EBA3500 Fall 2022
Exercises 4: Inference, transformations, more correlation

Jonas Moss

September 19, 2022

1 Inference

How do you define a p-value, and how do you use them in regression?

2 R^2 and the correlation

a) The R^2 in least absolute deviations regression

Make a function `rsq_lad` that calculates the R^2 for least absolute deviations (R^2_{lad} from now on). Make sure to test it up against at least two examples. Is R^2_{lad} symmetric, i.e., is the R^2_{lad} between y and x the same as the R^2_{lad} between x and y?

Hint: To avoid bugs due to naming of columns, use `data = pd.DataFrame('y' : y,' x' : x)` inside the regression function instead of `data = pd.DataFrame([y, x].T)`. In very special cases, the progam calculating the least absolute deviations will fail to give correct results. You have to return `np.max([0, rsq])`to make the R^2 correct. (This is equivalent to the slope being 0 and and the intercept equal to the median.)

b) Anscombe's Quartet

Anscombe's quartet are four dissimilar datasets with the same correlations / R^2s using least squares. Load the data and plot it using.

```
import seaborn as sns
sns.set_theme(style="ticks")
df = sns.load_dataset("anscombe")
sns.lmplot(x="x", y="y", col="dataset",
hue="dataset", data=df, col_wrap=2,
ci=None, palette="muted", height=4, scatter_kws={"s": 50, "alpha": 1})
plt.show()
```

Calculate the R^2_{lad} values for these four datasets, both $R^2_{lad}(x, y)$ and $R^2_{lad}(y, x)$. Compare the the least squares case.

Hint: You may want to use (and try to understand)

df.groupby("dataset").apply(lambdaz:rsq_lad(z["x"],z["y"]))

Be sure to read the hint in (a). Try to reverse the axes of the plot using x="y", y="x" inside sns.lmplot above.

c) Correlation using least absolute deviations

Using the equivalent definition of the correlation (for least squares) in the slides, propose a definition of the correlation for least absolute deviations. Make a Python function that calculates this "LAD-correlation".

3 Non-linear least squares

3.1 Estimator–plotter function

Make the following function:

```
nls_plotter(x, y, func):
'''''Estimate the parameters of func (as in curve_fit)
using non-linear least squares. Make a scatterplot of
x and y and add the curve defined by func using the estimated
parameters. Return the non-linear R squared.'''''
```

Hint: You may use `*params` to unpack arguments.

3.2 Four parameterized functions

Make the following functions in Python. Make sure they are compatible with `curve_fit`. See this link for more..

$$\frac{a}{1+b^{(x-c)}}+d \qquad \text{(Logistic curve)}$$

$$a+\frac{b}{x+c} \qquad \text{(Hyperbolic curve)}$$

$$a+bx^c \qquad \text{(Logarithmic curve)}$$

$$a+bc^x \qquad \text{(Exponential curve)}$$

3.3 4-plotter

Make a plotting function `grid_plotter` that makes 4 separate plots, one for each of the functions above, in a single window. The plot contains a scatterplot of y vs x and adds the non-linear least squares fitted curves for one of the functions above. The function should return a dictionary with the R^2 for each of the curves. Write a suitable docstring for the function.

Hint: Use e.g. this to make a suitable plot.

Nota bene: Be sure to make the function error-tolerant! For some combinations of data of functions, `curve_fit` will not work at all. Use the `try` and except commands to handle errors arising from curve_fit. See e.g. this link for more information.

3.4 Application

We will look at the dataset obtained from

```
from rdatasets import data
dataset = data("DNase")
dataset = dataset[dataset['Run'] == 1]
x = np.log(dataset["conc"])
y = dataset["density"]
```

Use the `grid_plotter` function on this data. Which functional form fits the data best?