Week 3: Regression modelling part 1

Contents

1 Introduction

Now that we are comfortable with visualising and manipulating data in R, we can now proceed onto modelling data. The key idea behind modelling data is to infer the relationship between an:

- **outcome (or response) variable** y and
- an **explanatory (or predictor) variable** x, which can also be referred to as an **independent variable** or **covariate**.

Modelling can be used for two purposes:

1. **Explanation**: For describing the relationship between an outcome variable y and an explanatory variable *x*, and determining the potential significance of such relationships using quantifiable measures.
2. **Prediction**: for predicting the outcome variable y given information from one or more explanatory variables.

There are many different modelling techniques. However, we will begin with one of the easier to understand and commonly-used approaches, **linear regression**. In particular, we will start by looking at **simple linear regression**, where we only have one explanatory variable.

Note: Additional information and examples can be found in Chapter 5 of An Introduction to Statistical and Data Science via R.

You can download today's session R script below:

> **Note**
>
> I recommend opening a new Quarto file and using it to create your own notes as you work through today's exercises. This will help you practice coding in a structured way, reinforce your learning, and keep a reusable record of your progress.

2 Simple linear regression

For a response variable y and an explanatory variable x, the data can be expressed as:

$$(y_i, x_i), \quad i = 1, \dots, n.$$

That is, we have n observations of y and x. A statistical model is a mathematical statement describing the variability in a random variable y, which includes any relationship with the explanatory variable x. The inclusion of random (unpredictable) components ϵ, makes the model statistical, rather than deterministic. A simple linear regression model involves, as the name suggests, fitting a linear regression line to the data. Hence, a simple linear regression model can be written as follows:

$$y_i = \alpha + \beta x_i + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

where

- y_i is the i^{th} observation of the response variable;
- α is the **intercept** of the regression line;
- β is the **slope** of the regression line;
- x_i is the i^{th} observation of the explanatory variable; and
- ϵ_i is the i^{th} random component.

The random components, ϵ_i, are normally distributed with mean zero and constant variance σ^2, such that we are essentially adding random white noise to the deterministic part of the model ($\alpha + \beta x_i$). Thus, the full probability model for y_i given x_i ($y_i|x_i$) can be written as

$$y_i|x_i \sim N(\alpha + \beta x_i, \sigma^2).$$

Hence, the mean comes from the deterministic part of the model, while the variance comes from the random part. We shall now look into fitting a simple linear regression model to some data.

3 Simple linear regression with one numerical explanatory variable

First, we need to load the following packages into R:

```
library(tidyverse)     # Data wrangling
library(ggplot2)       # Data visualization
library(performance)   # Model assessment
library(skimr)         # Exploratory analysis
library(sjPlot)        # Plot and tables for linear models
```

Student feedback in higher education is extremely important when it comes to the evaluation of teaching techniques, materials, and improvements in teaching methods and technologies. However, there have been studies into potential bias factors when feedback is provided, such as the physical appearance of the teacher; see Economics of Education Review for details. Here, we shall look at a study from student evaluations of $n = 463$ professors from The University of Texas at Austin. In particular, we will examine the evaluation scores of the instructors based purely on one numerical variable: their *beauty score*. Therefore, our simple linear regression model will consist of:

- the numerical outcome variable *teaching score* (y); and
- the numerical explanatory variable *beauty score* (x).

3.1 Exploratory data analysis

Before you ever do any statistical modelling of data, you should always perform an **exploratory data analysis** of the data. Performing an exploratory data analysis can give us an idea of the distribution of the data, and whether it contains any strange values, such as **outliers** or **missing values**. However, more importantly, it is used to inform which statistical model we should fit to the data. An exploratory data analysis may involve:

1. Looking at the raw values of the data, either by looking at the spreadsheet directly, or using R.
2. By computing various summary statistics, such as the *five-number summary*, means, and standard deviations.
3. Plotting the data using various data visualisation techniques.

Let's examine the data `evals`. First we read the data (stored in `.csv` format) and then we can look at the raw values from `evals` using the RStudio pop-up spreadsheet viewer using:

```
evals <- read.csv("evals.csv")
View(evals)
```

At the moment we are only really interested in the instructors teaching (`score`) and beauty (`bty_avg`) scores, and so we can look at a subset of the data as follows:

```
evals.scores <- evals %>%
  dplyr::select(score, bty_avg)
```

The outcome variable `score` is a numerical average of the average teaching score based on students' evaluations between 1 and 5. The explanatory variable `bty_avg` is the numerical variable of the average *beauty* score from a panel of six students' scores between 1 and 10. As both variables are numerical, we can compute summary statistics for them using the `skim` function from the `skimr` package as follows:

```
evals.scores %>%
  skim()
```

Table 1: Data summary

Name	Piped data
Number of rows	463
Number of columns	2

Column type frequency:	
numeric	2
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
score	0	1	4.17	0.54	2.30	3.80	4.30	4.6	5.00	□□□□□
bty_avg	0	1	4.42	1.53	1.67	3.17	4.33	5.5	8.17	□□□□□

This provides us with the following information:

- `missing`: the number of missing values.
- `complete`: the number of non-missing values.
- `n`: the total number of observations.
- `mean`: the mean or average.
- `sd`: the standard deviation.
- `p0`: the 0^{th} percentile: the value at which 0% of values are smaller than it (i.e. the *minimum*).
- `p25`: the 25^{th} percentile: the value at which 25% of values are smaller than it (i.e. the *1st quartile*).
- `p50`: the 50^{th} percentile: the value at which 50% of values are smaller than it (i.e. the *median*).
- `p75`: the 75^{th} percentile: the value at which 75% of values are smaller than it (i.e. the *3rd quartile*).
- `p100`: the 100^{th} percentile: the value at which 100% of values are smaller than it (i.e. the *maximum*).
- `hist`: provides a snapshot of a histogram of the variable.

These summary statistics give us an idea of how both variables are distributed. For example, the mean teaching score (`score`) is 4.17 out 5, while the mean beauty score (`bty_avg`) is 4.42 out of 10. Also, the middle 50% of the data for `score` lies between 3.8 and 4.6, while the middle 50% of `bty_avg` lies between 3.17 and 5.5.

3.2 Correlation

The above summary statistics provide information about each variable separately. However, we are interested in a potential relationship between the two variables and as such it would be of interest to evaluate some statistic that considers both variables simultaneously. One such statistic is the **correlation**, which ranges between -1 and 1 and describes the strength of the linear relationship between two numerical variables, such that

- -1 indicates a perfect *negative relationship*. That is, as the values of one variable increase, the values of the other decrease.
- 0 indicates no relationship. The values of both variables increase/decrease independently of one another.
- 1 indicates a perfect *positive relationship*. That is, the values of both variables increase simultaneously.

The correlation coefficient $\rho(\cdot)$ between two variables x and y can be computed as:

$$\rho(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Here, $\bar{x}$ and $\bar{y}$ denotes the mean of x_i and y_i respectively across $i = 1, \ldots, n$ observations.The plot below displays scatterplots for hypothetical numerical variables x and y simulated to have different levels of correlation.

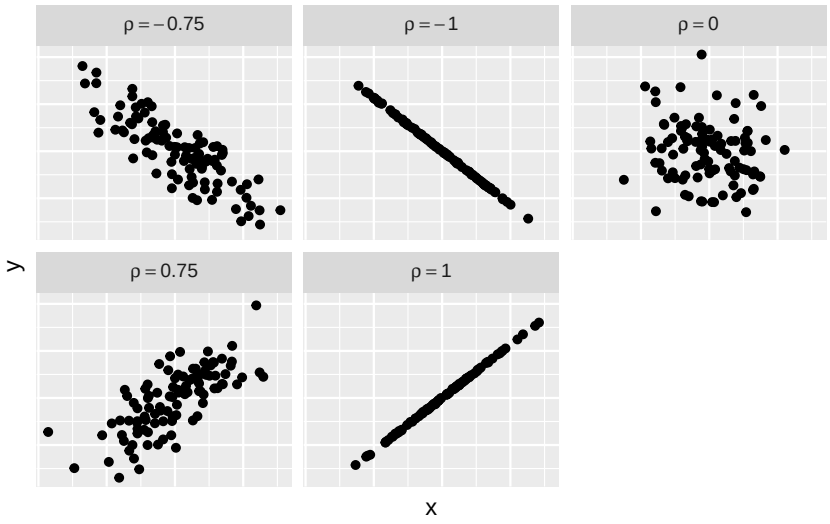


Figure 1: Differing levels of correlation between variables.

The correlation coefficient can be computed in R using the `cor` function as follows:

```
cor(evals.scores$score,evals.scores$bty_avg)
```

```
[1] 0.1871424
```

Here, we are given a correlation coefficient of 0.187 for the relationship between teaching (`score`) and beauty (`bty_avg`) scores. This suggests a rather *weakly positive* linear relationship between the two variables. There is some subjective interpretation surrounding correlation coefficients not very close to -1, 0, 1. The table below provides a rough guide as to the verbal interpretation of a correlation coefficient.

Correlation coefficient	Verbal interpretation
0.90 to 1.00 (-0.90 to -1.00)	Very strong positive (negative) correlation
0.70 to 0.90 (-0.70 to -0.90)	Strong positive (negative) correlation
0.50 to 0.70 (-0.50 to -0.70)	Moderate positive (negative) correlation
0.30 to 0.50 (-0.30 to -0.50)	Weak positive (negative) correlation
0.00 to 0.30 (0.00 to -0.30)	Very weak positive (negative) correlation

The next step in our exploratory data analysis is to visualise the data using appropriate plotting techniques. Here, a scatterplot is appropriate since both `score` and `bty_avg` are numerical variables:

```
ggplot(evals.scores, aes(x = bty_avg, y = score)) +
  geom_point() +
  labs(x = "Beauty Score", y = "Teaching Score", title = "Relationship of teaching and beaut
```

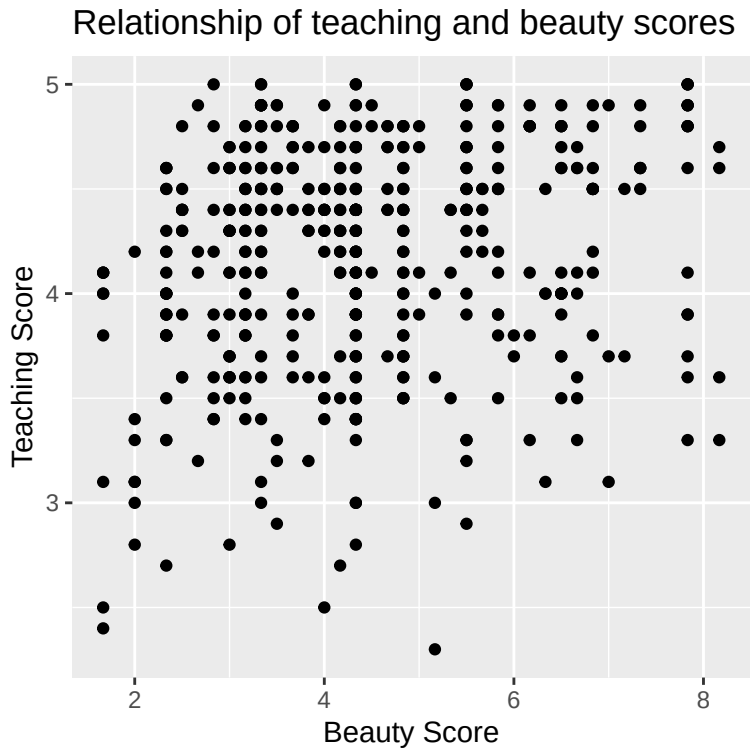


Figure 2: Relationship between teaching and beauty scores.

> **Note**
>
> The outcome variable should always be plotted on the y-axis.

What can we observe from the scatterplot? Well, here it can be hard to see the weakly positive linear relationship suggested by the correlation coefficient (0.187), which is why our correlation coefficient is considered *very weak* in the verbal interpretation.

Additionally, as our numerical variables are averages of integers (or whole numbers), a lot of the values will be plotted on top of one another. This is often referred to as **over-plotting**, and can be alleviated by slightly nudging (**jittering**) the points in a random direction (This can be done by adding a `geom_jitter()` layer in our ggplot object). For example, let's look at the three points in the top-right of the scatterplot that have a beauty score slightly less than 8. Are there really only three values plotted there, or are there more that we cannot see due to over-plotting? Let's find out by adding some jitter to the plot:

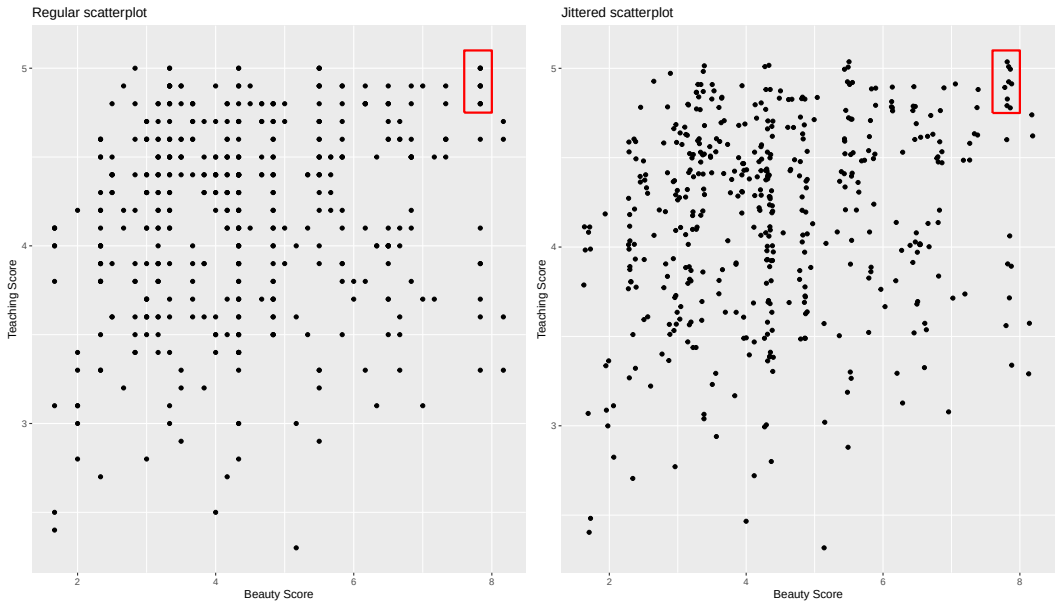


Figure 3: Comparing regular and jittered scatterplots.

From the jittered scatterplot we can see that:

1. There are actually more than just three points plotted in the top-right; and
2. There are more instructors with a beauty score between 3 and 4.5 than originally appears due to over-plotting.

> **Note**
>
> Jittering does not actually change the values within a data set, it is merely a tool for visualisation purposes. Hence, we shall continue on with plotting the original data.

3.3 Formal analysis

After completing an exploratory data analysis the next step is to perform a **formal analysis** on the data. This involves constructing an appropriate statistical model from the information gathered during the exploratory data analysis step. Here, we shall be fitting a simple linear regression model to the data on teaching and beauty scores, where our objective is to acquire the best fitting regression line. This is done by finding estimates of the intercept (α) and slope (β) which give us the best-fitting line to the data. This can be done in R using the `lm` function:

```
model <- lm(score ~ bty_avg, data = evals.scores)
tab_model(model,show.ci = F)
```

	score	
Predictors	Estimates	p
(Intercept)	3.88	**<0.001**
bty avg	0.07	**<0.001**
Observations	463	
R^2 / R^2 adjusted	0.035 / 0.033	

We have summarised the fitted model using the `tab_model` function from the `sjPlot` library (we have omitted confidence intervals for now). This tells us that our best-fitting line to the data is:

$$\widehat{\text{score}} = \hat{\alpha} + \hat{\beta} x_i = 3.88 + 0.07 \cdot \text{bty\_avg},$$

where

- $\hat{\alpha} = 3.88$ is the intercept coefficient and means that, for any instructor with a `bty_avg = 0`, their average teaching `score` would be 3.88. Note that `bty_avg = 0` is not actually possible as `bty_avg` is an average of beauty scores ranging between 1 and 10.
- $\hat{\beta} = 0.07$ is the slope coefficient associated with the exploratory variable `bty_avg`, and summarises the relationship between `score` and `bty_avg`. That is, as `bty_avg` increases, so does `score`, such that
  - For every 1 unit increase in `bty_avg`, there is an associated increase of, on average, 0.06664 units of `score`.

> **Note**
>
> The `broom` library is another alternative to `sjPlot` that allow us to visualize linear model output in a tidy way. For example, we can ask for a model summary using the `tidy` function that will return the output in a `data.frame` format:

```
library(broom)
tidy(model)
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)   3.88      0.0761     51.0  1.56e-191
2 bty_avg       0.0666    0.0163      4.09 5.08e-  5
```

Finally, we can superimpose our best-fitting line onto our scatterplot to see how it fits through the points using the `geom_smooth` function as follows:

```
ggplot(evals.scores, aes(x = bty_avg, y = score)) +
  geom_point() +
  labs(x = "Beauty Score", y = "Teaching Score",
       title = "Relationship of teaching and beauty scores") +
  geom_smooth(method = "lm", se = FALSE)
```

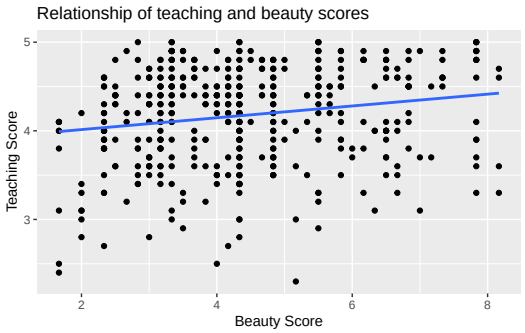


Figure 4: Relationship between teaching and beauty scores with regression line superimposed.

Now that we have fitted our simple linear regression model to the data, how do we use it to obtain information on individual data points? This can be done by looking at the **fitted values**. For example, let's say we are interested in looking at the 21st instructor who has the following teaching and beauty scores:

score	bty_avg
4.9	7.33

What would the `score` be on our best-fitting line for this instructor with a `bty_avg` of 7.33? We simply plug the instructor's `bty_avg` into our regression model:

$$\widehat{\text{score}} = 3.88034 + 0.06664 \cdot \text{bty\_avg} = 3.88034 + 0.06664 \cdot 7.33 = 4.369,$$

The regression model gives our instructor a `score` of 4.369. However, we know the `score` of the instructor is 4.9 meaning that our model was out by 0.531. This is known as the **residual** (ϵ) and can be thought of as the error or *lack of fit* of the regression line. In this case, the residual is given by:

$$\hat{\epsilon} = y - \hat{y} = 4.9 - 4.369 = 0.531.$$

This is essentially the distance between the fitted regression line and the observed (true) value. This can be seen on the following scatterplot:

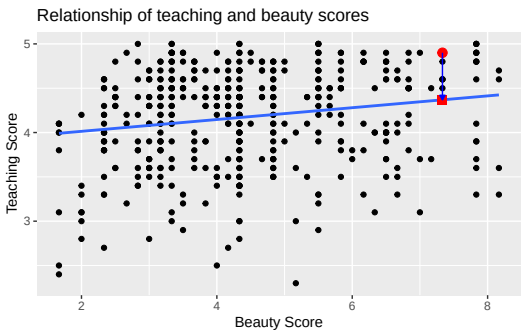


Figure 5: Example of observed value, fitted value, and residual.

where

- the red circle is the observed (true) `score` ($y = 4.9$) of the instructor;

- the red square is the fitted value ($\widehat{y} = 4.369$) from the regression line; and
- the blue arrow is the distance between the observed and fitted values, that is, the residual.

Residuals and fitted values can also be obtained directly from the fitted model by typing `model$fitted.values` and `model$residuals` respectively. We can appended these values to our original data using the `mutate` function as follows:

```
model_output <- evals.scores %>%
  mutate(score_hat  = model$fitted.values,
         residuals  = model$residuals)
model_output %>% slice(1:6)
```

```
  score bty_avg score_hat  residuals
1   4.7       5  4.213523  0.4864769
2   4.1       5  4.213523 -0.1135231
3   3.9       5  4.213523 -0.3135231
4   4.8       5  4.213523  0.5864769
5   4.6       3  4.080249  0.5197509
6   4.3       3  4.080249  0.2197509
```

3.4 Assessing model fit

When we fit a simple linear regression model there are five main assumptions that we need to hold true in order for the model to be an appropriate fit to the data. These assumptions are:

1. The deterministic part of the model captures all the non-random structure in the data, i.e. the residuals have mean zero.
2. The scale of the variability of the residuals is constant at all values of the explanatory variables (*homoscedasticity*).
3. The residuals are normally distributed.
4. The residuals are independent.
5. The values of the explanatory variables are recorded without error.

One way we can check our first assumption is to plot the residuals (`residuals`) against the explanatory variable (`bty_avg`). From this we should be able to check that the explanatory variable has a linear relationship with the outcome variable (`score`). We can plot the residuals against our explanatory variable using:

```
ggplot(model_output, aes(x = bty_avg, y = residuals)) +
  geom_point() +
  labs(x = "Beauty Score", y = "Residual") +
  geom_hline(yintercept = 0, col = "blue", linewidth = 1)
```

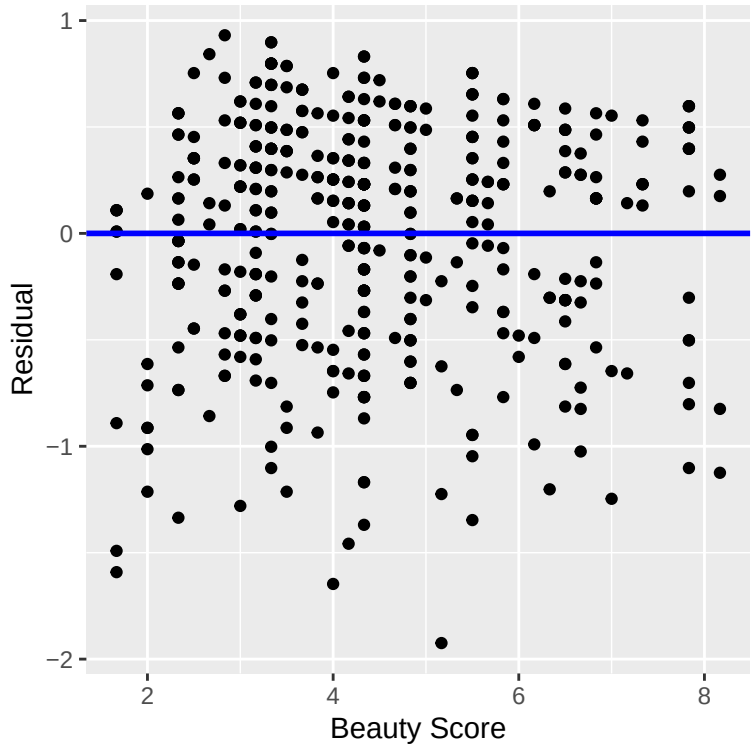


Figure 6: Residuals against beauty score.

Ideally, for the first assumption to hold we should observe the following:

- There should be no systematic pattern, i.e. the residuals should appear randomly scattered.
- The residuals should have mean zero. That is, they should be evenly scattered above and below the zero line. This is because the regression model will overestimate some of the fitted values, but it will also underestimate some, and hence, on average, they should even out to have mean zero.

Another way in which we can examine our first two assumptions is by plotting the residuals against the fitted values.

The `performance` package allows us to do so via the `check_model()` function as follows:

```
check_model(model,check = c("homogeneity","linearity"))
```

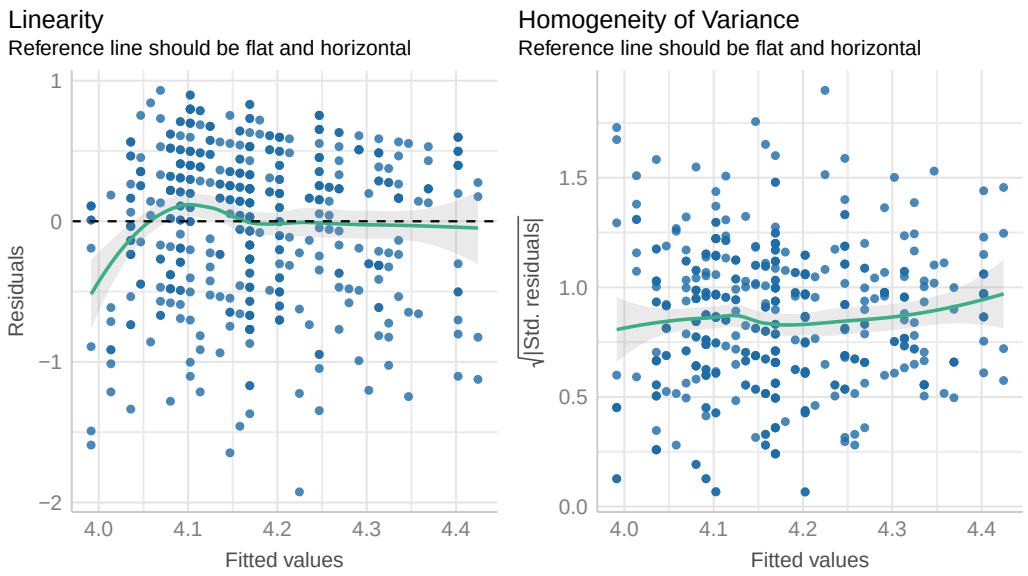


Figure 7: Residuals against fitted values.

Here we have asked to `check_model` function to check the *homoscedasticity*and linearity assumptions by setting `check = c("homogeneity","linearity")`. From the plot of the residuals against the fitted values we want to examine whether:

- The residuals have mean zero (left plot).
- If the residuals have constant variance across all levels of the fitted values. That is, the range (or spread) of the residuals should be similar across all levels of the fitted values and display no obvious changes in variability (right plot).

These two assumptions seems to hold for our model. To assess our third assumption that the residuals are normally distributed we can simply plot a histogram of the residuals and compare the theoretical quantiles of the normal distribution against the sample quantiles.

```
check_model(model,check = c("normality","qq"))
```

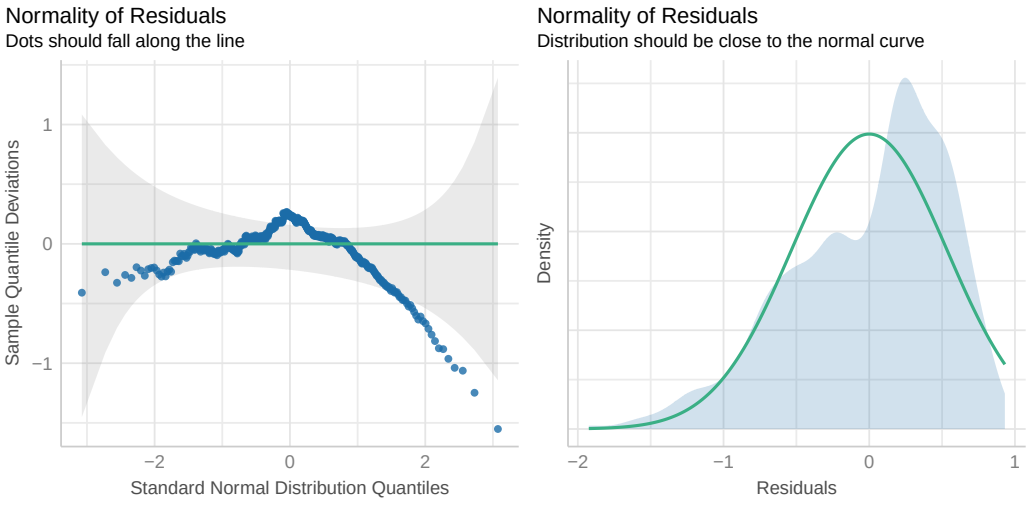
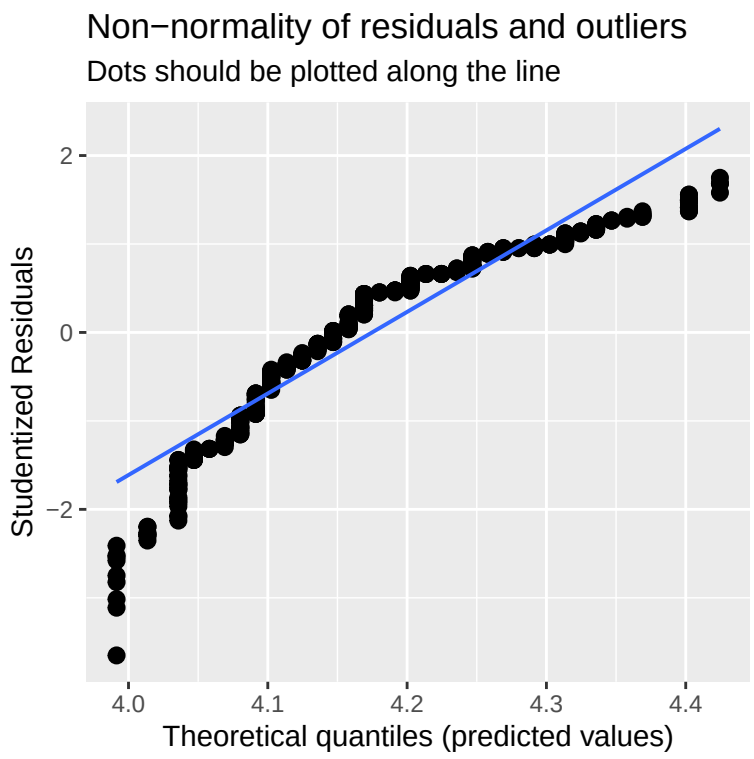


Figure 8: QQ-plot and Histogram of residuals.

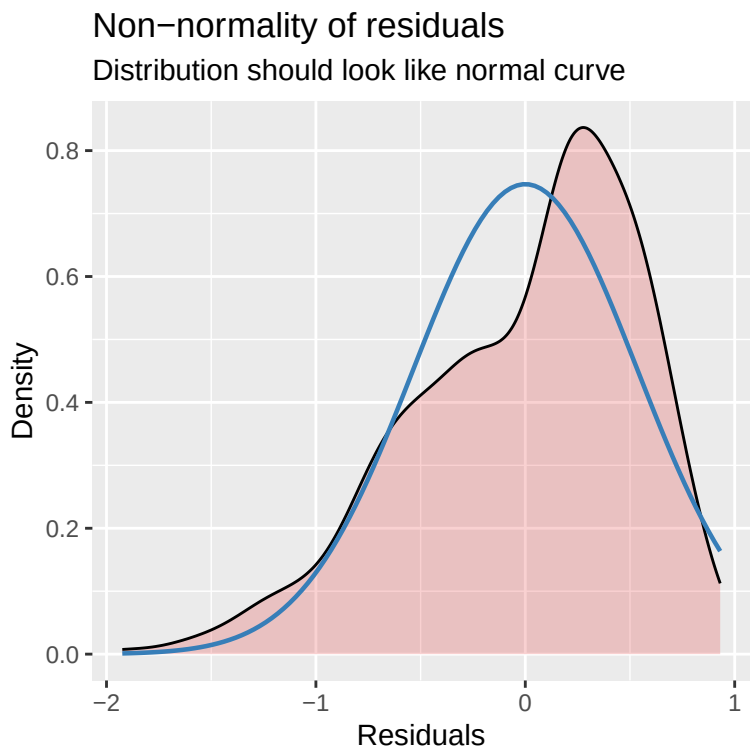
Ideally, for the assumption of normally distributed residuals, the histogram should be bell-shaped and centred at zero, i.e. the residuals have mean zero. However, in practice this will almost never be the case, and as such, like the plots of the residuals, there is some subjectivity in whether you believe the assumptions hold. For instance, here we can see that the histogram is slightly skewed to the left in that the distribution has a longer tail to the left. We can also assess these assumption by comparing the theoretical quantiles of the normal distribution against the sample quantiles. If the normality assumption holds, then we expect that most of the dots should fall along the line. However this seems not to be the case and while the histogram appears to be relatively symmetrical and bell-shaped, the assumption of normally distributed random errors seems dubious.

Note that we can get similar diagnostic plots using the `plot_model` function from the `sjPlot` package by adding the `type='diag'` option. This will produce a list of plots that can be accessed as follows:

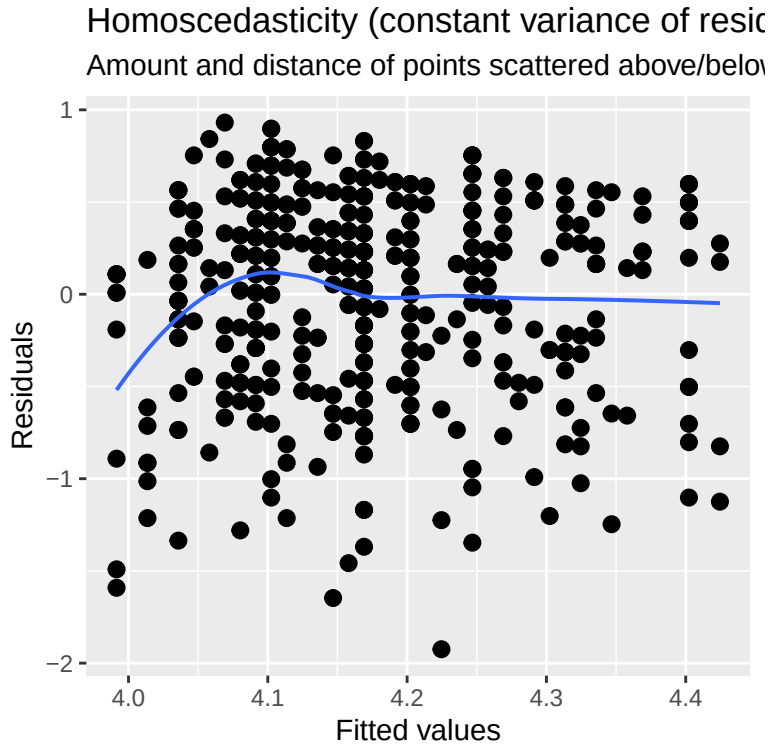
```
diag_plots = plot_model(model,type='diag')

# QQ plot
diag_plots[[1]]
```




```
# Histrogram/density plot
diag_plots[[2]]
```




```
# Residuals vs fitted values
diag_plots[[3]]
```



Finally, assumptions 4. and 5. are often justified on the basis of the experimental context and are not formally examined.

4 Simple linear regression with one categorical explanatory variable

Here, we will fit a simple linear regression model were the explanatory variable is categorical. A **categorical variable** is a variable of a fixed number of possible values, assigning units to particular groups (or categories) based on qualitative properties.

We shall examine the `gapminder` data set (available from the `gapminder` library) which you can download below:

This is a data set on life expectancy across various countries around the world. We will explore life expectancy and its potential differences:

- Between continents: Does life expectancy vary, on average, between the five continents of the world?; and
- Within continents: Does life expectancy vary, on average, within the five continents of the world?

Thus, we will be looking at:

- life expectancy as our numerical outcome variable y; and
- the continent a country is within as our categorical variable x.

4.1 Exploratory data analysis

Let's examine a subset of the `gapminder` data set relating to the year 2007. That is, we use the `filter` function to choose only the observations pertaining to 2007, and then `select` the variables we are interested in (remember to solve any conflicting libraries using the `conflicts` package or refer to which package each function is obtained from). Lastly we declare the `country` and `continent` continent variables as factors using the `mutate` function:

```
gapminder <- read.csv("gapminder.csv")

gapminder2007 <- gapminder %>%
  dplyr::filter(year == 2007) %>%
  dplyr::select(country, continent, lifeExp) %>%
  mutate(country   = as.factor(country),
         continent = as.factor(continent))
```

The new data set can be examined using either the `View` or `glimpse` functions, i.e.

```
glimpse(gapminder2007)
```

```
Rows: 142
Columns: 3
$ country   <fct> "Afghanistan", "Albania", "Algeria", "Angola", "Argentina", ~
$ continent <fct> Asia, Europe, Africa, Africa, Americas, Oceania, Europe, Asi~
$ lifeExp   <dbl> 43.828, 76.423, 72.301, 42.731, 75.320, 81.235, 79.829, 75.6~
```

Here, we can see that both `country` and `continent` are factors (`fct`), which is a way in how R stores categorical variables. Similarly to our previous exploratory data analysis, we can obtain summary statistics using the `skim` function. First, let's take a look at the life expectancy (`lifeExp`) and `continent` variables:

```
gapminder2007 %>%
  select(continent, lifeExp) %>%
  skim()
```

Table 6: Data summary

Name	Piped data
Number of rows	142
Number of columns	2
Column type frequency:	
factor	1
numeric	1
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
continent	0	1	FALSE	5	Afr: 52, Asi: 33, Eur: 30, Ame: 25

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
lifeExp	0	1	67.01	12.07	39.61	57.16	71.94	76.41	82.6	□□□□□

The summary output for the numerical outcome variable `lifeExp` is the same as we have seen previously. However, for the categorical variable `continent` we obtain:

- `n_unique`: the number of levels (or categories) of the variable, i.e. the number of continents.
- `top_counts`: the top counts from the top categories.
- `ordered`: whether the variable is *ordinal* or not. That is, whether or not the ordering of the categories matter.

> **!** Important
>
> It is important to always check your data after performing any kind of manipulation. This helps ensure that you have not accidentally overwritten a variable or introduced errors

We can summarise any differences in life expectancy by continent by taking a look at the median and mean life expectancies of each continent using the `summarize` functions as follows:

```
lifeExp.continent <- gapminder2007 %>%
  summarize(median = median(lifeExp), mean = mean(lifeExp),.by = continent)
lifeExp.continent
```

```
  continent  median     mean
1      Asia 72.3960 70.72848
2    Europe 78.6085 77.64860
3    Africa 52.9265 54.80604
4  Americas 72.8990 73.60812
5   Oceania 80.7195 80.71950
```

Boxplots are often used when examining the distribution of a numerical outcome variable across different levels of a categorical variable:

```
ggplot(gapminder2007, aes(x = continent, y = lifeExp)) +
  geom_boxplot() +
  labs(x = "Continent", y = "Life expectancy (years)",
       title = "Life expectancy by continent")
```

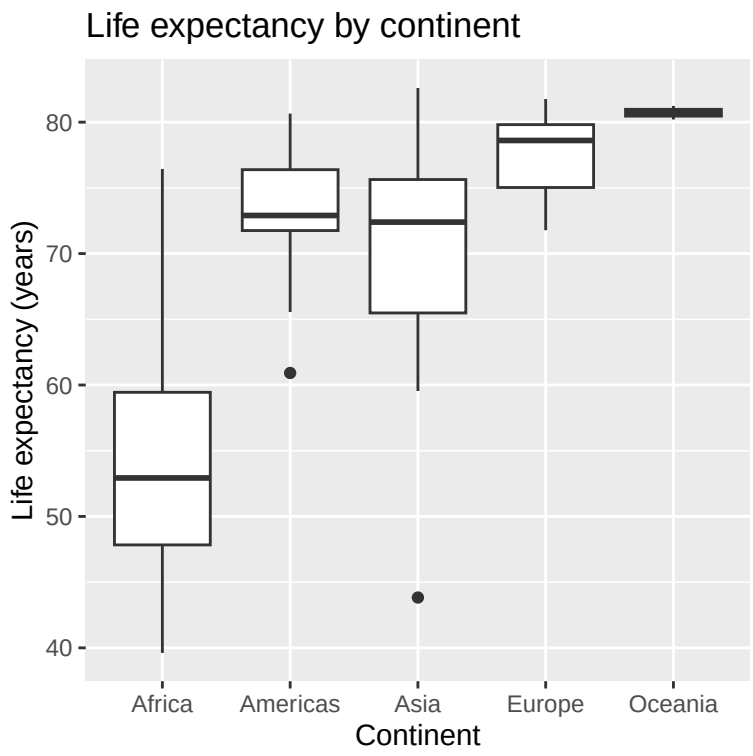


Figure 9: Life expectancy by continent in 2007.

Here, we can see that the middle 50% of the life expectancy distribution of Africa is much smaller than, and does not overlap with, the middle 50% of the remaining four continents, while the country with the highest life expectancy in Africa is less than all countries in Oceania. Speaking of Oceania, there is almost no variability (or spread) in life expectancy in this continent, however that may well be because it consists of only two countries (Australia and New Zealand). There is more variability in life expectancy in the continents of Africa and Asia.

4.2 Formal analysis

When examining the relationship between a numerical outcome variable y and a categorical explanatory variable x, we are not just looking to find the best-fitting line to the data as before, but are examining relative differences to a baseline category. For example, the table below displays the mean life expectancy of each continent, as well as the differences between the means of each continent and Africa. Now, in comparison with Africa we can see that the mean life expectancy of the other continents is around 18-26 years greater than that of Africa.

continent	mean	mean vs Africa
Africa	54.81	0.00
Americas	73.61	18.80
Asia	70.73	15.92
Europe	77.65	22.84
Oceania	80.72	25.91

Now let us fit our regression model to the data, where `lifeExp` is our outcome variable y and `continent` is our categorical explanatory variable x:

```
lifeExp.model <- lm(lifeExp ~ continent, data = gapminder2007)
tab_model(lifeExp.model,show.ci = F)
```

	life Exp	
Predictors	Estimates	p
(Intercept)	54.81	**<0.001**
continent [Americas]	18.80	**<0.001**
continent [Asia]	15.92	**<0.001**
continent [Europe]	22.84	**<0.001**
continent [Oceania]	25.91	**<0.001**
Observations	142	
R^2 / R^2 adjusted	0.635 / 0.625	

We obtain five estimates: the `intercept` term and four others relating to the continents (`continent [Americas]`, `continent [Asia]`, `continent [Europe]` and `continent [Oceania]`), such that our regression equation is given as:

$$\widehat{\text{life exp}} = \hat{\alpha} + \hat{\beta}_{\text{Amer}} \cdot \mathbb{I}_{\text{Amer}}(x) + \hat{\beta}_{\text{Asia}} \cdot \mathbb{I}_{\text{Asia}}(x) + \hat{\beta}_{\text{Euro}} \cdot \mathbb{I}_{\text{Euro}}(x) + \hat{\beta}_{\text{Ocean}} \cdot \mathbb{I}_{\text{Ocean}}(x),$$

where

- the intercept $\hat{\alpha}$ is the mean life expectancy for our baseline category Africa;
- $\hat{\beta}_{\text{continent}}$ is the difference in the mean life expectancy of a given continent relative to the baseline category Africa; and
- $\mathbb{I}_{\text{continent}}(x)$ is an indicator function such that

$$\mathbb{I}_{\text{continent}}(x) = \begin{cases} 1 & \text{if country } x \text{ is in the continent,} \\ 0 & \text{Otherwise.} \end{cases}$$

Essentially, the estimates for each continent are known as *offsets* relative to the baseline category (Africa in this case). For example, the mean life expectancy for Africa is simply equal to the intercept term $\hat{\alpha} = 54.8$. However, the mean life expectancy for Asia is:

$$\hat{\alpha} + \hat{\beta}_{\text{Asia}} \cdot \mathbb{I}_{\text{Asia}}(x) = 54.8 + 15.9 \cdot 1 = 70.7$$

If we just look at a $\hat{\beta}_{\text{continent}}$ on their own, then we would interpret these coefficients in relative terms with respect to the baseline category. E.g., looking at $\hat{\beta}_{\text{Asia}} = 15.9$, we would say that the life expectancy in Asia is on average 15.9 years greater than in Africa.

4.3 Assessing model fit

What do the fitted values $\hat{y}$ and the residuals $y - \hat{y}$ correspond to when we are dealing with a categorical explanatory variable? Let's explore the `gapminder2007` data set in order to understand how they work.

```
gapminder2007 %>% slice(1:8)
```

```
      country continent lifeExp
1 Afghanistan      Asia  43.828
2     Albania    Europe  76.423
3     Algeria    Africa  72.301
4      Angola    Africa  42.731
5   Argentina  Americas  75.320
6   Australia   Oceania  81.235
7     Austria    Europe  79.829
8     Bahrain      Asia  75.635
```

Here, we see the life expectancy of each country and the continent they are from. For example, let's remember the life expectancies of Afghanistan (43.8) and Bahrain (75.6). Now, we can obtain the fitted values and residuals in the same way we did previously:

```
lifeExp.model_output <- gapminder2007 %>%
  mutate(lifeExp_hat  = lifeExp.model$fitted.values,
         residual = lifeExp.model$residuals)

lifeExp.model_output %>% slice(1:10)
```

The first row of the regression table corresponds to the observed life expectancy (`lifeExp`), fitted value (`lifeExp_hat`) and the residual error (`residual`) for Afghanistan. Here, we see that the fitted value (`lifeExp_hat = 70.7`) is much greater than the life expectancy of Afghanistan (`lifeExp = 43.8`) with a `residual = -26.9`. Now, for Bahrain (`ID = 8`) we also have the same fitted value (`lifeExp_hat = 70.7`). This is because the fitted values for each `country` correspond to the mean life expectancy for that `continent` (you can use the search box in Table 11 above and type `Asia` to show only the observation in the Asia continent). Hence, all countries in Africa have the fitted value `lifeExp_hat = 70.7`, while all countries in Europe have the fitted value `lifeExp_hat = 77.6`. The `residual` error in this case is then how much a country deviates from the mean life expectancy of its respective continent.

Table 11: Observed life expectancy values, fitted values and residuals of a linear model fitted to the gapminder2007 data.

Show 10 entries Search:

	country	continent	lifeExp	lifeExp_hat	residual
1	Afghanistan	Asia	43.828	70.728	-26.900
2	Albania	Europe	76.423	77.649	-1.226
3	Algeria	Africa	72.301	54.806	17.495
4	Angola	Africa	42.731	54.806	-12.075
5	Argentina	Americas	75.32	73.608	1.712
6	Australia	Oceania	81.235	80.719	0.516
7	Austria	Europe	79.829	77.649	2.180
8	Bahrain	Asia	75.635	70.728	4.907
9	Bangladesh	Asia	64.062	70.728	-6.666
10	Belgium	Europe	79.441	77.649	1.792

Showing 1 to 10 of 142 entries Previous 1 2 3 4 5 … 15 Next

For assessing the assumptions surrounding the residuals for a categorical explanatory variable, we can plot the residuals for each continent:

```
ggplot(lifeExp.model_output, aes(x = continent, y = residual)) +
  geom_jitter(width = 0.1) +
  labs(x = "Continent", y = "Residual") +
  geom_hline(yintercept = 0, col = "blue")
```

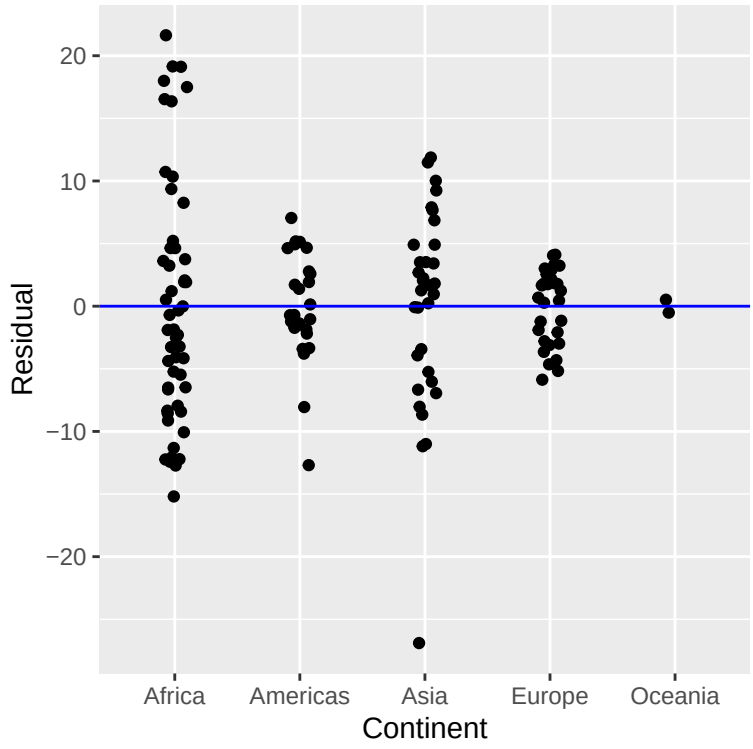


Figure 10: Residuals over continent.

> **Note**
>
> We have jittered the points for each continent in order to see the residuals for each country more clearly.

Here, we see that there is an even spread of the residuals above and below the zero line for each continent, and hence our assumption that the residuals have mean zero appears valid. There is an outlier observed for Asia with a large negative residual (relating to Afghanistan). We could also check this by plotting the residuals agianst the fitted values as follows:

```
check_model(lifeExp.model,check=c("homogeneity","linearity"))
```

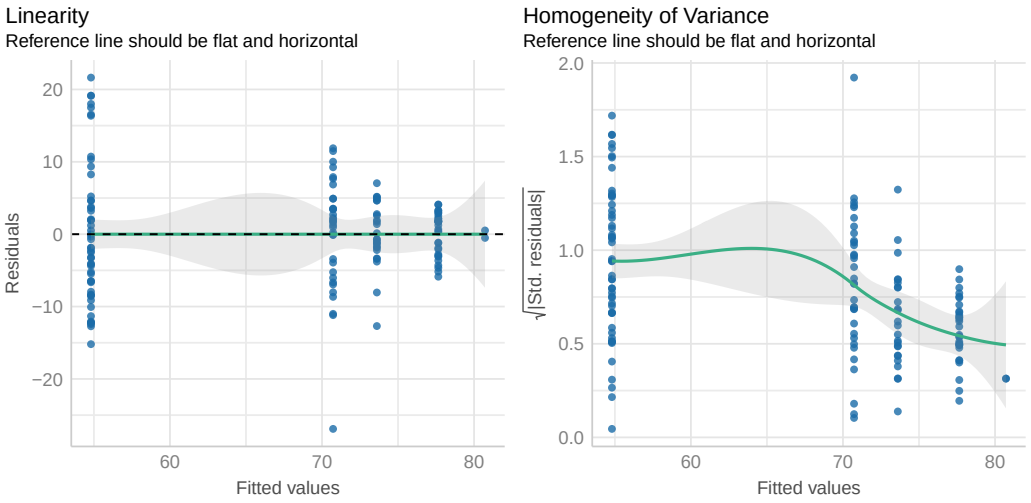


Figure 11: Residuals vs fitted values.

Again, our assumption that the residuals have mean zero seems to hold.

Question

What about the homoscedasticity assumption, is it valid?

- (A) yes, residuals are not heteroscedastic
- (B) no, there is an unbalanced number of residuals per country
- (C) yes, residuals show an even spread across countries
- (D) no, the spread of the residuals is not even across countries

To check that the residual errors are normally distributed, we plot a histogram and a QQplot of them:

```
check_model(lifeExp.model,check=c("qq","normality"))
```

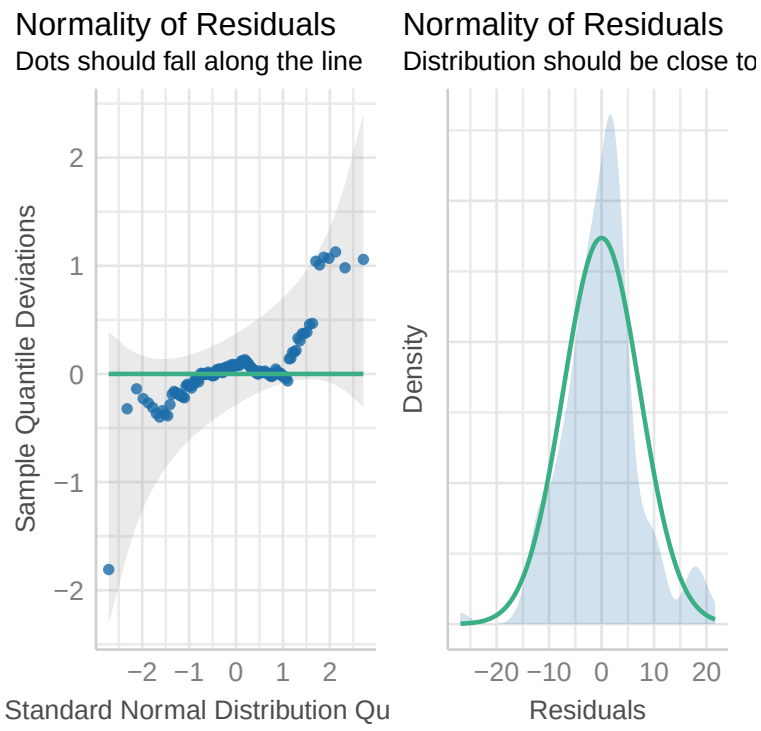


Figure 12: QQ-plot and Histogram of residuals.

While some of the dots deviate from the flat line in the QQ plot, these deviation occur at the tail where we have less data. Furthermore, the histogram of the residuals are close to a bell-shaped and thus, the assumption of normality seems valid.

5 Multiple regression with two numerical explanatory variables

In the last sections we introduced regression modelling where we modelled the relationship between an outcome variable y and a single explanatory variable x, which was either

a numerical or categorical variable. Here, we shall now examine fitting regression models with more than one explanatory variable. This is known as **multiple regression**.

When fitting regression models with multiple explanatory variables, the interpretation of an explanatory variable is made in association with the other variables. For example, if we wanted to model income then we may consider an individuals level of education, and perhaps the wealth of their parents. Then, when interpreting the effect an individuals level of education has on their income, we would also be considering the effect of the wealth of their parents simultaneously, as these two variables are likely to be related.

> **Note**
>
> Additional information and examples can be found in Chapter 6 of An Introduction to Statistical and Data Science via R.

We shall examine a data set within the `ISLR` package, which is an accompanying R package related to the textbook An Introduction to Statistical Learning with Applications in R. We will take a look at the `Credit` data set, which consists of predictions made on the credit card balance of 400 individuals, were the predictions are based on information relating to income, credit limit and the level of education of an individual.

> **Note**
>
> This is a simulated data set and is not based on credit card balances of actual individuals.

The regression model we will be considering contains the following variables:

- the numerical outcome variable y, the credit card balance of an individual; and
- two explanatory variables x_1 and x_2, which are an individuals credit limit and income (in thousands of dollars), respectively.

Lets begin by reading the data:

```
Credit = read.csv("Credit.csv")
```

5.1 Exploratory data analysis

> Task
>
> Start by subsetting the `Credit` data set so that we only have the variables we are interested in, that is, `Balance`, `Limit` and `Income`. Note, it is best to give your new data set a different name than Credit as to not overwrite the original `Credit` data set. Define a new data set named `Cred` containing only the aforementioned variables.
> Take a hint
> You can use the `select` function from the `dplyr` package to select different variables in a data frame.
> Click here to see the solution

```
Cred = Credit %>%
  dplyr::select(c(Balance,Limit,Income))
```

Take a look at summary statistics relating to our newly created data set using the `skim` function:

```
Cred %>%
  skim()
```

Table 12: Data summary

Name	Piped data
Number of rows	400
Number of columns	3
Column type frequency:	
numeric	3
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Balance	0	1	520.02	459.76	0.00	68.75	459.50	863.00	1999.00	□□□□□
Limit	0	1	4735.60	2308.20	855.00	3088.00	4622.50	5872.75	13913.00	□□□□□
Income	0	1	45.22	35.24	10.35	21.01	33.12	57.47	186.63	□□□□□

Now that we are looking at the relationship between an outcome variable and multiple explanatory variables, we need to examine the correlation between each of them. We can examine the correlation between `Balance`, `Limit` and `Income` by creating a table of correlations as follows:

```
Cred %>%
  cor()
```

```
          Balance     Limit    Income
Balance 1.0000000 0.8616973 0.4636565
Limit   0.8616973 1.0000000 0.7920883
Income  0.4636565 0.7920883 1.0000000
```

Question

Why are the diagonal components of our correlation table all equal to 1?

- (A) because variables have been standardized to have an unit variance
- (B) because they are the correlation of a column with itself
- (C) because we have a diagonal covariance-variance matrix

From our correlation table we can see that the correlation between our two explanatory variables is 0.792, which is a strong positive linear relationship. Hence, we say there is a high degree of *collinearity* between our explanatory variables.

Collinearity (or **multicollinearity**) occurs when an explanatory variable within a multiple regression model can be linearly predicted from the other explanatory variables with a high level of accuracy. For example, in this case, since `Limit` and `Income` are highly correlated, we could take a good guess as to an individual's `Income` based on their `Limit`. That is, having one or more highly correlated explanatory variables within a multiple regression model essentially provides us with redundant information. Normally, we would remove one of the highly correlated explanatory variables, however, for the purpose of this example we shall ignore the potential issue of collinearity and carry on. You may want to use the `pairs` function or the `ggpairs` function from the `GGally` package to look at potential relationships between all of the variables within a data set.

> **Note**
>
> When we have several potential explanatory variables a model selection technique can help to identify which explanatory variables are significant predictors (in addition to the others) and which variables should be removed from the model. One procedure that can be used is **stepwise regression**, which implements an automatic procedure for choosing which explanatory variables should be included within the final model. A common stepwise procedure compares models using the model fit criterion **Akaike Information Criterion** (AIC) and can be implemented in R using the `stepAIC` function from the `MASS` library. This procedure allows for forward selection and backward selection (or both), where forward selection starts with the simplest model before iteratively including one explanatory variable at a time until the AIC reaches a minimum. The backward selection approach starts with the most complex model before removing one explanatory variable at a time until the minimium AIC is achieved. We will cover more of this in the next session.

Let's now produce scatterplots of the relationship between the outcome variable and the explanatory variables. First, we shall look at the scatterplot of `Balance` against `Limit`:

```
ggplot(Cred, aes(x = Limit, y = Balance)) +
  geom_point() +
  labs(x = "Credit limit (in $)", y = "Credit card balance (in $)",
       title = "Relationship between balance and credit limit")
```

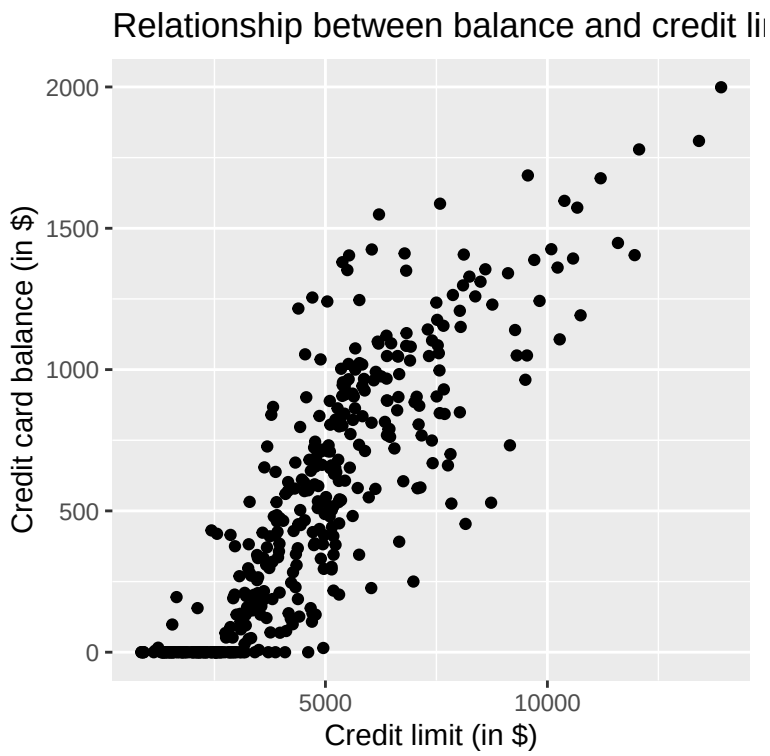


Figure 13: Relationship between balance and credit limit.

Now, let's look at a scatterplot of `Balance` and `Income`:

```
ggplot(Cred, aes(x = Income, y = Balance)) +
  geom_point() +
  labs(x = "Income (in $1000)", y = "Credit card balance (in $)",
       title = "Relationship between balance and income")
```

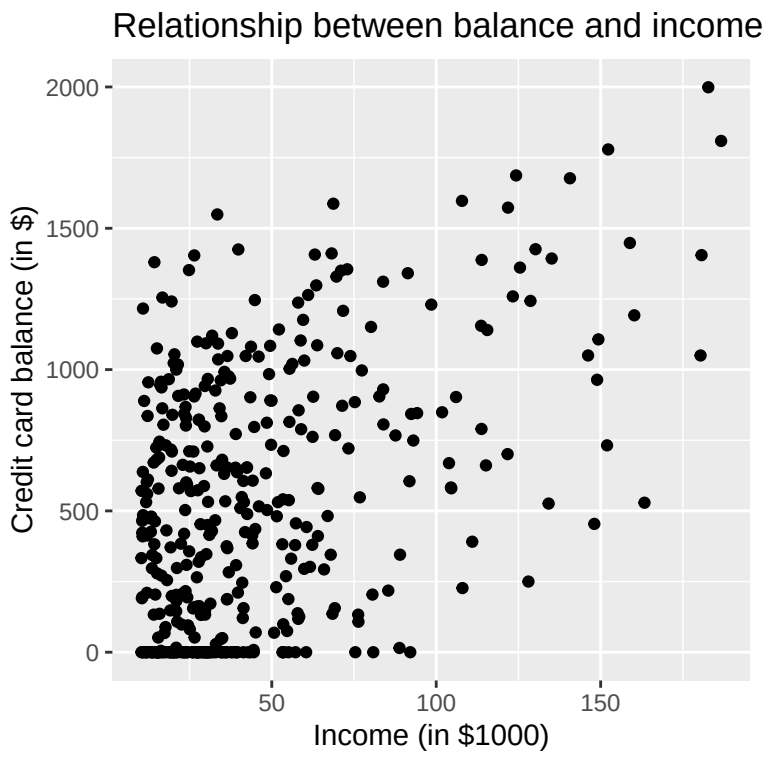


Figure 14: Relationship between balance and income.

The two scatterplots above focus on the relationship between the outcome variable `Balance` and each of the explanatory variables independently. In order to get an idea of the relationship between all three variables we can use the `plot_ly` function within the `plotly` library to plot a 3-dimensional scatterplot as follows:

```
library(plotly)
```

```
Attaching package: 'plotly'

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

    last_plot

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

    filter

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

    layout
```

```
plot_ly(Cred, x = ~Income, y = ~Limit, z = ~Balance,
        type = "scatter3d", mode = "markers")
```

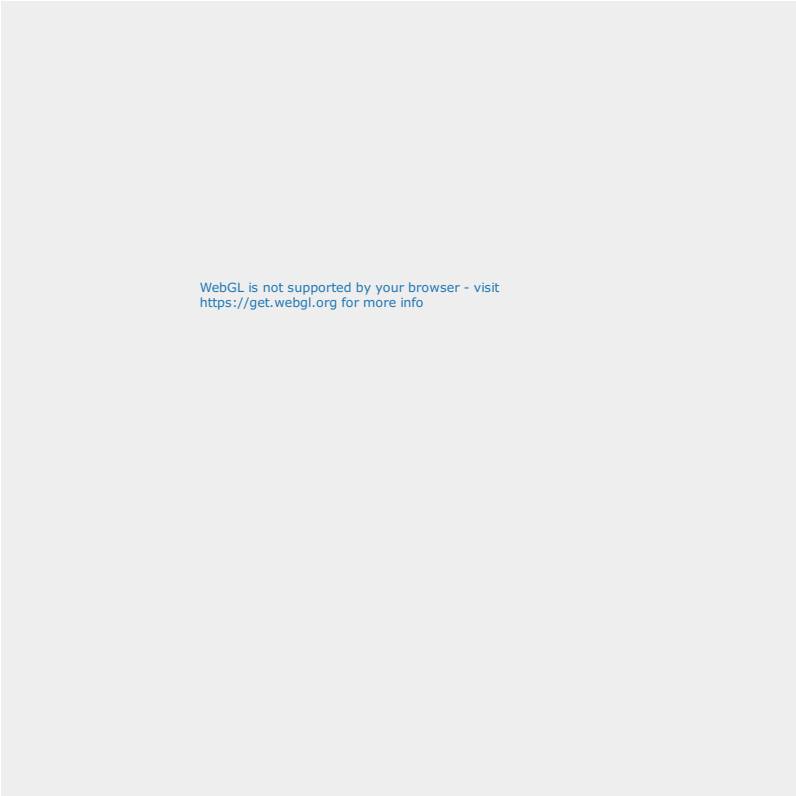


Figure 15: 3D scatterplot of balance, credit limit, and income.

When we fitting our regression model with just a single covariate we looked at the *best-fitting line*. However, now that we have more than one explanatory variable, we are looking at the *best-fitting plane*, which is a 3-dimensional generalisation of the best-fitting line.

5.2 Formal analysis

The multiple regression model we will be fitting to the credit balance data is given as:

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i, \quad \epsilon \sim N(0, \sigma^2),$$

where

- y_i is the balance of the i^{th} individual;
- α is the intercept and positions the best-fitting plane in 3D space;
- β_1 is the coefficient for the first explanatory variable x_1;
- β_2 is the coefficient for the second explanatory variable x_2; and
- ϵ_i is the i^{th} random error component.

Similarly to a simple linear regression, we use the `lm` function to fit the regression model and the `tab_model` function to view our parameter estimates:

```
Balance.model <- lm(Balance ~ Limit + Income, data = Cred)
tab_model(Balance.model,show.ci = F)
```

	Balance	
Predictors	Estimates	p
(Intercept)	-385.18	**<0.001**
Limit	0.26	**<0.001**
Income	-7.66	**<0.001**
Observations	400	
R^2 / R^2 adjusted	0.871 / 0.870	

> **Note**
>
> To include multiple explanatory variables within a regression model we simply use the + sign, that is `Balance ~ Limit + Income`.

How do we interpret our model estimates defining the regression plane? They can be interpreted as follows:

- The **intercept** represents the credit card balance (`Balance`) of an individual who has \$0 for both credit limit (`Limit`) and income (`Income`). However, the interpretation of the intercept in this case is somewhat limited as there are no individuals with \$0 credit limit and income in the data set, with the smallest credit card balance being \$0.
- The coefficient for credit limit (`Limit`) tells us that, *taking all other variables in the model into account*, that there is an associated increase, on average, in credit card balance of \$0.26.
- Similarly, the coefficient for income (`Income`) tells us that, *taking all other variables in the model into account*, that there is an associated decrease, on average, in credit card balance of \$7.66.

What do you notice that is strange about our coefficient estimates given our exploratory data analysis? Well, from our scatterplots of credit card balance against both credit limit and income, we seen that there appeared to be a positive linear relationship. Then, why do we then get a negative coefficient for income (-7.66)? This is due to a phenomenon known as **Simpson's Paradox**. This occurs when there are trends within different categories (or groups) of data, but that these trends disappear when the categories are grouped as a whole. For more details see Section 6.3.4 of An Introduction to Statistical and Data Sciences in R.

5.3 Assessing model fit for multiple regression

Now we need to assess our model assumptions. Similarly to simple regression, our model assumptions are:

1. The deterministic part of the model captures all the non-random structure in the data, i.e. the residuals have mean zero.
2. The scale of the variability of the residuals is constant at all values of the explanatory variables.
3. The residuals are normally distributed.
4. The residuals are independent.
5. The values of the explanatory variables are recorded without error.

First, we need to obtain the fitted values and residuals from our regression model:

```
Balance.model_output <-  Cred %>%
  mutate(Balance_hat  = Balance.model$fitted.values,
         residual = Balance.model$residuals)
```

We can assess our first two model assumptions by producing scatterplots of our residuals against each of our explanatory variables. First, let's begin with the scatterplot of the residuals against credit limit:

```
ggplot(Balance.model_output, aes(x = Limit, y = residual)) +
  geom_point() +
  labs(x = "Credit limit (in $)", y = "Residual", title = "Residuals vs credit limit")  +
  geom_hline(yintercept = 0, col = "blue", linewidth = 1)
```

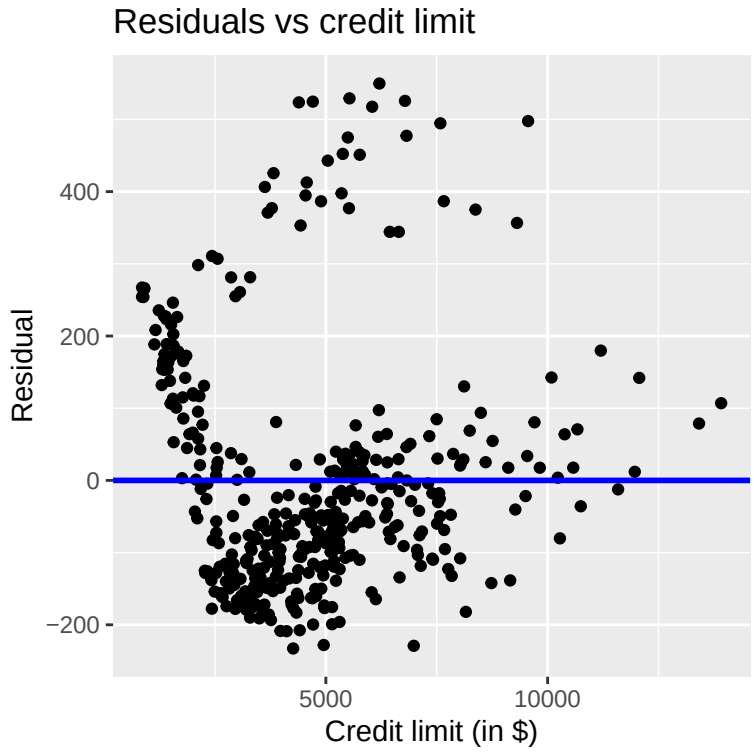


Figure 16: Residuals vs credit limit.

Now, let's plot a scatterplot of the residuals against income:

```
ggplot(Balance.model_output, aes(x = Income, y = residual)) +
  geom_point() +
  labs(x = "Income (in $1000)", y = "Residual", title = "Residuals vs income") +
  geom_hline(yintercept = 0, col = "blue", linewidth = 1)
```

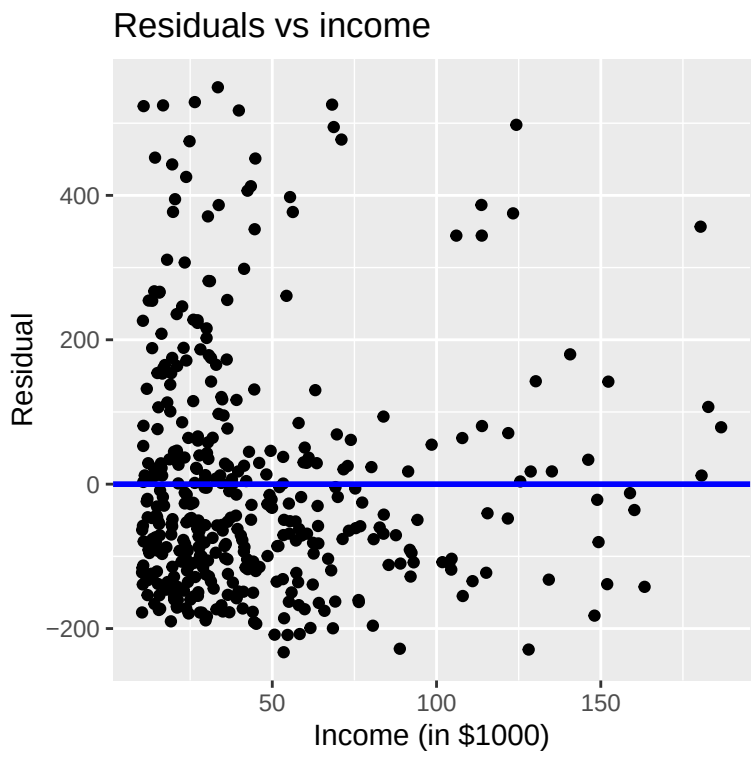


Figure 17: Residuals vs income.

Alternatively, we can use the `check_model` function to produce the standard diagnostic plots for a fitted linear regression model:

```
check_model(Balance.model, check= c("linearity","homogeneity","qq","normality"))
```

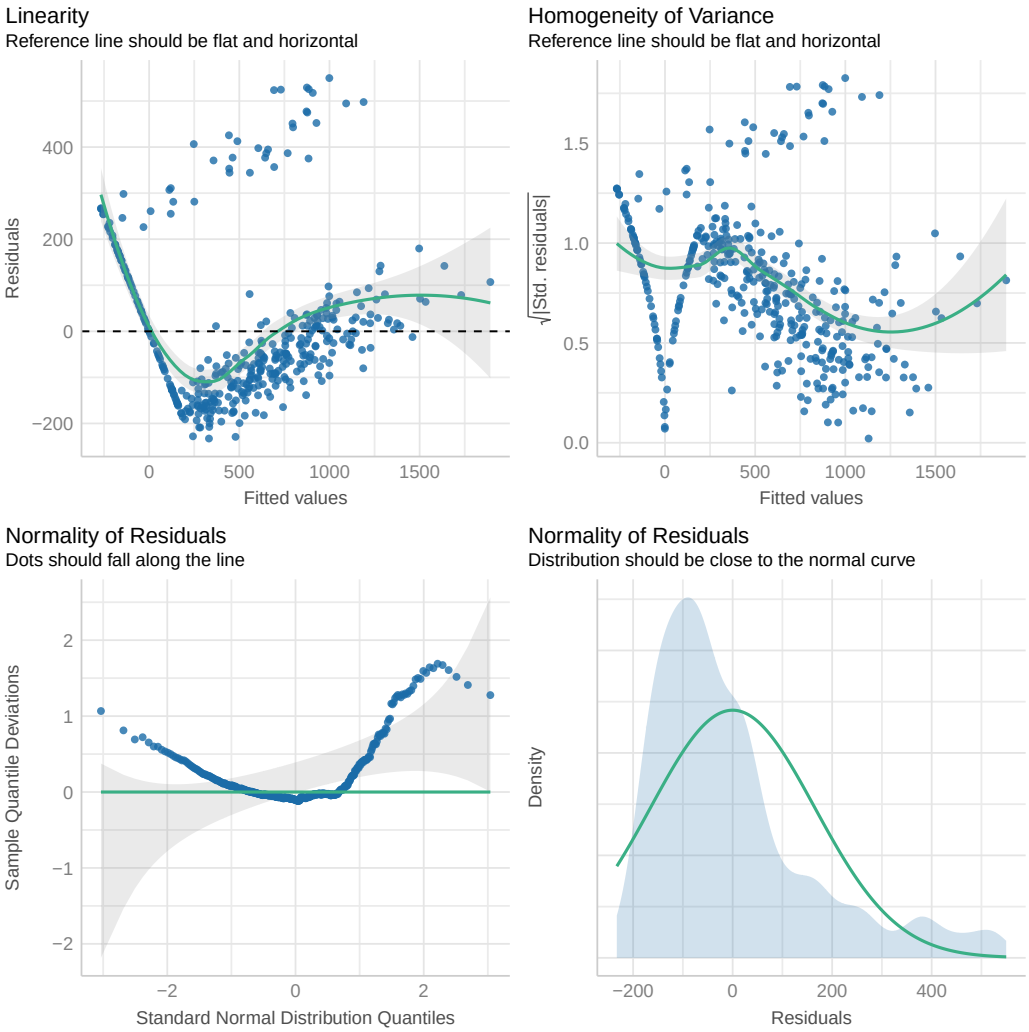


Figure 18: Balance.model Residuals checks.

Question

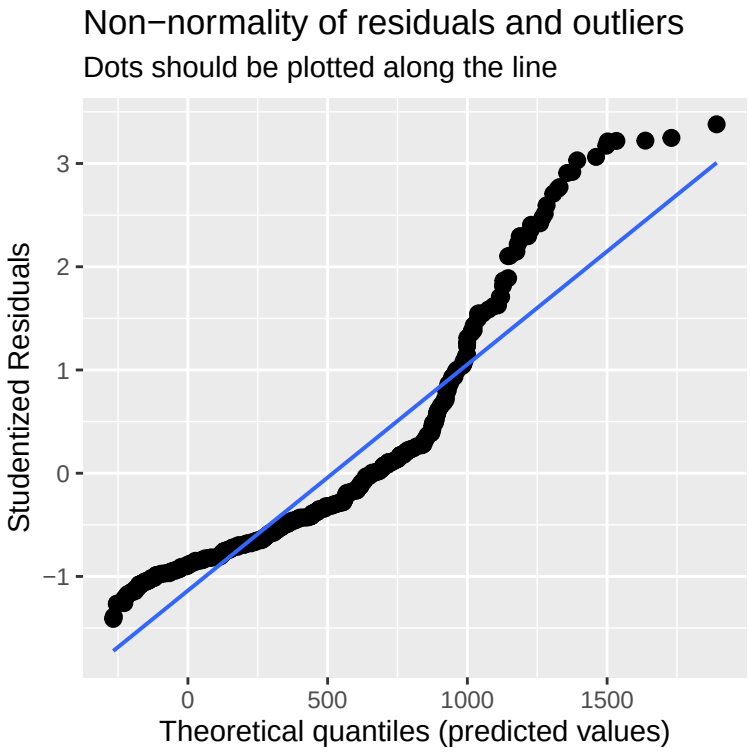
Which assumptions does each of the following plots address?

Task

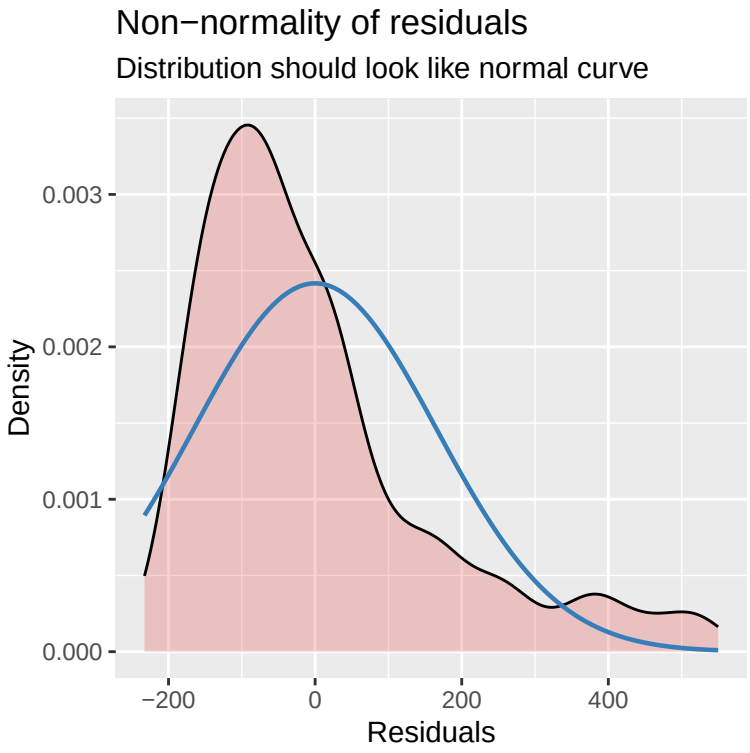
Use the `plot_model` function from the `sjPlot` package to obtain the same diagnostic plots produced via `check_model(Balance.model, check= c(homogeneity","qq","normality"))`.
Click here to see the solution

```
diag_plots = plot_model(Balance.model,type='diag')

# QQ plot
diag_plots[[2]]
```

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

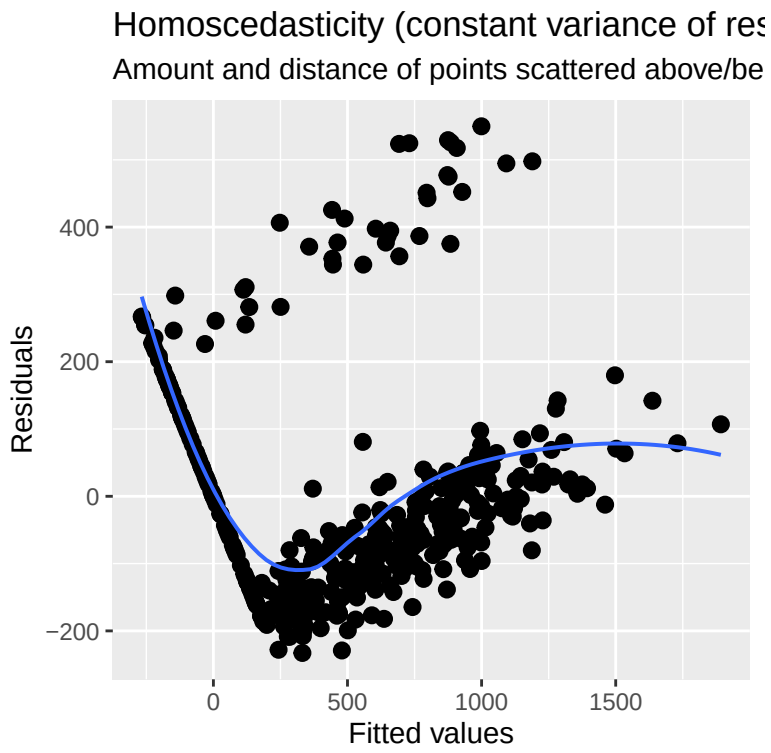



```
# Histrogram/density plot
diag_plots[[3]]
```




```
# Residuals vs fitted values
diag_plots[[4]]
```

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



Lastly, besides our usual diagnostic plots to checks we need to be carefull with the collinearity among our predictor since a high collinearity value may inflate the parameter uncertainty.

To check for colliniearity we can compute the VIF (Variance Inflation Factor) which measures how well a covariate j can be predicted by all other predictors in the model. It measures how much the variance of a regression coefficient is inflated due to collinearity.

It is computed by regressing a predicor x_j against all other predictors in the model and then obtaining the VIF as:

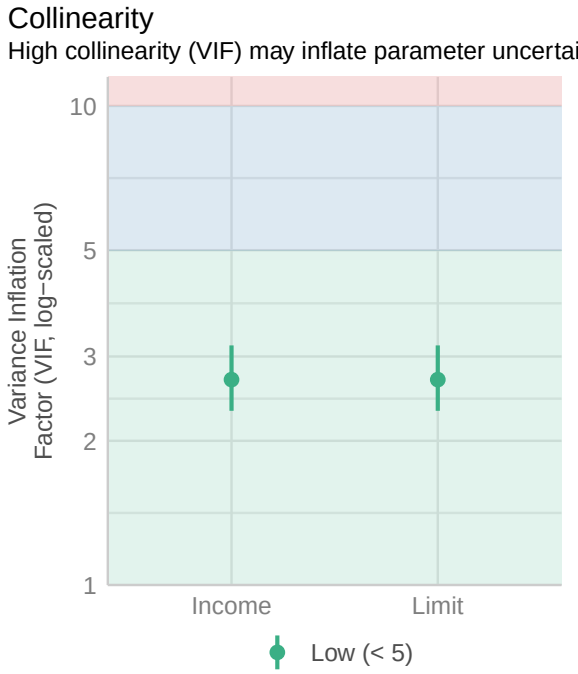
$$VIF_j = \frac{1}{1 - R_j^2}$$

where R_j^2 is the coefficient of determination from this auxiliary regression. In general terms we can interpret VIF as follows:

- VIF = 1: No multicollinearity (predictor is uncorrelated with others).
- 1 $<$ VIF < 5: Moderate correlation (usually acceptable).
- VIF $\geq$ 5: High multicollinearity (may inflate coefficient variance, reducing reliability).

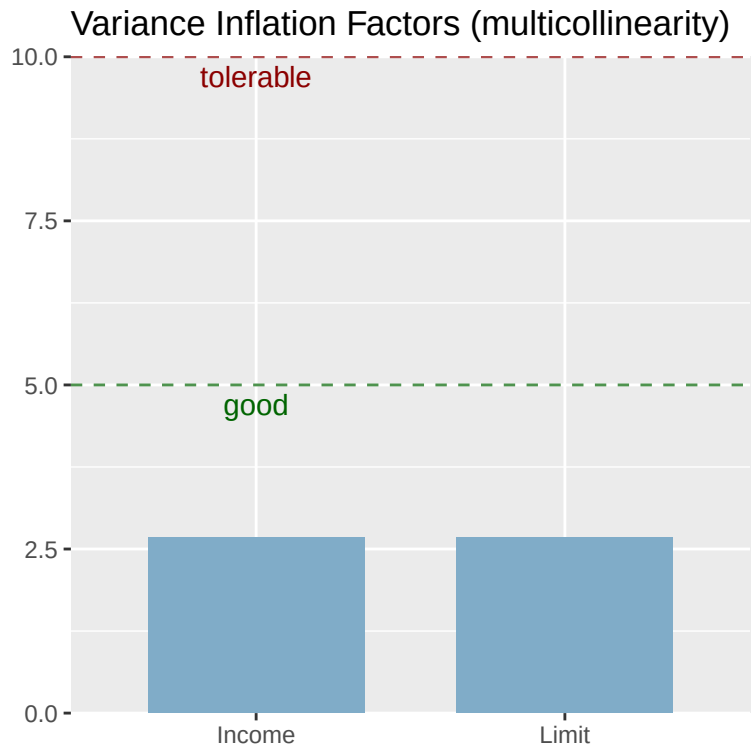
We can check for collinearity using the `check_model` function as follows:

```
check_model(Balance.model,check="vif")
```



Alternatively, the aforementioned `plot_model()` function from `sjPlot` also compute this as part of the default diagnostic plots:

```
diag_plots[[1]]
```



You can further enhance your data analysis skills by completing the additional tasks. Attempt these tasks first, then compare your solutions with the provided answers.