Simple Regression Model

Juergen Meinecke

Roadmap

Selected Topics

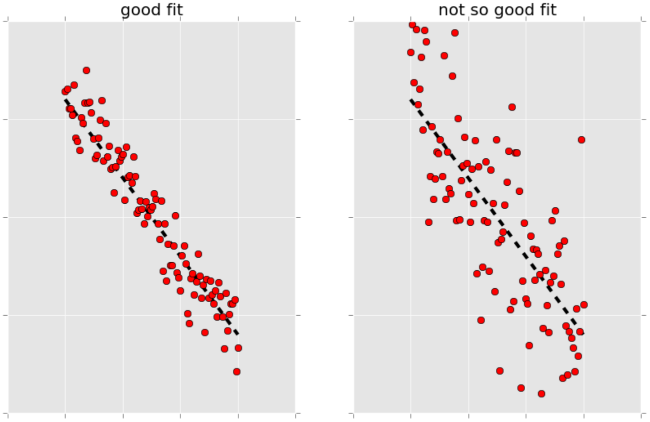
- Measures of Fit

There are two regression statistics that provide measures of how well the regression line “fits” the data:

- regression R^2, and
- standard error of the regression (SER)

Main idea: how closely does the scatterplot “fit” around the regression line?

Graphical illustration of “fit” of the regression line



The regression R^2 is the fraction of the sample variation of Y_i that is explained by the explanatory variable X_i

Total variation in the dependent variable can be broken down as

- total sum of squares (TSS)

  $TSS := \sum_{i=1}^{n}(Y_i - \bar{Y})^2$
- explained sum of squares (ESS)

  $ESS := \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2$
- residual sum of squares (RSS)

  $RSS := \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$

It follows that $TSS = ESS + RSS$

Definition

R^2 is defined by

$$R^2 := \frac{ESS}{TSS}.$$

Corollary

Based on the preceding terminology, it is easy to see that

$$R^2 = 1 - \frac{RSS}{TSS}$$

Therefore,

- $R^2 = 0$ means $ESS = 0$ (the regressor X explains nothing in the variation of the dependent variable Y)
- $R^2 = 1$ means $ESS = TSS$
  (the regressor X explains all the variation of the dependent variable Y)
- $0 \leqslant R^2 \leqslant 1$
- For a regression with a single regressor X, R^2 is the square of the sample correlation coefficient between X and Y
- `Python` routinely calculates and reports R^2 when it runs regressions

In contrast, the standard error of the regression measures the spread of the distribution of the errors

Because you don't observe the errors u_i you use the residuals $\hat{u}_i$ instead

It is defined as the estimator of the standard deviation of u_i:

$$\begin{aligned} SER &:= \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(\hat{u}_i - \bar{\hat{u}})^2} \\ &= \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}\hat{u}_i^2} = \sqrt{\frac{RSS}{n-2}} \end{aligned}$$

The second equality holds because $\bar{\hat{u}} := \frac{1}{n}\sum_{i=1}^{n}\hat{u}_i = 0$

The SER

- has the units of u, which are the units of Y
- measures the spread of the OLS residuals around the estimated PRF

Technical note: why divide by $n-2$ instead of $n-1$?

- Division by $n-2$ is a "degrees of freedom" correction – just like division by $n-1$ in s_Y^2, except that for the SER, two parameters have been estimated (β_0 and β_1), whereas in s_Y^2 only one has been estimated (μ_Y)
- When sample size n is large, it doesn't really matter whether n or $n-1$ or $n-2$ is being used

Simple Regression Model

Juergen Meinecke

Roadmap

Selected Topics

- Binary Regressor

Quite often an explanatory variable is binary

- $X_i = 1$ if small class size (else zero)
- $X_i = 1$ if identify as female (else zero)
- $X_i = 1$ if smokes (else zero)

Binary regressors are called *dummy variables*

So far, we have looked at β_1 as a *slope*

But does this make sense when X_i is binary?

How should we interpret β_1 and its estimator $\hat{\beta}_1$?

The linear model $Y_i = \beta_0 + \beta_1 X_i + u_i$ reduces to

- $Y_i = \beta_0 + u_i$ when $X_i = 0$
- $Y_i = \beta_0 + \beta_1 + u_i$ when $X_i = 1$

Analogously, the population regression functions are

- $E[Y_i|X_i = 0] = \beta_0$
- $E[Y_i|X_i = 1] = \beta_0 + \beta_1$

It therefore follows that

$$\beta_1 = E[Y_i|X_i = 1] - E[Y_i|X_i = 0]$$

In words: the coefficient β_1 captures the difference in group means

Do moms who smoke have babies with lower birth weight?

Python Code

```
> import pandas as pd
> df = pd.read_csv('birthweight.csv')
> smokers = df[df.smoker == 1]
> nonsmokers = df[df.smoker == 0]
> t_test(smokers.birthweight, nonsmokers.birthweight)
> t_test(smokers.birthweight, nonsmokers.birthweight)

Two-sample t-test
Mean in group 1: 3178.831615120275
Mean in group 2: 3432.0599669148055
Point estimate for difference in means: -253.22835179453068
Test statistic: -9.441398919580234
95% confidence interval: (-305.7976345612996, -200.65906902776175)
```

Regression with smoker dummy gives exact same numbers

Python Code (output edited)

```
> import statsmodels.formula.api as smf
> formula = 'birthweight ~ smoker'
> model1 = smf.ols(formula, data=df, missing='drop')
> reg1 = model1.fit(use_t=False)
> print(reg1.summary())

OLS Regression Results
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   3432.0600     11.871    289.115      0.000    3408.793    3455.327
smoker      -253.2284     26.951     -9.396      0.000    -306.052    -200.404
==============================================================================
Notes: [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

- $\hat{\beta}_0$ equal to average birthweight in sub-sample $X_i = 0$
- $\hat{\beta}_1$ equal to difference in average birthweighs b/w groups

Simple Regression Model

Juergen Meinecke

Roadmap

Selected Topics

- Gauss-Markov Theorem

OLS estimator is not the only estimator of the PRF

You can nominate anything you want as your estimator

Similar to lecture 2, here are some alternative estimators:

- $\underset{b_0,b_1}{\text{argmin}} \sum_{i=1}^{n} (Y_i - b_0 - b_1 X_i)^p$,
  where p is any natural number
- $\underset{b_0,b_1}{\text{argmin}} \sum_{i=1}^{n} \left| Y_i - b_0 - b_1 X_i \right|$
  this is called the *least absolute deviations estimator*
- the number 42
  (the 'answer to everything estimator')

Clearly, these are all estimators
(they satisfy the definition given earlier)

Are they sensible estimators?

Clearly, the last one is silly

The point is: there always exist an endless number of possible estimators for any given estimation problem

Most of them do not make any sense

What then constitutes a good estimator?

Let’s determine ‘goodness’ of an estimator by two properties:

1. bias
2. variance

Let’s briefly look at these again

Definition

An estimator $\hat{\theta}$ for an unobserved population parameter θ is **unbiased** if its expected value is equal to θ, that is

$$\mathrm{E}[\hat{\theta}] = \theta$$

Definition

An estimator $\hat{\theta}$ for an unobserved population parameter θ has **minimum variance** if its variance is (weakly) smaller than the variance of any other estimator of θ. Sometimes we will also say that the estimator is **efficient**.

Let's see if the OLS estimator satisfies these two properties

But first we need to take a brief detour:

Definition

An estimator $\hat{\theta}$ is linear in Y_i if it can be written as

$$\hat{\theta} = \sum_{i=1}^{n} a_i Y_i,$$

where the weights a_i are functions of X_i but not of Y_i.

It is easy to see that the OLS estimator is a linear estimator

Definition

A **Best Linear Unbiased Estimator (BLUE)** is an estimator that is linear, unbiased, and has minimal variance (efficient).

If an estimator is BLUE, you can't beat it, it's the optimum

When we did univariate statistics (we only looked at one random variable Y_i) we discovered that the sample average was indeed BLUE

Currently we are doing bivariate statistics (we study the joint distribution between Y_i and X_i)

Our estimator of choice is the OLS estimator

Now, similarly to the sample average in the univariate world, a powerful result holds for the OLS estimator...

Theorem

Under OLS Assumptions 1 through 4a, the OLS estimator

$$\hat{\beta}_0, \hat{\beta}_1 := \underset{b_0, b_1}{argmin} \sum_{i=1}^{n} (Y_i - b_0 - b_1 X_i)^2$$

is BLUE.

The Gauss-Markov theorem provides a theoretical justification for using OLS

This theorem holds only for the subset of estimators that are linear in Y_i

There may be nonlinear estimators that are better

Simple Regression Model

Juergen Meinecke

Roadmap

Selected Topics

- Homoskedasticity versus Heteroskedasticity

We've introduced the idea of homoskedasticity last week

We learned about it in OLS Assumption 4a

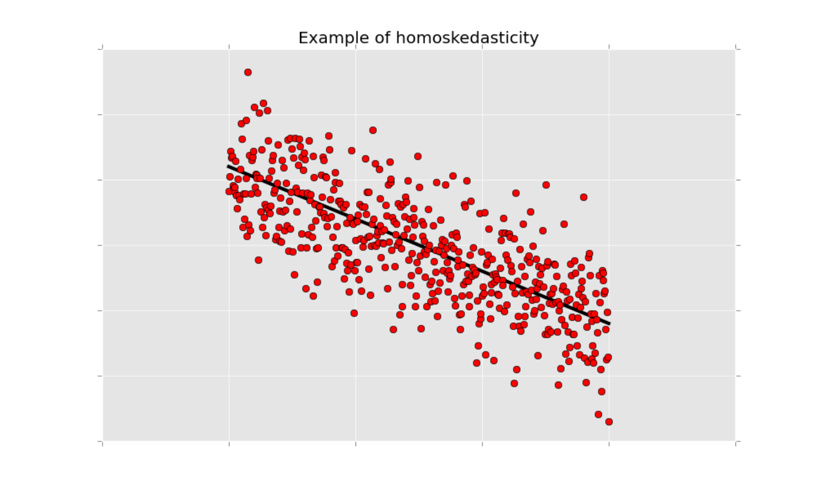
Homoskedasticity concerns the variance of the error terms u_i

Mathematically, the error terms are homoskedastic when

$$\text{Var}\,(u_i|X_i) = \sigma_u^2$$

The essence of this equation is that the variance of u_i *is not a function of* X_i; instead, the variance is just a constant σ_u^2 whatever the value of X_i

Example of homoskedasticity



Scatterplot is distributed evenly around PRF

Variance of error term is constant; does not vary with X_i

But why would we want to assume this?

It seems a bit arbitrary to make an assumption about the variance of the unobserved error term

After all, the error term is unobserved; so why would we make assumptions on the variance of it?

Well, the reason I gave during lecture 5 was that homoskedasticity makes the derivation of the asymptotic distribution a little bit easier

The results just look a little bit cleaner

But homoskedasticity is not a necessary assumption

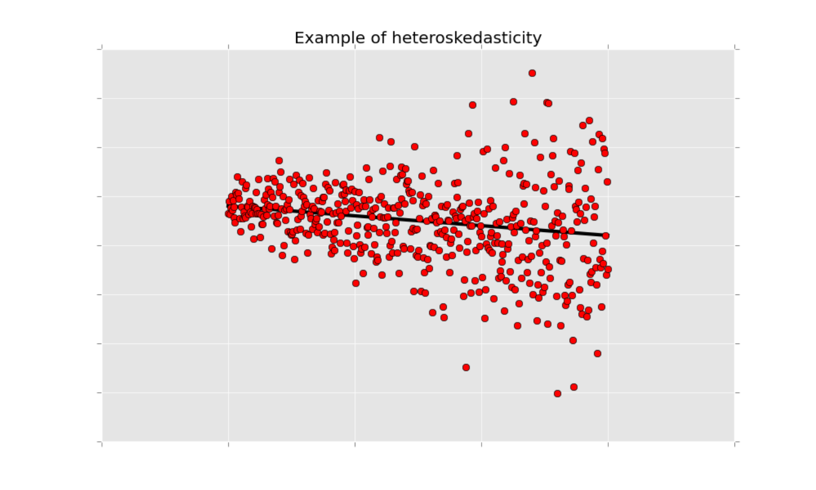
If the error terms are not homoskedastic, what are they?

If they are not homoskedastic, they are called heteroskedastic

How should we think about them?

The next three pictures illustrate…

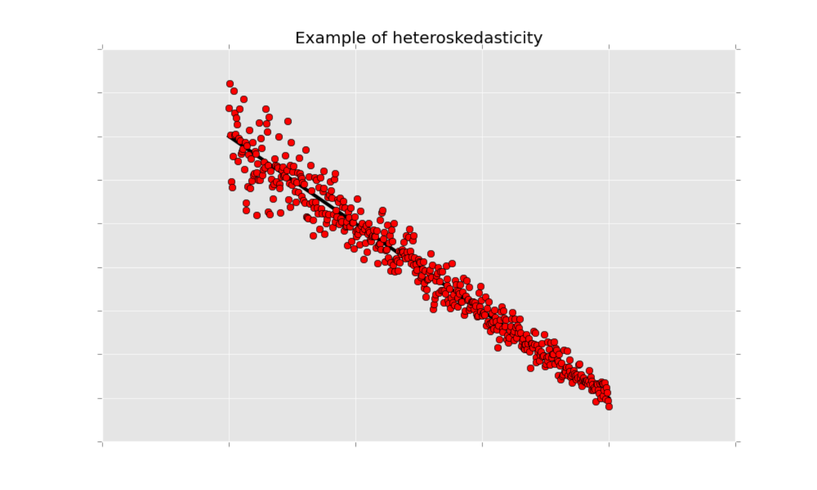
Example of heteroskedasticity



Scatterplot gets wider as X increases

Variance of error term increases in X

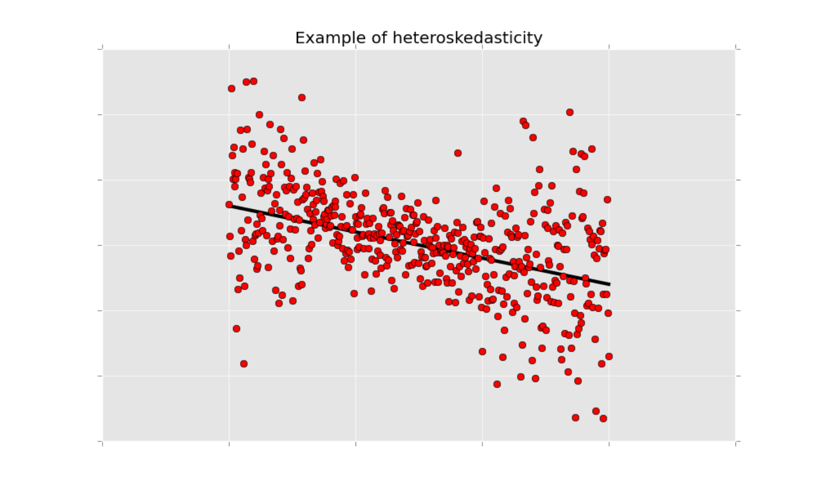
Example of heteroskedasticity



Scatterplot gets narrower as X increases

Variance of error term decreases in X

Example of heteroskedasticity



Scatterplot gets narrower at first but then gets wider again

Variance of error term increases in X, then decreases again

What do these three pictures have in common?

The variance of Y_i itself varies in X_i

The following assumption clarifies what we mean by heteroskedasticity

Assumption (OLS Assumption 4b)

The error terms u_i are ***heteroskedastic*** *if their variance has the following form:*

$$Var\,(u_i|X_i) = \sigma_u^2(X_i),$$

that is, the variance is a function in X_i.

Corollary

If the error terms u_i are not homoskedastic, they are heteroskedastic.

How do the OLS standard errors from last week change if the error terms are heteroskedastic instead of homoskedastic?

Recall the asymptotic distribution of the OLS estimator $\hat{\beta}_1$

$$\hat{\beta}_1 \overset{approx.}{\sim} \mathrm{N}\left(\beta_1, \frac{1}{n}\frac{\sigma_u^2}{\sigma_X^2}\right)$$

This result only holds under OLS Assumptions 1 through 4a

In particular, it only holds under homoskedasticity (Assumption 4a)

If the error terms are heteroskedastic instead, we have to adjust the asymptotic variance

This is tedious, but let's do it!

Recall from lecture 5 how the asymptotic variance collapses to something nice and simple under homoskedasticity:

$$
\begin{aligned}
\text{Var}(\hat{\beta}_1|X_i) = \ldots &= \frac{1}{\left(\sum_{i=1}^n (X_i - \bar{X})^2\right)^2} \sum_{i=1}^n (X_i - \bar{X})^2 \text{Var}(u_i \big| X_i) \\
&= \frac{1}{\left(\sum_{i=1}^n (X_i - \bar{X})^2\right)^2} \sum_{i=1}^n (X_i - \bar{X})^2 \sigma_u^2 \\
&= \frac{\sigma_u^2}{\left(\sum_{i=1}^n (X_i - \bar{X})^2\right)^2} \sum_{i=1}^n (X_i - \bar{X})^2 \\
&\simeq \frac{\sigma_u^2}{(n\sigma_X^2)^2} n\sigma_X^2 \\
&= \frac{1}{n} \frac{\sigma_u^2}{\sigma_X^2},
\end{aligned}
$$

where we plugged in $\sum_{i=1}^n (X_i - \bar{X})^2 \simeq n\sigma_X^2$ and Var $(u_i|X_i) = \sigma_u^2$

In contrast, under heteroskedasticity, we make our lives a bit easier by imposing an asymptotic approximation at a much earlier stage:

$$\begin{aligned}
\mathsf{Var}(\hat{\beta}_1|X_i) = \ldots &= \frac{1}{\left(\sum_{i=1}^n (X_i - \bar{X})^2\right)^2} \sum_{i=1}^n \mathsf{Var}\left((X_i - \bar{X})u_i \middle| X_i\right) \\
&\simeq \frac{1}{(n\sigma_X^2)^2} n\mathsf{Var}\left((X_i - \mu_X)u_i\right) \\
&= \frac{1}{n} \frac{\mathsf{Var}\left((X_i - \mu_X)u_i\right)}{\sigma_X^4}
\end{aligned}$$

(Note: the use of the conditional variance and the subsequent approximation are a bit dubious; the actual math is a bit more complicated and I am taking shortcuts here to make things easy)

Putting things together and invoking the CLT once more

Theorem

The ***asymptotic distribution of the OLS estimator*** $\hat{\beta}_1$ *under OLS Assumptions 1 through 4b is*

$$\hat{\beta}_1 \overset{approx.}{\sim} N\left(\beta_1, \frac{1}{n}\frac{Var\left((X_i - \mu_X)u_i\right)}{\sigma_X^4}\right)$$

A similar theorem holds for $\hat{\beta}_0$, it just looks a little bit uglier

The previous theorem is the basis for deriving confidence intervals for β_1 under heteroskedasticity

With our knowledge from the previous weeks, it is easy to propose a 95% confidence interval

$$CI(\beta_1) := \left[\hat{\beta}_1 - 1.96 \cdot \frac{\sqrt{\text{Var}\left((X_i - \mu_X)u_i\right)}}{\sqrt{n}\sigma_X^2}, \right.$$
$$\left. \hat{\beta}_1 + 1.96 \cdot \frac{\sqrt{\text{Var}\left((X_i - \mu_X)u_i\right)}}{\sqrt{n}\sigma_X^2} \right]$$

Only problem: we do not know $\text{Var}\left((X_i - \mu_X)u_i\right)$ and σ_X

But can estimate them easily instead:

- Var $\left((X_i - \mu_X)u_i\right)$ is estimated by

$$s_{ux}^2 := \frac{1}{n}\sum_{i=1}^{n}\left((X_i - \bar{X})\hat{u}_i\right)^2$$

- σ_X is estimated by s_X

(Do you remember the definition of $\hat{u}_i$ and s_X?)

An operational version of the confidence interval therefore is given by

$$CI(\beta_1) := \left[\hat{\beta}_1 - 1.96 \cdot \frac{s_{ux}}{\sqrt{n} s_X^2}, \hat{\beta}_1 + 1.96 \cdot \frac{s_{ux}}{\sqrt{n} s_X^2}\right]$$

The ratio $s_{ux}/(\sqrt{n} s_X^2)$ is, of course, the standard error under heteroskedasticity

The standard error will differ under homoskedasticity and heteroskedasticity

The standard error under heteroskedasticity has the term s_{ux} in the numerator which makes it seem a little bit more complicated to calculate

But it is actually less complicated than it looks

In practice, `Python` computes this for you anyway

Default in Python is homoskedasticity

Python Code (output edited)

```
> import pandas as pd
> import statsmodels.formula.api as smf
> df = pd.read_csv('caschool.csv')
> formula = 'testscr ~ str'
> model1 = smf.ols(formula, data=df, missing='drop')
> reg1 = model1.fit(use_t=False)
> print(reg1.summary())

OLS Regression Results
==============================================================================
Dep. Variable:                testscr   R-squared:                       0.051
Model:                            OLS   Adj. R-squared:                  0.049
Method:                 Least Squares   F-statistic:                     22.58
No. Observations:                 420
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    698.9330      9.467     73.825      0.000     680.377     717.489
str           -2.2798      0.480     -4.751      0.000      -3.220      -1.339
==============================================================================
Notes: [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

New way to do things:

Python Code (output edited)

```
> reg1_heterosk = model1.fit(cov_type='HC1', use_t=False)
> print(reg1_heterosk.summary())

OLS Regression Results
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    698.9330     10.364     67.436      0.000     678.619     719.247
str           -2.2798      0.519     -4.389      0.000      -3.298      -1.262
==============================================================================
Notes: [1] Standard Errors are heteroscedasticity robust (HC1)
```

Using the option `cov_type='HC1'` inside `ols.fit()` is `Python`'s way of adjusting for heteroskedasticity

This is called the *heteroskedasticity robust* option

(Aside: `cov_type='HC1'` makes the same standard error adjustment as Stata's `robust`)

Homoskedastic standard errors are only correct if OLS Assumption 4a is satisfied

Heteroskedastic standard errors are correct under both OLS Assumption 4a and Assumption 4b

Practical implication

- If you know for sure that the error terms are homoskedastic, you should simply use `Python`'s `ols.fit()`
- If you know for sure that the error terms are heteroskedastic, you should use `Python`'s `ols.fit(cov_type='HC1')`
- If you do not know for sure, it is always safer to use heteroskedasticity robust standard errors