5 0 0 0 0
F =  0 1 1 2 3 5 8 0 0 0
```

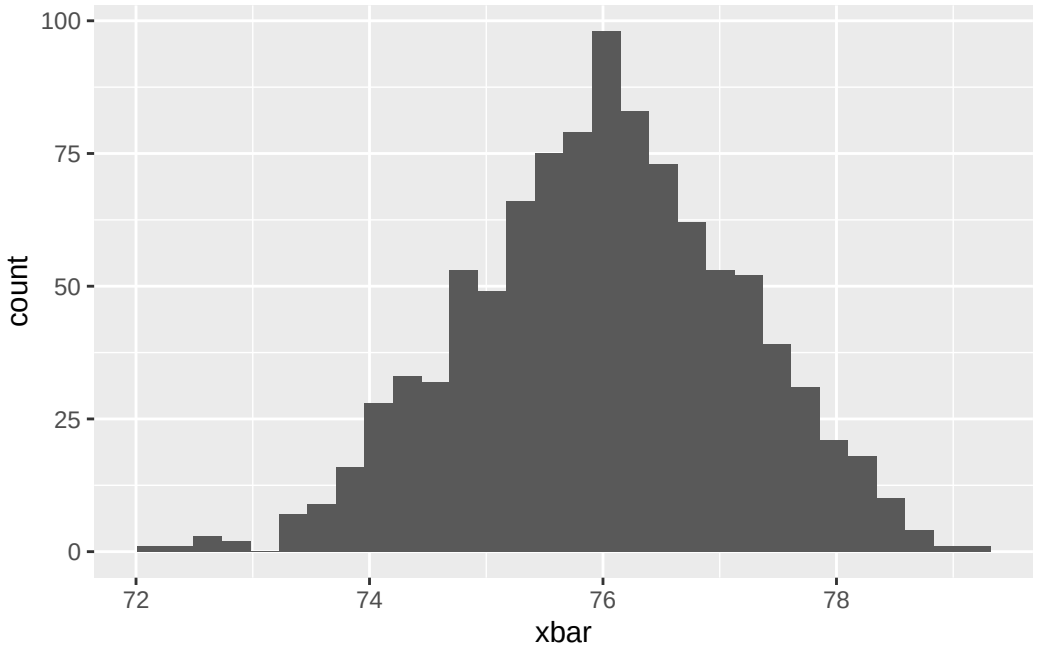
```
F =  0 1 1 2 3 5 8 13 0 0
F =  0 1 1 2 3 5 8 13 21 0
F =  0 1 1 2 3 5 8 13 21 34
```

bootstrap estimate of a sampling distribution

- bootstrap estimate of a sampling distribution

```
library(dplyr)
library(ggplot2)
SampDist <- data.frame()

for (i in 1:1000){
  SampDist <- trees %>%
    slice_sample(n=30, replace =TRUE) %>%
    dplyr::summarise(xbar=mean(Height)) %>%
    rbind(SampDist)
}
ggplot(SampDist,aes(x=xbar)) +
  geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Functions

Functions condtruction

copy of x

```
k <- 3
example.func <- function(x){
  x <- sort(x)
  if (k > 1){
    print(x)
  }
}
x <- c(3,1,5,4,2)
example.func(x)
```

```
[1] 1 2 3 4 5
```

```
x # x is changed inside the function but not outsied the function
```

```
[1] 3 1 5 4 2
```

Ellipses

```
# a function that draws the regression line and confidence interval
# notice it doesn't return anything, all it does is draw a plot
show.lm <- function(m, interval.type='confidence', fill.col='light grey', ...){
  x <- m$model[,2]       # extract the predictor variable
  y <- m$model[,1]       # extract the response
  pred <- predict(m, interval=interval.type)
  plot(x, y, ...)
  polygon( c(x,rev(x)),                          # draw the ribbon defined
           c(pred[,'lwr'], rev(pred[,'upr'])),   # by lwr and upr - polygon
           col='light grey')                     # fills in the region defined by
  lines(x, pred[, 'fit'])                        # a set of vertices, need to reverse
  points(x, y)                                   # the uppers to make a nice figure
}
```

Linear regression and multiple linear regression (Lab 10)

1. Load the `bloodpress.txt`

```
bloodpress <- read.table("bloodpress.txt", header=T)
bloodpress
```

```
   Pt  BP Age Weight  BSA  Dur Pulse Stress
1   1 105  47   85.4 1.75  5.1    63     33
2   2 115  49   94.2 2.10  3.8    70     14
3   3 116  49   95.3 1.98  8.2    72     10
4   4 117  50   94.7 2.01  5.8    73     99
5   5 112  51   89.4 1.89  7.0    72     95
6   6 121  48   99.5 2.25  9.3    71     10
7   7 121  49   99.8 2.25  2.5    69     42
8   8 110  47   90.9 1.90  6.2    66      8
9   9 110  49   89.2 1.83  7.1    69     62
10 10 114  48   92.7 2.07  5.6    64     35
11 11 114  47   94.4 2.07  5.3    74     90
12 12 115  49   94.1 1.98  5.6    71     21
13 13 114  50   91.6 2.05 10.2    68     47
14 14 106  45   87.1 1.92  5.6    67     80
15 15 125  52  101.3 2.19 10.0    76     98
16 16 114  46   94.5 1.98  7.4    69     95
17 17 106  46   87.0 1.87  3.6    62     18
18 18 113  46   94.5 1.90  4.3    70     12
19 19 110  48   90.5 1.88  9.0    71     99
20 20 122  56   95.7 2.09  7.0    75     99
```

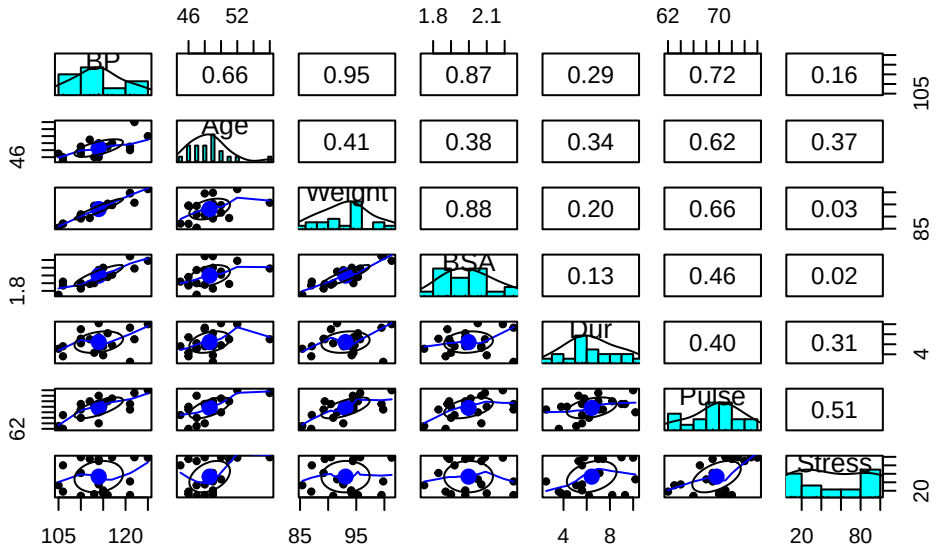
2. Use `pairs.panels()` function from `psych` pacakge to draw scatterplots, histograms, and calculate correlations between variables.

```
library(psych)
```

```
Attaching package: 'psych'

The following objects are masked from 'package:mosaic':

    logit, rescale
```

```
The following object is masked from 'package:car':

    logit

The following objects are masked from 'package:ggplot2':

    %+%, alpha
```

```
pairs.panels(bloodpress[, -1])
```



3. Fit a simple linear regression model of BP vs Stress. Is Stress significant?

```
model.1 <- lm(BP ~ Stress, data=bloodpress)
summary(model.1)
```

```
Call:
lm(formula = BP ~ Stress, data = bloodpress)

Residuals:
    Min      1Q  Median      3Q     Max
-8.6394 -3.3014  0.0722  2.2181  9.9287

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept) 112.71997    2.19345  51.389   <2e-16 ***
Stress        0.02399    0.03404   0.705     0.49
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.502 on 18 degrees of freedom
Multiple R-squared:  0.02686,   Adjusted R-squared:  -0.0272
F-statistic: 0.4969 on 1 and 18 DF,  p-value: 0.4899
```

4. Fit a simple linear regression model of BP vs Weight.

```
model.2 <- lm(BP ~ Weight, data=bloodpress)
summary(model.2)
```

```
Call:
lm(formula = BP ~ Weight, data = bloodpress)

Residuals:
    Min      1Q  Median      3Q     Max
-2.6933 -0.9318 -0.4935  0.7703  4.8656

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.20531    8.66333   0.255    0.802
Weight       1.20093    0.09297  12.917 1.53e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.74 on 18 degrees of freedom
Multiple R-squared:  0.9026,    Adjusted R-squared:  0.8972
F-statistic: 166.9 on 1 and 18 DF,  p-value: 1.528e-10
```

5. Fit a simple linear regression model of BP vs BSA.

```
model.3 <- lm(BP ~ BSA, data=bloodpress)
summary(model.3)
```

```
Call:
lm(formula = BP ~ BSA, data = bloodpress)
```

```
Residuals:
   Min     1Q Median     3Q    Max
-5.314 -1.963 -0.197  1.934  4.831

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   45.183      9.392   4.811  0.00014 ***
BSA           34.443      4.690   7.343 8.11e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.79 on 18 degrees of freedom
Multiple R-squared:  0.7497,    Adjusted R-squared:  0.7358
F-statistic: 53.93 on 1 and 18 DF,  p-value: 8.114e-07
```

6. Fit a multiple linear regression model of BP vs Weight + BSA. Is BSA still significant? Why?

```r
model.4 <- lm(BP ~ Weight + BSA, data=bloodpress)
summary(model.4)
```

```
Call:
lm(formula = BP ~ Weight + BSA, data = bloodpress)

Residuals:
    Min      1Q  Median      3Q     Max
-1.8932 -1.1961 -0.4061  1.0764  4.7524

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.6534     9.3925   0.602    0.555
Weight        1.0387     0.1927   5.392 4.87e-05 ***
BSA           5.8313     6.0627   0.962    0.350
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.744 on 17 degrees of freedom
Multiple R-squared:  0.9077,    Adjusted R-squared:  0.8968
F-statistic: 83.54 on 2 and 17 DF,  p-value: 1.607e-09
```

8. Predict BP for Weight=92 and BSA=2 for the two simple linear regression models and the multiple linear regression model, by hand and by `predict()` function.

```
2.20531 + 1.20093 * 92
```

```
[1] 112.6909
```

```
predict(model.2,
        newdata=data.frame(Weight=92))
```

```
      1
112.691
```

```
45.183 + 34.443 * 2
```

```
[1] 114.069
```

```
predict(model.3,
        newdata=data.frame(BSA=2))
```

```
       1
114.0689
```

```
5.6534 + 1.0387 * 92 + 5.8313 * 2
```

```
[1] 112.8764
```

```
predict(model.4,
        newdata=data.frame(Weight=92, BSA=2))
```

```
       1
112.8794
```

7. Fit a multiple linear regression model of BP vs Age + Weight. Set argument `x` and `y` as TRUE. Save the output of `lm()` as `model.5`. How do we interpret each estimated coefficients?

```
model.5 <- lm(BP ~ Age + Weight, data=bloodpress, x=TRUE, y=TRUE)
summary(model.5)
```

```
Call:
lm(formula = BP ~ Age + Weight, data = bloodpress, x = TRUE,
    y = TRUE)

Residuals:
     Min       1Q   Median       3Q      Max
-0.89968 -0.35242  0.06979  0.35528  0.82781

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -16.57937    3.00746  -5.513 3.80e-05 ***
Age           0.70825    0.05351  13.235 2.22e-10 ***
Weight        1.03296    0.03116  33.154  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5327 on 17 degrees of freedom
Multiple R-squared:  0.9914,    Adjusted R-squared:  0.9904
F-statistic: 978.2 on 2 and 17 DF,  p-value: < 2.2e-16
```

8. Use the `plot_ly` function in the `plotly` package to create a 3D scatterplot of the data with the fitted plane for a multiple linear regression model of BP vs Age + Weight.

```
library(plotly)
plot_ly(x=bloodpress$Age, y=bloodpress$Weight, z=bloodpress$BP, type='scatter3d', mode='marke
```

9. Extract the matrix `x` and `y` of `model.5` and assign it to a new object `X` and `y`. Remember, if you save the output of `lm()` as an object, this object contains many elements. After we set `x=TRUE` and `y=TRUE` in question 8, we can find `x` and `y` in this list.

```
X <- model.5$x
y <- model.5$y
```

10. Calculate X^TX, X^Ty, $(X^TX)^{-1}$, and $(X^TX)^{-1}X^Ty$. Use `t()` for transpose, `%*%` for matrix multiplication, and `solve()` for inverse of matrix. For the last one, is your result same as the estimated values you obtained in question 7? –Of course!

```
t(X) %*% X
```

```
            (Intercept)     Age   Weight
(Intercept)        20.0   972.0   1861.8
Age               972.0 47358.0  90566.6
Weight           1861.8 90566.6 173665.4
```

```
t(X) %*% y
```

```
               [,1]
(Intercept)   2280.0
Age         110978.0
Weight      212666.1
```

```
solve(t(X) %*% X)
```

```
            (Intercept)          Age       Weight
(Intercept)  31.8748075 -0.267669593 -0.202127676
Age          -0.2676696  0.010092130 -0.002393468
Weight       -0.2021277 -0.002393468  0.003420885
```

```
solve(t(X) %*% X) %*% (t(X) %*% y)
```

```
                 [,1]
(Intercept) -16.5793694
Age           0.7082515
Weight        1.0329611
```

11. Use the anova function to display the ANOVA table with sequential (type I) sums of squares for the `model.5`.

$SS_{\text{Variable}} = \sum_{i=1}^{n} (\hat{y}_{\text{Variable},i} - \bar{y})^2$

$\hat{y}_{\text{Variable},i}$ is the model including only varible i.

F is the ratio of the mean square for the variable to the mean square for residuals.

If (F $\approx$ 1) : It indicates that the sizes of MS_Variable and MS_Residuals are approximately the same, suggesting that the explanatory power of the independent variable for the dependent variable is comparable to the random error, and the null hypothesis ((H_0)) cannot be rejected.

If (F $\gg$ 1) : It indicates that MS_Variable is significantly greater than MS_Residuals, suggesting that the independent variable has a significant influence on the dependent variable, and the null hypothesis ((H_0)) can be rejected.

```r
anova(model.5)
```

```
Analysis of Variance Table

Response: BP
          Df  Sum Sq Mean Sq F value    Pr(>F)
Age        1 243.266 243.266  857.29 5.481e-16 ***
Weight     1 311.910 311.910 1099.20 < 2.2e-16 ***
Residuals 17   4.824   0.284
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# remark
sum((model.5$y-mean(model.5$y))^2) == 243.266+311.910+4.824
```

```
[1] TRUE
```

$SS_{\text{Total}} = SS_{\text{Age}} + SS_{\text{Weight}} + SS_{\text{Residuals}}$

12. Use the `residuals` element in fitted model or `residuals()` function to extract the fitted residuals. Calculate the sum of square of these residual values. Extract the `df.residual` element in fitted model and use the above elements to calculate the MSE. Is your result same as the `anova()` output?

```r
sum((model.5$residuals)^2)/model.5$df.residual
```

```
[1] 0.2837604
```

13. Fit a multiple linear regression model of BP vs Age + Weight + Pulse. Save the output of `lm()` as `model.6`.

```r
model.6 <- lm(BP ~ Age + Weight + Pulse, data=bloodpress)
summary(model.6)
```

```
Call:
lm(formula = BP ~ Age + Weight + Pulse, data = bloodpress)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-0.71174 -0.45422 -0.01909  0.41745  0.88743

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -16.69000    2.93761  -5.681 3.40e-05 ***
Age           0.75018    0.06074  12.350 1.36e-09 ***
Weight        1.06135    0.03695  28.722 3.40e-15 ***
Pulse        -0.06566    0.04852  -1.353    0.195
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5201 on 16 degrees of freedom
Multiple R-squared:  0.9923,    Adjusted R-squared:  0.9908
F-statistic: 684.7 on 3 and 16 DF,  p-value: < 2.2e-16
```

14. Use `anova()` function to obtain the ANOVA table for `model.6`. We may consider `model.6` as full model, and `model.5` as reduced model in this question. Based on the obtained ANOVA table and the output of question 11, calculate the F-statistic for testing the reduced model by hand. You may use the Residuals Sum sq and the corresponding Residuals Df from both tables. Then, calculate the p-value using $pf()$ function, don't forget about the `lower.tail`.

```
anova(model.6)
```

```
Analysis of Variance Table

Response: BP
          Df  Sum Sq Mean Sq   F value    Pr(>F)
Age        1 243.266 243.266  899.2446 1.726e-15 ***
Weight     1 311.910 311.910 1152.9909 2.433e-16 ***
Pulse      1   0.496   0.496    1.8319    0.1947
Residuals 16   4.328   0.271
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Fstat <- (4.824-4.328)/(17-16) / (4.328/16)
Fstat
```

```
[1] 1.833641
```

```r
pf(Fstat, 1, 16, lower.tail = F)
```

```
[1] 0.1945157
```

15. Use `anova()` function to do the F-test on `model.5` and `model.6`. Compare the output with your answers of question 14. What is the conclustion of the F-test?

$\text{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Sum of Sq = RSS(Model 1) − RSS(Model 2) (It represents the variation in the interpretation of the dependent variable by the newly added variable Pulse.)

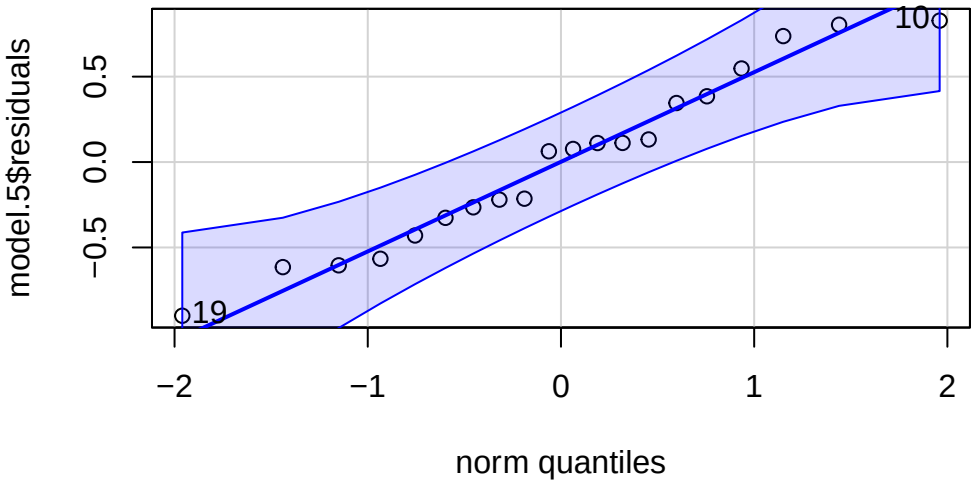
$$F = \frac{\text{Sum of Sq/Df}}{\text{RSS(Model 2)/Res.Df(Model 2)}} F = \frac{0.49557}{0.270525} \approx 1.8319$$

```r
anova(model.5, model.6)
```

```
Analysis of Variance Table

Model 1: BP ~ Age + Weight
Model 2: BP ~ Age + Weight + Pulse
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     17 4.8239
2     16 4.3284  1   0.49557 1.8319 0.1947
```

16. Plot the qqPlot for residuals of `model.5`. What is the x-axis and y-axis of the qqPlot? What can we say about the qqPlot?

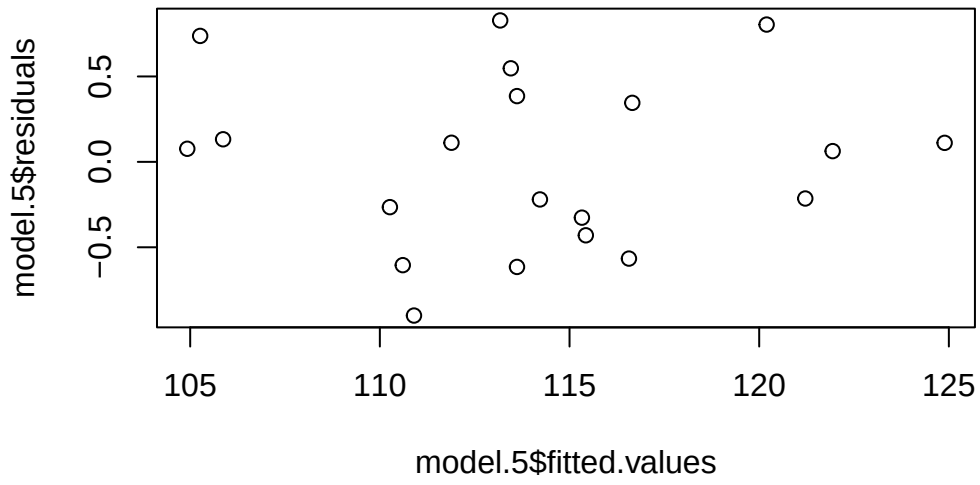
```r
library(car)
qqPlot(model.5$residuals)
```



```
[1] 19 10
```

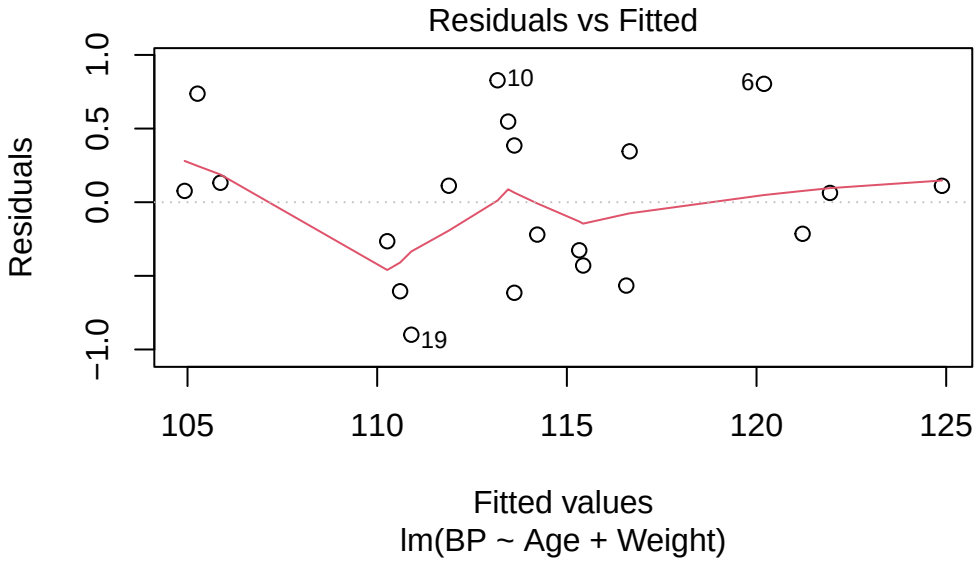
17. Plot the residual vs fitted plot of `model.5`. You may extract `fitted.values` from `model.5` and use it as `x` in `plot()`.

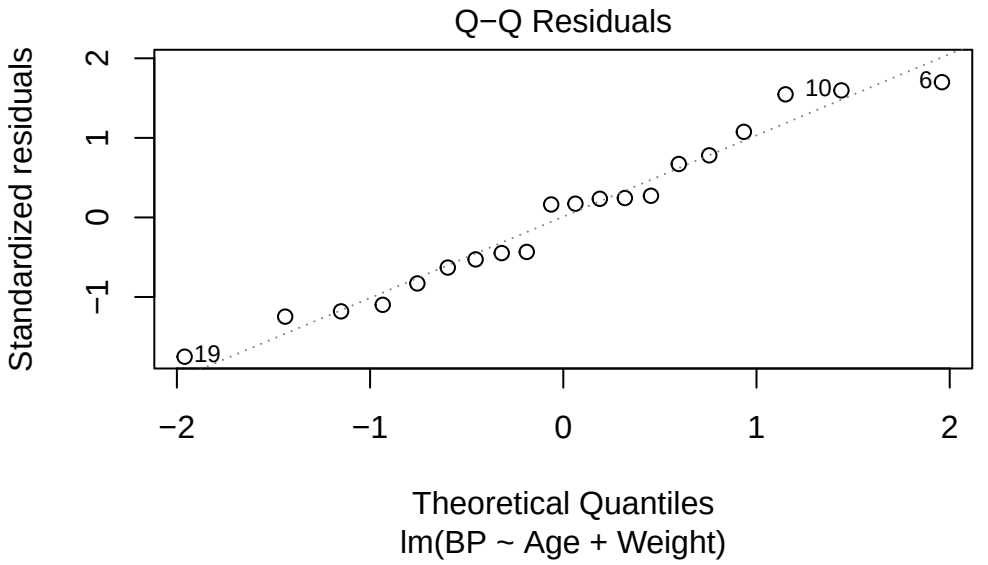
```
plot(x=model.5$fitted.values, y=model.5$residuals)
```

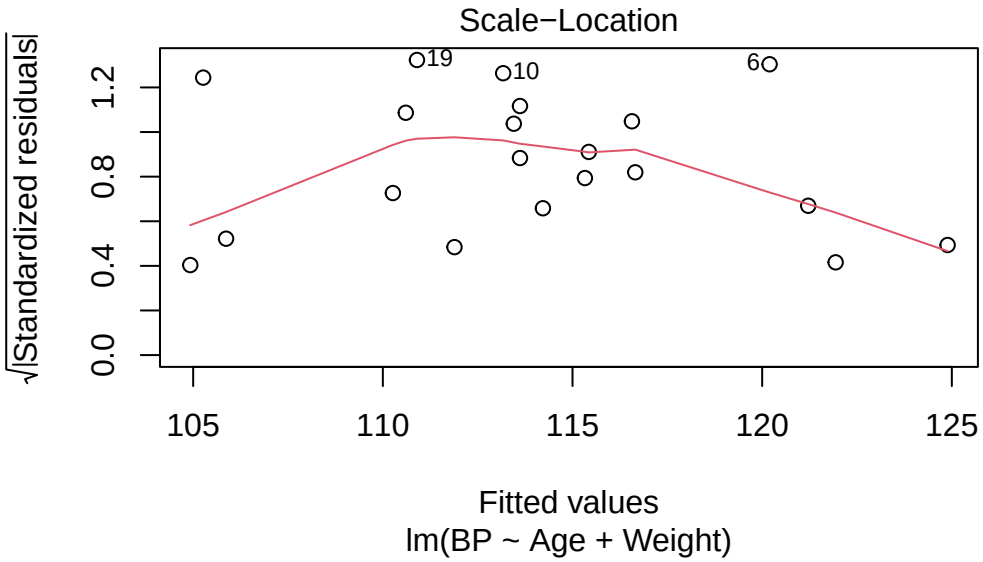


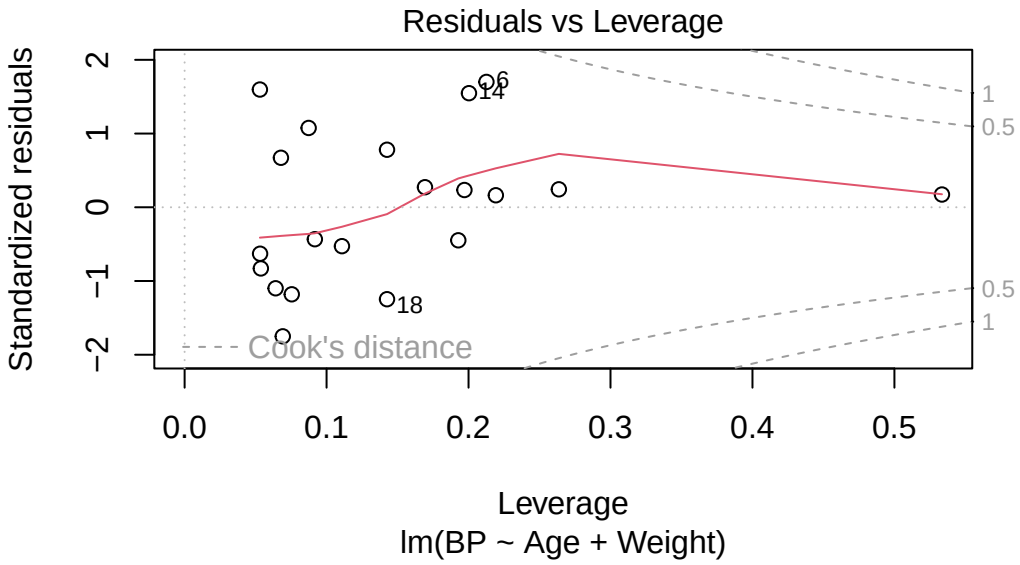
18. Directly use `plot()` function on `model.5`.

```
plot(model.5)
```




Q−Q Residuals
Standardized residuals
−2
−1
0
1
2
19
10
6
Theoretical Quantiles
lm(BP ~ Age + Weight)


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
105
110
115
120
125
19
10
6
Fitted values
lm(BP ~ Age + Weight)



19. Load the `hospital_infct.txt` data and select observations with Stay <= 14.

```
infectionrisk <- read.table("/Users/luyu/Desktop/NOTEsAPH101/hospital_infct.txt", header=T)
infectionrisk <- infectionrisk[infectionrisk$Stay<=14,]
infectionrisk
```

	ID	Stay	Age	InfctRsk	Culture	Xray	Beds	MedSchool	Region	Census	Nurses
1	1	7.13	55.7	4.1	9.0	39.6	279	2	4	207	241
2	2	8.82	58.2	1.6	3.8	51.7	80	2	2	51	52
3	3	8.34	56.9	2.7	8.1	74.0	107	2	3	82	54
4	4	8.95	53.7	5.6	18.9	122.8	147	2	4	53	148
5	5	11.20	56.5	5.7	34.5	88.9	180	2	1	134	151
6	6	9.76	50.9	5.1	21.9	97.0	150	2	2	147	106
7	7	9.68	57.8	4.6	16.7	79.0	186	2	3	151	129
8	8	11.18	45.7	5.4	60.5	85.8	640	1	2	399	360
9	9	8.67	48.2	4.3	24.4	90.8	182	2	3	130	118
10	10	8.84	56.3	6.3	29.6	82.6	85	2	1	59	66
11	11	11.07	53.2	4.9	28.5	122.0	768	1	1	591	656
12	12	8.30	57.2	4.3	6.8	83.8	167	2	3	105	59
13	13	12.78	56.8	7.7	46.0	116.9	322	1	1	252	349
14	14	7.58	56.7	3.7	20.8	88.0	97	2	2	59	79
15	15	9.00	56.3	4.2	14.6	76.4	72	2	3	61	38
16	16	11.08	50.2	5.5	18.6	63.6	387	2	3	326	405
17	17	8.28	48.1	4.5	26.0	101.8	108	2	4	84	73
18	18	11.62	53.9	6.4	25.5	99.2	133	2	1	113	101
19	19	9.06	52.8	4.2	6.9	75.9	134	2	2	103	125

20	20	9.35	53.8	4.1	15.9	80.9	833	2	3	547	519
21	21	7.53	42.0	4.2	23.1	98.9	95	2	4	47	49
22	22	10.24	49.0	4.8	36.3	112.6	195	2	2	163	170
23	23	9.78	52.3	5.0	17.6	95.9	270	1	1	240	198
24	24	9.84	62.2	4.8	12.0	82.3	600	2	3	468	497
25	25	9.20	52.2	4.0	17.5	71.1	298	1	4	244	236
26	26	8.28	49.5	3.9	12.0	113.1	546	1	2	413	436
27	27	9.31	47.2	4.5	30.2	101.3	170	2	1	124	173
28	28	8.19	52.1	3.2	10.8	59.2	176	2	1	156	88
29	29	11.65	54.5	4.4	18.6	96.1	248	2	1	217	189
30	30	9.89	50.5	4.9	17.7	103.6	167	2	2	113	106
31	31	11.03	49.9	5.0	19.7	102.1	318	2	1	270	335
32	32	9.84	53.0	5.2	17.7	72.6	210	2	2	200	239
33	33	11.77	54.1	5.3	17.3	56.0	196	2	1	164	165
34	34	13.59	54.0	6.1	24.2	111.7	312	2	1	258	169
35	35	9.74	54.4	6.3	11.4	76.1	221	2	2	170	172
36	36	10.33	55.8	5.0	21.2	104.3	266	2	1	181	149
37	37	9.97	58.2	2.8	16.5	76.5	90	2	2	69	42
38	38	7.84	49.1	4.6	7.1	87.9	60	2	3	50	45
39	39	10.47	53.2	4.1	5.7	69.1	196	2	2	168	153
40	40	8.16	60.9	1.3	1.9	58.0	73	2	3	49	21
41	41	8.48	51.1	3.7	12.1	92.8	166	2	3	145	118
42	42	10.72	53.8	4.7	23.2	94.1	113	2	3	90	107
43	43	11.20	45.0	3.0	7.0	78.9	130	2	3	95	56
44	44	10.12	51.7	5.6	14.9	79.1	362	1	3	313	264
45	45	8.37	50.7	5.5	15.1	84.8	115	2	2	96	88
46	46	10.16	54.2	4.6	8.4	51.5	831	1	4	581	629
48	48	10.90	57.2	5.5	10.6	71.9	593	2	2	446	211
49	49	7.67	51.7	1.8	2.5	40.4	106	2	3	93	35
50	50	8.88	51.5	4.2	10.1	86.9	305	2	3	238	197
51	51	11.48	57.6	5.6	20.3	82.0	252	2	1	207	251
52	52	9.23	51.6	4.3	11.6	42.6	620	2	2	413	420
53	53	11.41	61.1	7.6	16.6	97.9	535	2	3	330	273
54	54	12.07	43.7	7.8	52.4	105.3	157	2	2	115	76
55	55	8.63	54.0	3.1	8.4	56.2	76	2	1	39	44
56	56	11.15	56.5	3.9	7.7	73.9	281	2	1	217	199
57	57	7.14	59.0	3.7	2.6	75.8	70	2	4	37	35
58	58	7.65	47.1	4.3	16.4	65.7	318	2	4	265	314
59	59	10.73	50.6	3.9	19.3	101.0	445	1	2	374	345
60	60	11.46	56.9	4.5	15.6	97.7	191	2	3	153	132
61	61	10.42	58.0	3.4	8.0	59.0	119	2	1	67	64
62	62	11.18	51.0	5.7	18.8	55.9	595	1	2	546	392
63	63	7.93	64.1	5.4	7.5	98.1	68	2	4	42	49

64	64	9.66	52.1	4.4	9.9	98.3	83	2	2	66	95
65	65	7.78	45.5	5.0	20.9	71.6	489	2	3	391	329
66	66	9.42	50.6	4.3	24.8	62.8	508	2	1	421	528
67	67	10.02	49.5	4.4	8.3	93.0	265	2	2	191	202
68	68	8.58	55.0	3.7	7.4	95.9	304	2	3	248	218
69	69	9.61	52.4	4.5	6.9	87.2	487	2	3	404	220
70	70	8.03	54.2	3.5	24.3	87.3	97	2	1	65	55
71	71	7.39	51.0	4.2	14.6	88.4	72	2	2	38	67
72	72	7.08	52.0	2.0	12.3	56.4	87	2	3	52	57
73	73	9.53	51.5	5.2	15.0	65.7	298	2	3	241	193
74	74	10.05	52.0	4.5	36.7	87.5	184	1	1	144	151
75	75	8.45	38.8	3.4	12.9	85.0	235	2	2	143	124
76	76	6.70	48.6	4.5	13.0	80.8	76	2	4	51	79
77	77	8.90	49.7	2.9	12.7	86.9	52	2	1	37	35
78	78	10.23	53.2	4.9	9.9	77.9	752	1	2	595	446
79	79	8.88	55.8	4.4	14.1	76.8	237	2	2	165	182
80	80	10.30	59.6	5.1	27.8	88.9	175	2	2	113	73
81	81	10.79	44.2	2.9	2.6	56.6	461	1	2	320	196
82	82	7.94	49.5	3.5	6.2	92.3	195	2	2	139	116
83	83	7.63	52.1	5.5	11.6	61.1	197	2	4	109	110
84	84	8.77	54.5	4.7	5.2	47.0	143	2	4	85	87
85	85	8.09	56.9	1.7	7.6	56.9	92	2	3	61	61
86	86	9.05	51.2	4.1	20.5	79.8	195	2	3	127	112
87	87	7.91	52.8	2.9	11.9	79.5	477	2	3	349	188
88	88	10.39	54.6	4.3	14.0	88.3	353	2	2	223	200
89	89	9.36	54.1	4.8	18.3	90.6	165	2	1	127	158
90	90	11.41	50.4	5.8	23.8	73.0	424	1	3	359	335
91	91	8.86	51.3	2.9	9.5	87.5	100	2	3	65	53
92	92	8.93	56.0	2.0	6.2	72.5	95	2	3	59	56
93	93	8.92	53.9	1.3	2.2	79.5	56	2	2	40	14
94	94	8.15	54.9	5.3	12.3	79.8	99	2	4	55	71
95	95	9.77	50.2	5.3	15.7	89.7	154	2	2	123	148
96	96	8.54	56.1	2.5	27.0	82.5	98	2	1	57	75
97	97	8.66	52.8	3.8	6.8	69.5	246	2	3	178	177
98	98	12.01	52.8	4.8	10.8	96.9	298	2	1	237	115
99	99	7.95	51.8	2.3	4.6	54.9	163	2	3	128	93
100	100	10.15	51.9	6.2	16.4	59.2	568	1	3	452	371
101	101	9.76	53.2	2.6	6.9	80.1	64	2	4	47	55
102	102	9.89	45.2	4.3	11.8	108.7	190	2	1	141	112
103	103	7.14	57.6	2.7	13.1	92.6	92	2	4	40	50
104	104	13.95	65.9	6.6	15.6	133.5	356	2	1	308	182
105	105	9.44	52.5	4.5	10.9	58.5	297	2	3	230	263
106	106	10.80	63.9	2.9	1.6	57.4	130	2	3	69	62

107	107	7.14	51.7	1.4	4.1	45.7	115	2	3	90	19
108	108	8.02	55.0	2.1	3.8	46.5	91	2	2	44	32
109	109	11.80	53.8	5.7	9.1	116.9	571	1	2	441	469
110	110	9.50	49.3	5.8	42.0	70.9	98	2	3	68	46
111	111	7.70	56.9	4.4	12.2	67.9	129	2	4	85	136
113	113	9.41	59.5	3.1	20.6	91.7	29	2	3	20	22

	Facilities
1	60.0
2	40.0
3	20.0
4	40.0
5	40.0
6	40.0
7	40.0
8	60.0
9	40.0
10	40.0
11	80.0
12	40.0
13	57.1
14	37.1
15	17.1
16	57.1
17	37.1
18	37.1
19	37.1
20	77.1
21	17.1
22	37.1
23	57.1
24	57.1
25	57.1
26	57.1
27	37.1
28	37.1
29	37.1
30	37.1
31	57.1
32	54.3
33	34.3
34	54.3
35	54.3
36	54.3

37	34.3
38	34.3
39	54.3
40	14.3
41	34.3
42	34.3
43	34.3
44	54.3
45	34.3
46	74.3
48	51.4
49	11.4
50	51.4
51	51.4
52	71.4
53	51.4
54	31.4
55	31.4
56	51.4
57	31.4
58	51.4
59	51.4
60	31.4
61	31.4
62	68.6
63	28.6
64	28.6
65	48.6
66	48.6
67	48.6
68	48.6
69	48.6
70	28.6
71	28.6
72	28.6
73	48.6
74	68.6
75	48.6
76	28.6
77	28.6
78	68.6
79	48.6
80	45.7

```
81        65.7
82        45.7
83        45.7
84        25.7
85        45.7
86        45.7
87        65.7
88        65.7
89        45.7
90        45.7
91        25.7
92        25.7
93         5.7
94        25.7
95        25.7
96        45.7
97        45.7
98        45.7
99        42.9
100       62.9
101       22.9
102       42.9
103       22.9
104       62.9
105       42.9
106       22.9
107       22.9
108       22.9
109       62.9
110       22.9
111       62.9
113       22.9
```

20. Create new dummy/indicator columns (`i1`, `i2`, `i3`, `i4`) for regions using `ifelse()` function. For example, i1 = 1 when Region = 1 and i1 = 0 when Region is not equal to 1; i2 = 1 when Region = 2 and i2 = 0 when Region is not equal to 2; …

```
infectionrisk$i1 <- ifelse(infectionrisk$Region == 1, 1, 0)
infectionrisk$i2 <- ifelse(infectionrisk$Region == 2, 1, 0)
infectionrisk$i3 <- ifelse(infectionrisk$Region == 3, 1, 0)
infectionrisk$i4 <- ifelse(infectionrisk$Region == 4, 1, 0)
```

21. Fit a multiple linear regression model of InfctRsk on Stay + Xray + i2 + i3 + i4.

```r
model.7 <- lm(InfctRsk ~ Stay + Xray + i2 + i3 + i4, data=infectionrisk)
summary(model.7)
```

```
Call:
lm(formula = InfctRsk ~ Stay + Xray + i2 + i3 + i4, data = infectionrisk)

Residuals:
     Min       1Q   Median       3Q      Max
-2.66492 -0.65420  0.04265  0.64034  2.51391

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.134259   0.877347  -2.433  0.01668 *
Stay         0.505394   0.081455   6.205 1.11e-08 ***
Xray         0.017587   0.005649   3.113  0.00238 **
i2           0.171284   0.281475   0.609  0.54416
i3           0.095461   0.288852   0.330  0.74169
i4           1.057835   0.378077   2.798  0.00612 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.036 on 105 degrees of freedom
Multiple R-squared:  0.4198,    Adjusted R-squared:  0.3922
F-statistic: 15.19 on 5 and 105 DF,  p-value: 3.243e-11
```

23. Can we include i1 + i2 + i3 + i4 in this multiple linear regression? Why?

No. In the context of using dummy variables for categorical data in regression analysis, it's essential to designate a reference category. This reference category is represented by a coefficient of zero, while the coefficients for the other categories are interpreted as deviations from this reference point.

```r
model.8 <- lm(InfctRsk ~ Stay + Xray + i1 + i2 + i3 + i4, data=infectionrisk)
summary(model.8)
```

```
Call:
lm(formula = InfctRsk ~ Stay + Xray + i1 + i2 + i3 + i4, data = infectionrisk)

Residuals:
```

```
     Min       1Q   Median       3Q      Max
-2.66492 -0.65420  0.04265  0.64034  2.51391

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.076424   0.721361  -1.492  0.13864
Stay         0.505394   0.081455   6.205 1.11e-08 ***
Xray         0.017587   0.005649   3.113  0.00238 **
i1          -1.057835   0.378077  -2.798  0.00612 **
i2          -0.886551   0.339887  -2.608  0.01042 *
i3          -0.962374   0.323365  -2.976  0.00362 **
i4                NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.036 on 105 degrees of freedom
Multiple R-squared:  0.4198,    Adjusted R-squared:  0.3922
F-statistic: 15.19 on 5 and 105 DF,  p-value: 3.243e-11
```

24. Conduct an F-test (use `anova()` function) to see if at least one of i2, i3, and i4 are useful.

```r
model.9 <- lm(InfctRsk ~ Stay + Xray, data=infectionrisk)
anova(model.7, model.9)
```

```
Analysis of Variance Table

Model 1: InfctRsk ~ Stay + Xray + i2 + i3 + i4
Model 2: InfctRsk ~ Stay + Xray
  Res.Df    RSS Df Sum of Sq      F  Pr(>F)
1    105 112.71
2    108 123.56 -3   -10.849 3.3687 0.02135 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Logistic regression

Simple example

```r
logit_m2 =glm(formula = LUNG_CANCER ~ ANXIETY+PEER_PRESSURE+`CHRONIC DISEASE`+FATIGUE+ALLERG
```

Confusion Matrix

Total population $= \mathrm{P} + \mathrm{N}$	Predicted Positive (PP)	Predicted Negative (PN)	Informedness, bookmaker informedness (BM) $= \mathrm{TPR} + \mathrm{TNR} - 1$	Prevalence threshold (PT) $= \frac{\sqrt{\mathrm{TPR} \times \mathrm{FPR}} - \mathrm{FPR}}{\mathrm{TPR} - \mathrm{FPR}}$
Positive (P) [a]	True positive (TP), hit [b]	False negative (FN), miss, underestimation	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power $= \frac{\mathrm{TP}}{\mathrm{P}} = 1 - \mathrm{FNR}$	False negative rate (FNR), miss rate type ll error [c] $= \frac{\mathrm{FN}}{\mathrm{P}} = 1 - \mathrm{TPR}$
Negative (N) [d]	False positive (FP), false alarm, overestimation	True negative (TN), correct rejection [e]	False positive rate (FPR), probability of false alarm, fall-out type I error [7] $= \frac{\mathrm{FP}}{\mathrm{N}} = 1 - \mathrm{TNR}$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{\mathrm{TN}}{\mathrm{N}} = 1 - \mathrm{FPR}$

example

Lung Cancer Classification (https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer?select=survey+lung+cancer.csv)

The effectiveness of cancer prediction system helps the people to know their cancer risk with low cost and it also helps the people to take the appropriate decision based on their cancer risk status. The data is collected from the website online lung cancer prediction system.

Total no. of attributes: 16 No. of instances: 284

Attribute information:

Gender: M(male), F(female) Age: Age of the patient Smoking: YES=2, NO=1. Yellow fingers: YES=2, NO=1. Anxiety: YES=2, NO=1. Peer_pressure: YES=2, NO=1. Chronic Disease: YES=2, NO=1. Fatigue: YES=2, NO=1. Allergy: YES=2, NO=1. Wheezing: YES=2, NO=1. Alcohol: YES=2, NO=1. Coughing: YES=2, NO=1. Shortness of Breath: YES=2, NO=1. Swallowing Difficulty: YES=2, NO=1. Chest pain: YES=2, NO=1. Lung Cancer: YES, NO.

Goal: It is your job to classify Lung Cancer using other variables

Example Code

```r
#Load the dataset
library(readr)
data = read_csv('/Users/luyu/Desktop/survey_lung_cancer.csv', show_col_types = FALSE)
data$LUNG_CANCER <- ifelse(data$LUNG_CANCER=="YES", 1, 0)
summary(data)
```

```
    GENDER                AGE            SMOKING       YELLOW_FINGERS
 Length:309          Min.   :21.00   Min.   :1.000   Min.   :1.00
 Class :character    1st Qu.:57.00   1st Qu.:1.000   1st Qu.:1.00
 Mode  :character    Median :62.00   Median :2.000   Median :2.00
                     Mean   :62.67   Mean   :1.563   Mean   :1.57
                     3rd Qu.:69.00   3rd Qu.:2.000   3rd Qu.:2.00
                     Max.   :87.00   Max.   :2.000   Max.   :2.00
    ANXIETY       PEER_PRESSURE   CHRONIC DISEASE    FATIGUE
 Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
 1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
 Median :1.000   Median :2.000   Median :2.000   Median :2.000
 Mean   :1.498   Mean   :1.502   Mean   :1.505   Mean   :1.673
 3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000
 Max.   :2.000   Max.   :2.000   Max.   :2.000   Max.   :2.000
    ALLERGY         WHEEZING     ALCOHOL CONSUMING    COUGHING
 Min.   :1.000   Min.   :1.000   Min.   :1.000     Min.   :1.000
 1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000     1st Qu.:1.000
 Median :2.000   Median :2.000   Median :2.000     Median :2.000
 Mean   :1.557   Mean   :1.557   Mean   :1.557     Mean   :1.579
 3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000     3rd Qu.:2.000
 Max.   :2.000   Max.   :2.000   Max.   :2.000     Max.   :2.000
 SHORTNESS OF BREATH SWALLOWING DIFFICULTY   CHEST PAIN     LUNG_CANCER
 Min.   :1.000       Min.   :1.000         Min.   :1.000   Min.   :0.0000
 1st Qu.:1.000       1st Qu.:1.000         1st Qu.:1.000   1st Qu.:1.0000
 Median :2.000       Median :1.000         Median :2.000   Median :1.0000
 Mean   :1.641       Mean   :1.469         Mean   :1.557   Mean   :0.8738
 3rd Qu.:2.000       3rd Qu.:2.000         3rd Qu.:2.000   3rd Qu.:1.0000
 Max.   :2.000       Max.   :2.000         Max.   :2.000   Max.   :1.0000
```

```r
library(ggplot2)
ggplot(data, aes(x = factor(SMOKING), fill = factor(LUNG_CANCER))) +
geom_bar(position = "fill") +
scale_fill_manual(values = c("0" = "lightblue", "1" = "salmon")) +
theme_minimal()
```

```r
### Data SAMPLING ####
library(caret)
```

```
Attaching package: 'caret'
```

```
The following object is masked from 'package:purrr':

    lift

The following object is masked from 'package:mosaic':

    dotPlot

The following object is masked from 'package:survival':

    cluster
```

```r
set.seed(101)
split = createDataPartition(data$LUNG_CANCER, p = 0.80, list = FALSE)
train_data = data[split,]
test_data = data[-split,]
nrow(train_data)
```

```
[1] 248
```

```r
nrow(test_data)
```

```
[1] 61
```

```r
#error metrics -- Confusion Matrix
err_metric=function(CM)
{
  TN =CM[1,1]
  TP =CM[2,2]
  FP =CM[1,2]
  FN =CM[2,1]
  precision =(TP)/(TP+FP)
  recall_score =(TP)/(TP+FN)
  f1_score=2*((precision*recall_score)/(precision+recall_score))
  accuracy_model  =(TP+TN)/(TP+TN+FP+FN)
  False_positive_rate =(FP)/(FP+TN)
  False_negative_rate =(FN)/(FN+TP)
  print(paste("Precision value of the model: ",round(precision,2)))
  print(paste("Accuracy of the model: ",round(accuracy_model,2)))
  print(paste("Recall value of the model: ",round(recall_score,2)))
  print(paste("False Positive rate of the model: ",round(False_positive_rate,2)))
  print(paste("False Negative rate of the model: ",round(False_negative_rate,2)))
  print(paste("F1 score of the model: ",round(f1_score,2)))
}
```

```r
# Logistic regression
logit_m =glm(formula = LUNG_CANCER ~ ., data = train_data, family = 'binomial')
summary(logit_m)
```

```
Call:
glm(formula = LUNG_CANCER ~ ., family = "binomial", data = train_data)

Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)            -32.13449    6.57393  -4.888 1.02e-06 ***
GENDERM                 -0.84716    0.82793  -1.023 0.306203
AGE                      0.01537    0.03490   0.440 0.659756
SMOKING                  1.10495    0.82246   1.343 0.179119
YELLOW_FINGERS           1.10971    0.82005   1.353 0.175982
ANXIETY                  2.08645    1.08610   1.921 0.054725 .
PEER_PRESSURE            1.94159    0.74009   2.623 0.008704 **
`CHRONIC DISEASE`        3.91378    1.12190   3.489 0.000486 ***
FATIGUE                  2.62906    0.89032   2.953 0.003148 **
ALLERGY                  1.42702    0.83764   1.704 0.088454 .
WHEEZING                 1.07933    0.90761   1.189 0.234357
`ALCOHOL CONSUMING`      2.41116    0.98495   2.448 0.014365 *
COUGHING                 3.14783    1.22386   2.572 0.010110 *
`SHORTNESS OF BREATH`   -0.22681    0.84245  -0.269 0.787757
`SWALLOWING DIFFICULTY`  2.24613    1.24347   1.806 0.070865 .
`CHEST PAIN`             0.89356    0.73930   1.209 0.226791
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 198.230  on 247  degrees of freedom
Residual deviance:  77.353  on 232  degrees of freedom
AIC: 109.35

Number of Fisher Scoring iterations: 8
```

```r
# Logistic regression
logit_m2 =glm(formula = LUNG_CANCER ~ ANXIETY+PEER_PRESSURE+`CHRONIC DISEASE`+FATIGUE+ALLERGY+`ALCOHO
summary(logit_m2)
```

```
Call:
glm(formula = LUNG_CANCER ~ ANXIETY + PEER_PRESSURE + `CHRONIC DISEASE` +
    FATIGUE + ALLERGY + `ALCOHOL CONSUMING` + COUGHING + `SWALLOWING DIFFICULTY`,
    family = "binomial", data = train_data)
```

```
Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)             -27.3830     5.5206  -4.960 7.05e-07 ***
ANXIETY                   2.5514     0.8659   2.947 0.003213 **
PEER_PRESSURE             2.1822     0.7245   3.012 0.002593 **
`CHRONIC DISEASE`         3.5120     0.9696   3.622 0.000292 ***
FATIGUE                   2.4939     0.6979   3.573 0.000353 ***
ALLERGY                   2.0104     0.7428   2.707 0.006796 **
`ALCOHOL CONSUMING`       2.5084     0.8653   2.899 0.003744 **
COUGHING                  3.2220     0.9540   3.377 0.000732 ***
`SWALLOWING DIFFICULTY`   2.5273     1.0096   2.503 0.012309 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 198.230  on 247  degrees of freedom
Residual deviance:  83.501  on 239  degrees of freedom
AIC: 101.5

Number of Fisher Scoring iterations: 8
```

```
library(dplyr)
logit_P_prob = predict(logit_m, newdata = select(test_data, -LUNG_CANCER), type = 'response')
logit_P_prob[1:3]
```

```
        1         2         3
0.9912600 0.9997708 0.9980692
```

```
logit_P <- ifelse(logit_P_prob > 0.5, 1, 0) # Probability check
logit_P[1:3]
```

```
1 2 3
1 1 1
```

```
CM = table(test_data$LUNG_CANCER, logit_P)
print(CM)
```

```
   logit_P
     0  1
  0  3  2
  1  1 55
```

```
err_metric(CM)
```

```
[1] "Precision value of the model:  0.96"
[1] "Accuracy of the model:  0.95"
[1] "Recall value of the model:  0.98"
[1] "False Positive rate of the model:  0.4"
[1] "False Negative rate of the model:  0.02"
[1] "F1 score of the model:  0.97"
```

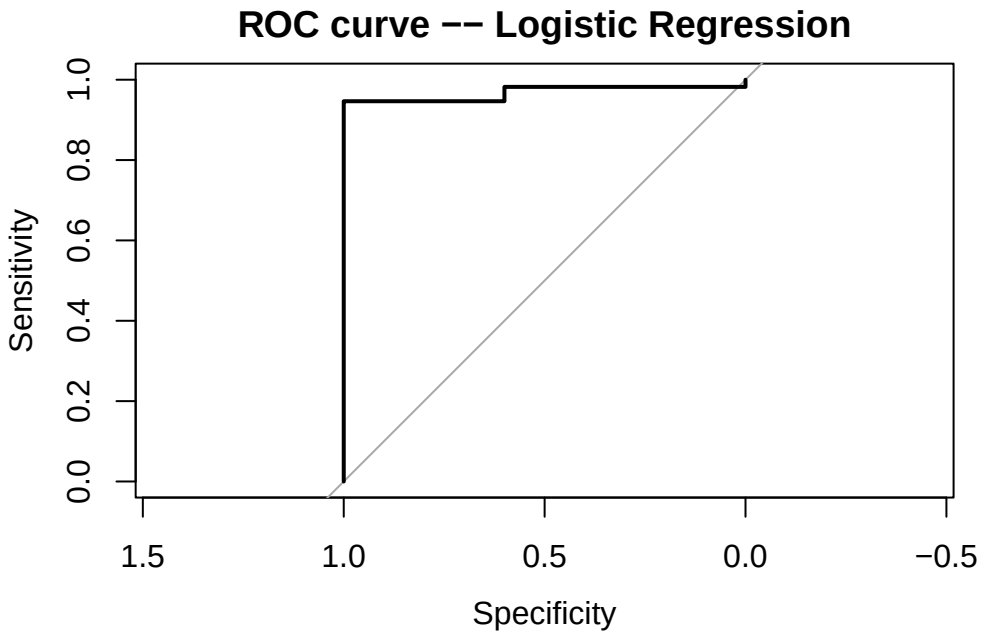
```
#ROC-curve using pROC library
library(pROC)
```

```
Type 'citation("pROC")' for a citation.


Attaching package: 'pROC'

The following objects are masked from 'package:mosaic':

    cov, var

The following objects are masked from 'package:stats':

    cov, smooth, var
```

```
roc_score=roc(test_data$LUNG_CANCER, logit_P_prob) #AUC score
```

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
plot(roc_score, main = "ROC curve -- Logistic Regression")
```


ROC curve −− Logistic Regression
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1.5
1.0
0.5
0.0
−0.5
Specificity