```yaml
Course: Regression_Models
Lesson: Residuals


- Class: text
  Output: "Residuals. (Slides for this and other Data Science courses may be found at github https://github.com/DataScienceSpecialization/courses. If you care to use them, they must be downloaded as a zip file and viewed locally. This lesson corresponds to Regression_Models/01_03_ols. Galton data is from John Verzani's website, http://wiener.math.csi.cuny.edu/UsingR/)"

- Class: text
  Output: This lesson will focus on the residuals, the distances between the actual children's heights and the estimates given by the regression line. Since all lines are characterized by two parameters, a slope and an intercept, we'll use the least squares criteria to provide two equations in two unknowns so we can solve for these parameters, the slope and intercept.

- Class: text
  Output: The first equation says that the "errors" in our estimates, the residuals, have mean zero. In other words, the residuals are "balanced" among the data points; they're just as likely to be positive as negative. The second equation says that our residuals must be uncorrelated with our predictors, the parents’ height. This makes sense - if the residuals and predictors were correlated then you could make a better prediction and reduce the distances (residuals) between the actual outcomes and the predictions.


- Class: cmd_question
  Output: We'll demonstrate these concepts now. First regenerate the regression line and call it fit. Use the R function lm. Recall that by default its first argument is a formula such as "child ~ parent" and its second is the dataset, in this case galton.
  CorrectAnswer: fit <- lm(child ~ parent, galton)
  AnswerTests: creates_lm_model('fit <- lm(child ~ parent, galton)')
  Hint: Type "fit <- lm(child ~ parent, galton)" at the R prompt.

- Class: cmd_question
  Output: Now we'll examine fit to see its slope and intercept. The residuals we're interested in are stored in the  928-long vector fit$residuals. If you type fit$residuals you'll see a lot of numbers scroll by which isn't very useful; however if you  type "summary(fit)" you will see a more concise display of the regression data. Do this now.
  CorrectAnswer: summary(fit)
  AnswerTests: omnitest(correctExpr='summary(fit)')
  Hint: Type "summary(fit)" at the R prompt.

- Class: cmd_question
  Output: First check the mean of fit$residuals to see if it's close to 0.
  CorrectAnswer: mean(fit$residuals)
  AnswerTests: omnitest(correctExpr='mean(fit$residuals)')
  Hint: Type "mean(fit$residuals)" at the R prompt.

- Class: cmd_question
  Output: Now check the correlation between the residuals and the predictors. Type "cov(fit$residuals, galton$parent)" to see if it's close to 0.
  CorrectAnswer: cov(fit$residuals,galton$parent)
  AnswerTests: ANY_of_exprs('cov(fit$residuals,galton$parent)','cov(galton$parent,fit$residuals)')
  Hint: Type "cov(fit$residuals, galton$parent)" at the R prompt.

- Class: text
  Output: As shown algebraically in the slides, the equations for the intercept and slope are found by supposing a change is made to the intercept and slope. Squaring out the resulting expressions produces three summations. The first sum is the original term squared, before the slope and intercept were changed. The third sum totals the squared changes themselves. For instance, if we had changed fit’s intercept by adding 2, the third sum would be the total of 928 4’s. The middle sum is guaranteed to be zero precisely when the two equations (the conditions on the residuals) are satisfied.
```

```yaml

- Class: text
  Output: We'll verify these claims now. We've defined for you two R functions, est and sqe. Both take two inputs, a slope and an intercept. The function est calculates a child's height (y-coordinate) using the line defined by the two parameters, (slope and intercept), and the parents' heights in the Galton data as x-coordinates.

- Class: mult_question
  Output: Let "mch" represent the mean of the galton childrens' heights and "mph" the mean of the galton parents' heights. Let "ic" and "slope" represent the intercept and slope of the regression line respectively. As shown in the slides and past lessons, the point (mph,mch) lies on the regression line. This means
  AnswerChoices: mch = ic + slope*mph; mph = ic + slope*mch; I haven't the slightest idea.
  CorrectAnswer: mch = ic + slope*mph
  AnswerTests: omnitest(correctVal='mch = ic + slope*mph')
  Hint: A line is the set of all points (x,y) satisfying the equation y = mx + b, where m is the slope of the line and b is its intercept. Remember that the point (mph,mch) lies on the regression line with intercept ic and slope "slope".

- Class: text
  Output: The function sqe calculates the sum of the squared residuals, the differences between the actual children's heights and the estimated heights specified by the line defined by the given parameters (slope and intercept).  R provides the function deviance to do exactly this using a fitted model (e.g., fit) as its argument. However, we provide sqe because we'll use it to test regression lines different from fit.

- Class: text
  Output: We'll see that when we vary or tweak the slope and intercept values of the regression line which are stored in fit$coef, the resulting squared residuals are approximately equal to the sum of two sums of squares - that of the original regression residuals and that of the tweaks themselves. More precisely, up to numerical error,

- Class: text
  Output: sqe(ols.slope+sl,ols.intercept+ic) == deviance(fit) + sum(est(sl,ic)^2 )

- Class: text
  Output: Equivalently, sqe(ols.slope+sl,ols.intercept+ic) == sqe(ols.slope, ols.intercept) + sum(est(sl,ic)^2 )


- Class: text
  Output: The left side of the equation represents the squared residuals of a new line, the "tweaked" regression line. The terms "sl" and "ic" represent the variations in the slope and intercept respectively. The right side has two terms. The first represents the squared residuals of the original regression line and the second is the sum of squares of the variations themselves.

- Class: cmd_question
  Output: We'll demonstrate this now. First extract the intercept from fit$coef and put it in a variable called ols.ic . The intercept is the first element in the fit$coef vector, that is fit$coef[1].
  CorrectAnswer: ols.ic <- fit$coef[1]
  AnswerTests: omnitest(correctExpr='ols.ic <- fit$coef[1]')
  Hint: Type "ols.ic <- fit$coef[1]" at the R prompt.

- Class: cmd_question
  Output: Now extract the slope from fit$coef and put it in the variable ols.slope; the slope is the second element in the fit$coef vector, fit$coef[2].
  CorrectAnswer: ols.slope <- fit$coef[2]
  AnswerTests: omnitest(correctExpr='ols.slope <- fit$coef[2]')
  Hint: Type "ols.slope <- fit$coef[2]" at the R prompt.

- Class: figure
  Output: Now we'll show you some R code which generates the left and right sides of this equation.  Take a moment to look it over. We've formed two 6-long vectors of variations, one for the slope and one for the intercept. Then we have two "for" loops
```

```
  to generate the two sides of the equation.
  Figure: demofile.R
  FigureType: new

- Class: cmd_question
  Output: Subtract the right side, the vector rhs, from the left, the vector lhs, to see the relationship between them. You should get a vector of very small, almost 0, numbers.
  CorrectAnswer: lhs-rhs
  AnswerTests: omnitest(correctExpr='lhs-rhs')
  Hint: Type "lhs-rhs" at the R prompt.

- Class: cmd_question
  Output: You could also use the R function all.equal with lhs and rhs as arguments to test for equality. Try it now.
  CorrectAnswer: all.equal(lhs,rhs)
  AnswerTests: ANY_of_exprs('all.equal(lhs,rhs)','all.equal(rhs,lhs)')
  Hint: Type "all.equal(lhs,rhs)" at the R prompt.

- Class: cmd_question
  Output: Now we'll show that the variance in the children's heights is the sum of the variance in the OLS estimates and the variance in the OLS residuals. First use the R function var to calculate the variance in the children's heights and store it in the variable varChild.
  CorrectAnswer: varChild <- var(galton$child)
  AnswerTests: omnitest(correctExpr='varChild <- var(galton$child)')
  Hint: Type "varChild <- var(galton$child)" at the R prompt.

- Class: cmd_question
  Output: Remember that we've calculated the residuals and they're stored in fit$residuals. Use the R function var to calculate the variance in these residuals now and store it in the variable varRes.
  CorrectAnswer: varRes <- var(fit$residuals)
  AnswerTests: omnitest(correctExpr='varRes <- var(fit$residuals)')
  Hint: Type "varRes <- var(fit$residuals)" at the R prompt.

- Class: cmd_question
  Output: Recall that the function "est" calculates the estimates (y-coordinates) of values along the regression line defined by the variables "ols.slope" and "ols.ic". Compute the variance in the estimates and store it in the variable varEst.
  CorrectAnswer: varEst <- var(est(ols.slope, ols.ic))
  AnswerTests: omnitest(correctExpr='varEst <- var(est(ols.slope, ols.ic))')
  Hint: Type "varEst <- var(est(ols.slope, ols.ic))" at the R prompt.

- Class: cmd_question
  Output: Now use the function all.equal to compare varChild and the sum of varRes and varEst.
  CorrectAnswer: all.equal(varChild,varEst+varRes)
  AnswerTests: ANY_of_exprs('all.equal(varChild,varEst+varRes)','all.equal(varEst+varRes,varChild)','all.equal(varChild,varRes+varEst)','all.equal(varRes+varEst,varChild)')
  Hint: Type "all.equal(varChild,varEst+varRes)" at the R prompt.


- Class: text
  Output: Since variances are sums of squares (and hence always positive), this equation which we've just demonstrated,  var(data)=var(estimate)+var(residuals), shows that the variance of the estimate is ALWAYS less than the variance of the data.

- Class: mult_question
  Output: Since  var(data)=var(estimate)+var(residuals) and variances are always positive,  the variance of residuals
  AnswerChoices: is less than the variance of data; is greater than the variance of data; is unknown without actual data
  CorrectAnswer: is less than variance of data
  AnswerTests: omnitest(correctVal='is less than the variance of data')
  Hint: The equation says var(residuals)=var(data)-var(estimate); we're subtracting a positive number from var(data) to give us var(residuals)

```

```
- Class: text
  Output: The two properties of the residuals we've emphasized here can be applied to datasets which have multiple predictors. In this lesson we've loaded the dataset attenu which gives data for 23 earthquakes in California. Accelerations are estimated based on two predictors, distance and magnitude.


- Class: cmd_question
  Output: Generate the regression line for this data. Type efit <- lm(accel ~ mag+dist, attenu) at the R prompt.
  CorrectAnswer: efit <- lm(accel ~ mag+dist, attenu)
  AnswerTests: creates_lm_model('efit <- lm(accel ~ mag+dist, attenu)')
  Hint: Type "efit <- lm(accel ~ mag+dist, attenu)" at the R prompt.

- Class: cmd_question
  Output: Verify the mean of the residuals is 0.
  CorrectAnswer: mean(efit$residuals)
  AnswerTests: omnitest(correctExpr='mean(efit$residuals)')
  Hint: Type "mean(efit$residuals)" at the R prompt.

- Class: cmd_question
  Output: Using the R function cov verify the residuals are uncorrelated with the magnitude predictor, attenu$mag.
  CorrectAnswer: cov(efit$residuals, attenu$mag)
  AnswerTests: ANY_of_exprs('cov(efit$residuals, attenu$mag)','cov(attenu$mag,efit$residuals)')
  Hint: Type "cov(efit$residuals, attenu$mag)" at the R prompt.

- Class: cmd_question
  Output: Using the R function cov verify the residuals are uncorrelated with the distance predictor, attenu$dist.
  CorrectAnswer: cov(efit$residuals, attenu$dist)
  AnswerTests: ANY_of_exprs('cov(efit$residuals, attenu$dist)','cov(attenu$dist,efit$residuals)')
  Hint: Type "cov(efit$residuals, attenu$dist)" at the R prompt.

- Class: text
  Output: Congrats! You've finished the course on Residuals. We hope it hasn't left a bad taste in your mouth.

- Class: mult_question
  Output: "Would you like to receive credit for completing this course on
    Coursera.org?"
  CorrectAnswer: NULL
  AnswerChoices: Yes;No
  AnswerTests: coursera_on_demand()
  Hint: ""
```