Lecture 2.2 - Simple regression activity

Student

2025-04-01

Table of contents

Simple regression

Planning

Expectations

In this exercise, we are going to try and predict `calories`

1. Pick a predictor variable that you think will affect `calories`

Write down your expected relationship between the two variables – specify what you think the correlation will be (high, medium, low, positive, negative)

2. Create a scatterplot of the relationship between the two variables

You can create a scatterplot (replacing the text `<...>` with your data names) with:

```
ggplot(<dataset>, aes(x=<variable1>, y=<variable2>)) + geom_point()
```

- Make a high-quality scatterplot of your two variables.

- Write down a brief description of the relationship between the two, including your prediction of the correlation of the two and what values you expect the slope and intercept to be of the best fit line

Investigation

3. Calculate the correlation between the two variables and record it.

- Was it stronger or weaker than you expected?

Remember, the code for correlation is:

```
cor(<dataset>$<variable1>, <dataset>$<variable2>, use="complete.obs")
```

Replace `<dataset>` with your dataset's name and `<variable>` with the variable's name that you are analyzing.

4. Now let's create some lines through the data. Below is sample code you will need to copy and paste into a code block.

```
scatterplot <- ggplot(<dataset>, aes(x=<predictor.variable>, y=<response.variable>)) + geom_
```

```
scatterplot + geom_abline(slope=<your.slope.estimate>, intercept=<your.intercept.estimate>)
```

   Remember to replace the parts in `<>` with your own information.

- How well do you feel your best fit line fits the data?

5. Next run a regression to estimate the linear relationship between the two variables. We can do this with:

```
lmodel <- lm(<response.variable> ~ <predictor.variable>, data=<dataset>)
summary(lmodel)
```

Remember to replace `<response.variable>`, `<predictor.variable>` and `<dataset>` with the data and variables you are using.

- How close to your line was the regression line? If it was different, why do you think that happened?
- Interpret your results – for every change in your predictor variable, how much does your response variable change?
- Compared to your expectations, is the predicted change a lot or a little?

Checking your work

7. For the final step of part I, add a regression line to your plot. Do this by inserting the following commands into a new code block:

```
scatterplot <- ggplot(<dataset>, aes(x=<predictor.variable>, y=<response.variable>))+ geom_p

scatterplot + geom_abline(slope=<regression.slope.estimate>, intercept=<regression.intercept
```

Instead of using your estimate from before, this time enter the slope and intercept generated by the regression in the previous step.

- How close was your predicted best fit line compared to the regression line R generated? If they were different, why were they different?
- Would either of your variables benefit from a transformation?

Simple regression extension

Calculating residuals

8. Make a small table using the Quarto (manually make a table in visual mode) built-in table function. Make a table with five columns and four rows. Then pick any three observations in the dataset and, in the table, record the following information:

- Item name
- Item's calories
- Item's observed value for calories
- The predicted value of calories based on the regression model
- The size of the residual.

How large were your residuals? Did the size of the residuals indicate to you that the regression line is a good fit or not?

Residual plotting

9. Make a residual plot & standard deviation

Fortunately, RStudio can easily make a residual plot. When you run a regression, within the stored regression RStudio stores the values of β.

Note that you will need to load `broom` library for this code to work. You will be using the `augment()` function that adds the residuals to a dataset for easy display and manipulation.

Note that in the lecture we viewed the predictor variable on the x axis of the residual plot. Here we are switching to viewing the response variable on the x axis here. If you think about it, the two plots are equivalent, simply rotated. But once we switch to thinking in higher dimensions, the only plot that makes sense is the residual plot with the response variable plotted.

```
# Save the model
<modelname> <- lm(<predictor.variable> ~ <response.variable>, data=<dataset>)
# Create the residuals database
<modelname>.augmented <- augment(<modelname>, <dataset>)

# Residual histogram plot
ggplot(<modelname>.augmented, aes(x=.resid)) +
  geom_histogram(fill="blue4") +
  labs(x="Residuals", y="Count")

# Residual scatterplot
ggplot(<modelname>.augmented, aes(<response variable>, .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "blue", linetype='dashed') +
  labs(y = "Residuals", x="<name of response variable>")
```

- What can you conclude from the residual plots and the standard deviation calculation of the residuals? Does it indicate a good model fit or not?

Model fit

10. Interpret the `R` squared

What can you conclude about your model based on the `R` squared?

11. Which observations are obvious outliers? How does your line of best fit change if you exclude some of the outliers?

    If you are finished with this analysis, conduct the exercise again for a different predictor variable.