Data Science Concepts and Analysis

Week 6: The simple linear model

- Statistical models
- The simple linear model

Project comments

This week you and your group should be looking around for a dataset you'd like to work with and getting started on your plan report.

The data should meet some minimum requirements:

- less than 100MB raw file;
- some documentation available;
- at least 100 observations x 4 variables (tidy);
- no more than 100K observations x 100 variables (tidy).

The plan report requires you to tidy up the data, but is otherwise mostly a description of the dataset and background information.

- mainly a chance to get feedback from us;
- there's a template you can fill out as you go;
- your plan doesn't have to pan out, and that's okay -- you can't know what an analysis will produce before you start.

This week: the simple linear model

Objective: introduce statistical models in general and the simple linear model in particular.

- **Statistical models**
    - What makes a model 'statistical'?
    - Goals of modeling: prediction, description, and inference
    - When to avoid models
- **The simple linear model**
    - Line of best fit by least squares
    - A model for the error distribution
    - The simple linear model
    - Interpretation of estimates
    - Uncertainty quantification

Statistical models

- What makes a model 'statistical'?
- Goals of modeling: prediction, description, and inference
- When to avoid models

Models, in general

> *A model is an idealized representation of a system. You likely use models every day. For instance, a weather forecast is [based on] a model. (PTDS)*

This is a pretty general definition.

In the context of quantitative fields, a **model** is typically a ***mathematical description*** of some system.

So what makes a model a *statistical* one?

What makes a model 'statistical'?

One straightforward view is that a **statistical model** is simply a ***probability distribution for a dataset***. Under this view:

> *A statistical model represents [a random] data-generating process.*

The word *process* is important there.

- For a probability distribution to provide a sensible description of a dataset, one needs to be able to at least imagine collecting multiple datasets with the same basic structure.
- So implicit in any statistical model there's an idea of a fixed *process* by which the data are collected.

That's why we spent all that time talking about sampling design!

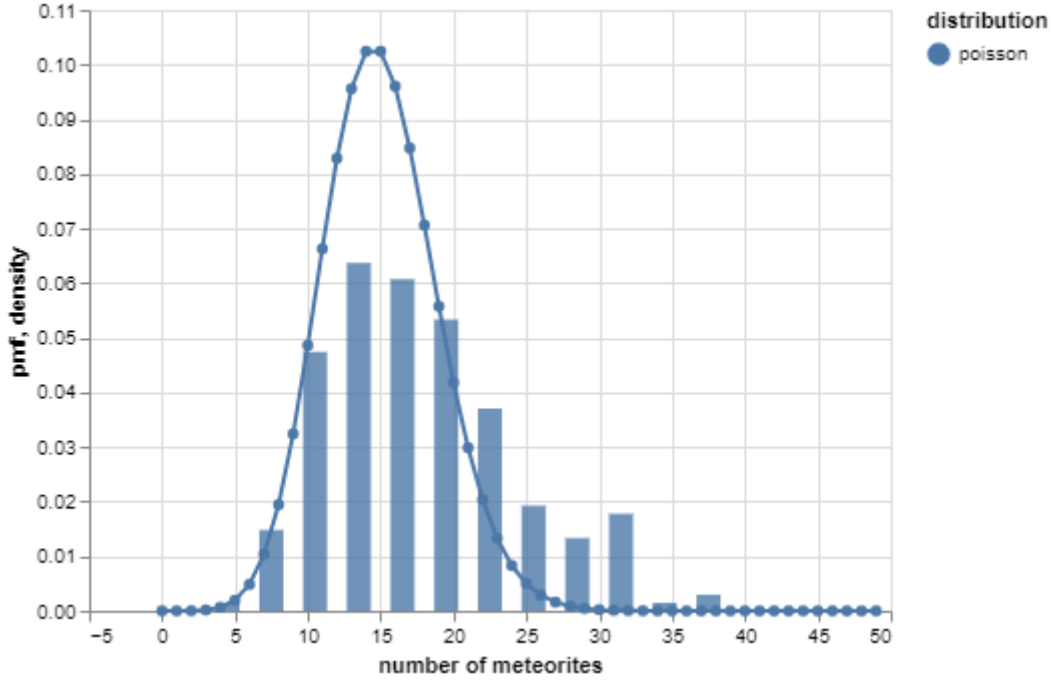
- A good sampling design fixes the process by which data are collected.
- That makes it possible to use statistical models in a meaningful way.

Simple example: univariate models

All the distributions you learned in 120A are very simple statistical models for univariate data.

For example, suppose you have a dataset comprising the number of meteorites that hit earth on each of 225 days.

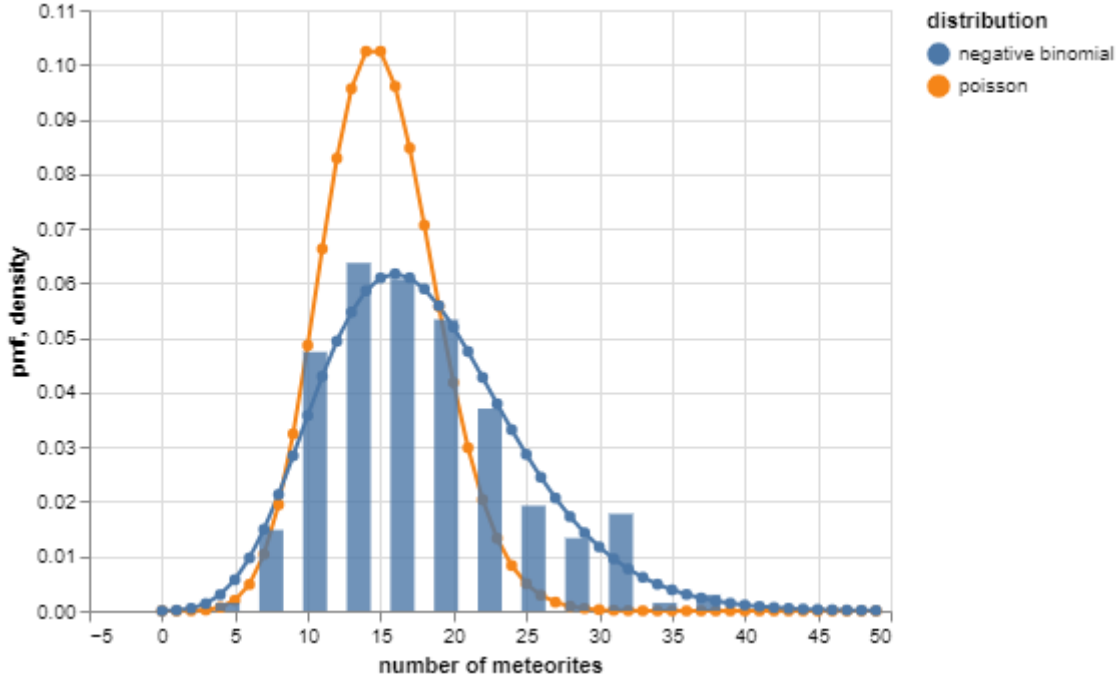
The Poisson distribution provides one possible model for the data -- specifically, that the counts are independent Poisson random variables.



But it doesn't quite match the distribution of values closely enough.

Simple example

The negative binomial does much better here:



Why model the data-generating process?

A good description of a data-generating process usually captures two aspects of a system:

- the deterministic aspects, allowing one to identify structure in the data; and
- the random aspects or 'noise', allowing one to quantify uncertainty.

In the univariate example, the better model captured both the most common value (a kind of structure) and the variation (noise).

Modeling goals

Models serve one of three main purposes:

- **Prediction**: predict new data before it is observed.
- **Inference**: make conclusions about a larger population than the observed data.
- **Description**: less common, but sometimes models provide a convenient description of observed data.

We probably wouldn't use a univariate model in a predictive capacity, but one could be used to make an inference -- we could estimate the probability that over 40 meteorites (rarely observed) hit earth any given day.

Models you've seen already

The exploratory analysis techniques you've seen are actually very flexible models often used for descriptive purposes.

- Kernel density estimates are models for univariate data.
- LOESS curves are models for trends in bivariate data.
- Principal components are models for correlation structures.

It's a little tricky to see, but these correspond to *classes* of probability distributions rather than specific ones. That's why they're so flexible.

It’s okay not to model data

There are a lot of situations when modeling simply isn't appropriate or feasible. Two especially common scenarios are described below.

Sketchy sampling

Every statistical model makes some set of assumptions that translate to specific ways data were collected.

These don't always need to hold exactly, but it's probably best to consider other possibilities if:

- the way data were collected is highly opaque or nonsystematic;
- the sampling design or measurement procedures have serious flaws or inconsistencies;
- you have no information whatsoever about the data source.

Sparse data

Fitted models are highly variable (sensitive to the specific dataset, and so less reliable) with small quantities of data. Typically, most models only require modest amounts of data for reliable fitting, perhaps as few as 10-15 observations.

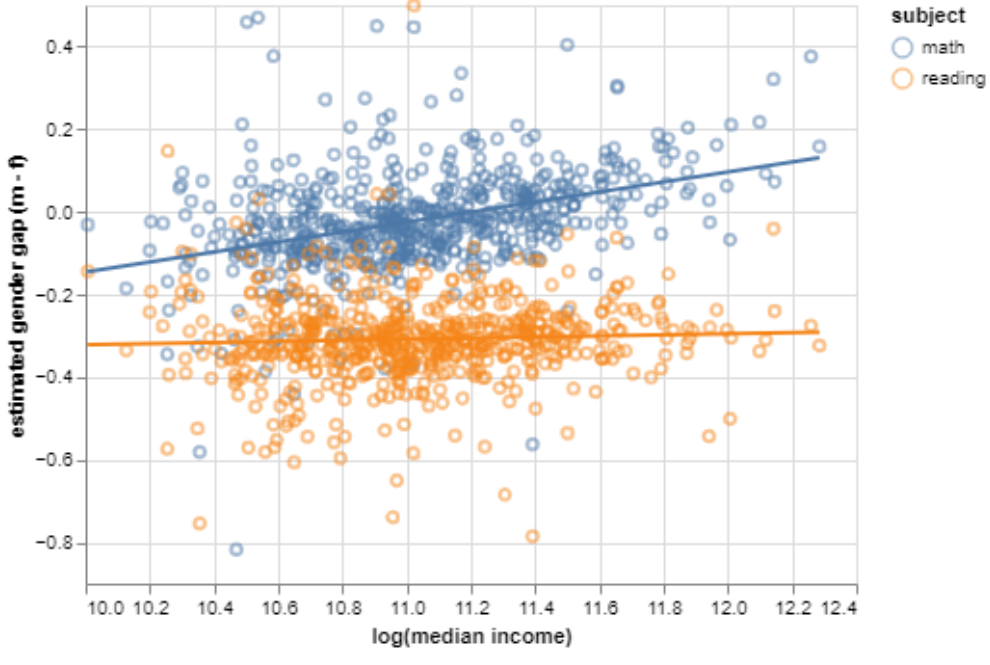
It's probably best to seek other strategies when the number of observations is exceedingly low relative to the modeling objective.

The simple linear model

- Line of best fit by least squares
- The error distribution
- The simple linear model
- Interpretation of estimates
- Uncertainty quantification

A familiar example

In HW2, you generated this plot:



You may not have realized it at the time, but those lines are *simple linear models*.

- They describe the mean gaps as linear functions of log median income.

Applications of linear models

Linear models can be used for prediction, inference, or both.

- Predict the gender achievement gaps for a new district based on median income in the district.
- Quantify the association between median income and achievement gaps.

We're going to talk in detail about the model itself:

- definition;
- estimation;
- assumptions.

Data setting

Let's first introduce the kind of data that the simple linear model describes.

- There are two variables, X and Y.
- The data values are n observations of these two variables:

$$(x_1, y_1), \ldots, (x_n, y_n)$$

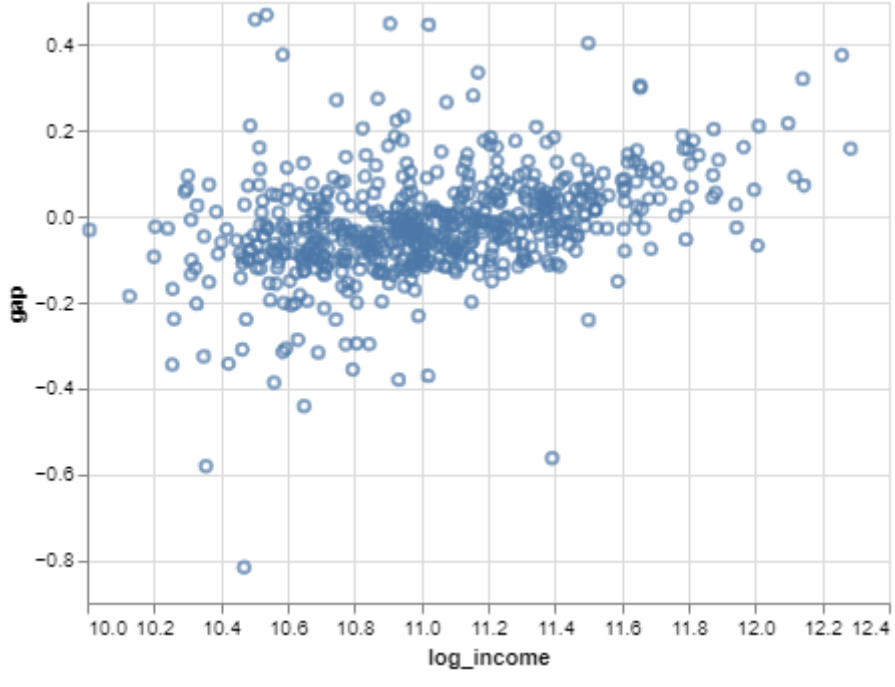
The notation in tuples indicates the pairing of the values when measured on the same observational unit.

If it helps, think of the tuples as an alternative description of a dataframe with n rows and 2 columns:

X	Y
x_1	y_1
x_2	y_2
$\vdots$	$\vdots$
x_n	y_n

Data setting

The notation above is just a mathematical description of data that looks like this:



In our notation, X would represent log median income, and Y would represent the math gap.

Data setting

So to better align the notation we'll be using with our example, the data in tabular form are:

In [2]:
```
# import grade-aggregated seda data from hw2
seda = pd.read_csv('data/seda.csv')

# filter to math and remove NaNs
regdata = seda[seda.subject == 'math'].dropna().drop(columns = 'subject').set_index('id')
regdata.head(3)
```

Out[2]:

id	log_income	gap
600001	11.392048	-0.562855
600006	11.607236	0.061163
600011	10.704570	-0.015417

In [3]:
```
n, p = regdata.shape
n
```

Out[3]:
```
625
```

The tuples would be:

$$(\text{log\_income}_1, \text{gap}_1)\ ,\ (\text{log\_income}_2, \text{gap}_2)\ ,\ \ldots\ ,\ (\text{log\_income}_{625}, \text{gap}_{625})$$

Or more specifically:

Lines and data

A line in slope-intercept form is given by the equation:

$$y = ax + b$$

Data values never fall exactly on a line. So in general for every a, b:

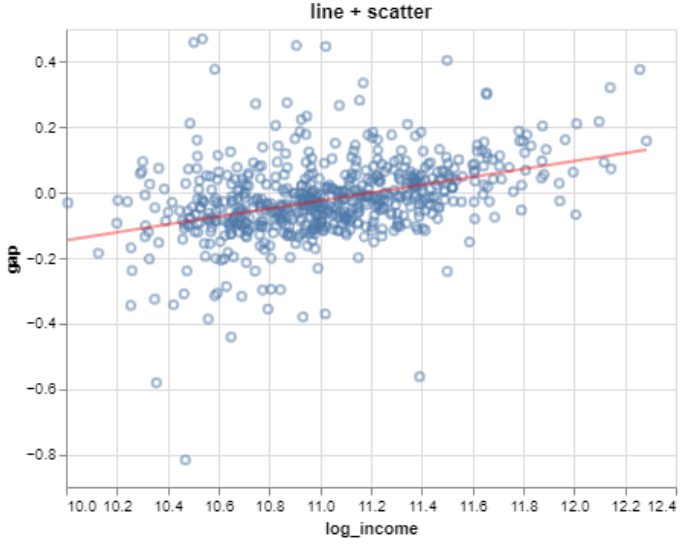
$$y_i \neq ax_i + b$$

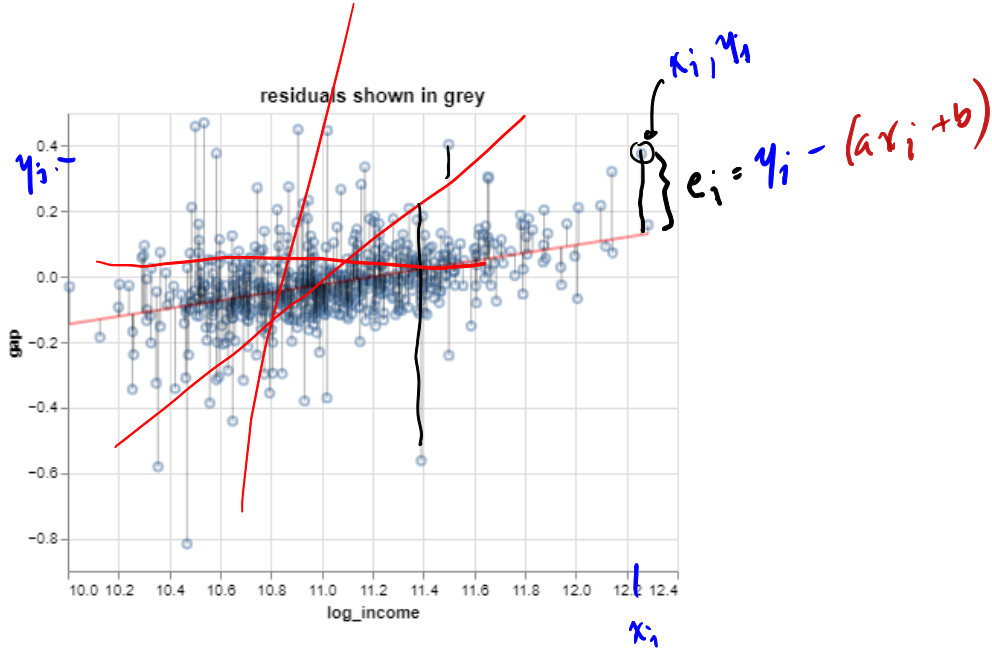
But we can describe any dataset as a line and a 'residual':

$$y_i = \underbrace{ax_i + b}_{\text{line}} \underbrace{+ e_i}_{\text{residual}}$$

Lines and data

Here's a picture:





Each *residual* is simply the vertical distance of a value of Y from the line:

$$e_i = y_i - (ax_i + b)$$

Many possible lines

This makes it possible to express Y as a linear function of X!

However, the mathematical description is somewhat tautological, since for *any* a, b, there are residuals $e_1, \ldots, e_n$ such that

$$y_i = ax_i + b + e_i$$

In other words, there are **infinitely many possible lines**.

So, which values of a and b should be chosen for a given set of data values?

The least squares line

A sensible criterion is to find the line for which:

- the average residual $\bar{e}$ is zero; and
- the residuals vary the least.

If $\bar{e} = 0$, then the residual variance is:

$$\frac{1}{n-1}\sum_{i=1}^{n}(e_i - \bar{e})^2 = \frac{1}{n-1}\sum_{i=1}^{n} e_i^2$$

So the values of a, b that minimize the *sum of squared residuals* give the 'best' line (in one sense of the word 'best'):

$$(a^*, b^*) = \underbrace{\arg\min}_{(a,b)} \left\{ \sum_{i=1}^{n} \underbrace{\big(y_i - (ax_i + b)\big)^2}_{e_i^2} \right\}$$

↑ ↑ minimizers

Calculating the least squares line

The least squares solution (a^*, b^*) has a unique closed form. The line can be written in matrix form as $\mathbf{y} = \mathbf{X}\mathbf{a} + \mathbf{e}$, where:

$$\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}}_{\mathbf{X}} \underbrace{\begin{bmatrix} a \\ b \end{bmatrix}}_{\mathbf{a}} + \underbrace{\begin{bmatrix} e_1 \\ \vdots \\ e_n \end{bmatrix}}_{\mathbf{e}} = \begin{bmatrix} b + ax_1 + e_1 \\ b + ax_2 + e_2 \\ \vdots \\ b + ax_n + e_n \end{bmatrix}$$

Then, the sum of squared residuals is:

$$\mathbf{e}'\mathbf{e} = (\mathbf{y} - \mathbf{X}\mathbf{a})'(\mathbf{y} - \mathbf{X}\mathbf{a})$$

Using vector calculus, one can show that:

$$\nabla_{\mathbf{a}} \mathbf{e}'\mathbf{e} = 0 \implies 2\mathbf{X}'\mathbf{y} = 2\mathbf{X}'\mathbf{X}\mathbf{a} \implies \boxed{\mathbf{a} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}}$$

derivatives

And that this is a minimum.

The solution $\mathbf{a} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$ is known as the **ordinarly least squares** (OLS) solution.

Computation

Luckily, most software efficiently computes OLS solutions.

In [4]:
```
# save explanatory variable and response variable separately as arrays
x = regdata.log_income.values[:, np.newaxis]
y = regdata.gap.values

# configure regression module
slr = LinearRegression()

# fit slr model
slr.fit(x, y)

# store estimates
slope, intercept = slr.coef_, slr.intercept_

ols = np.append(intercept, slope)
ols
```

Out[4]:
```
array([-1.35616996,  0.12105696])
```

We can check the calculations by computing the closed-form expression:

In [5]:
```
# ols solution, by hand
x_mx = np.vstack([np.repeat(1, len(x)), x[:, 0]]).transpose() # X
xtx = x_mx.transpose().dot(x_mx) # X'X
xtx_inv = np.linalg.inv(xtx) # (X'X)^{-1}
xtx_inv.dot(x_mx.transpose()).dot(y) # (X'X)^{-1} X'y
```

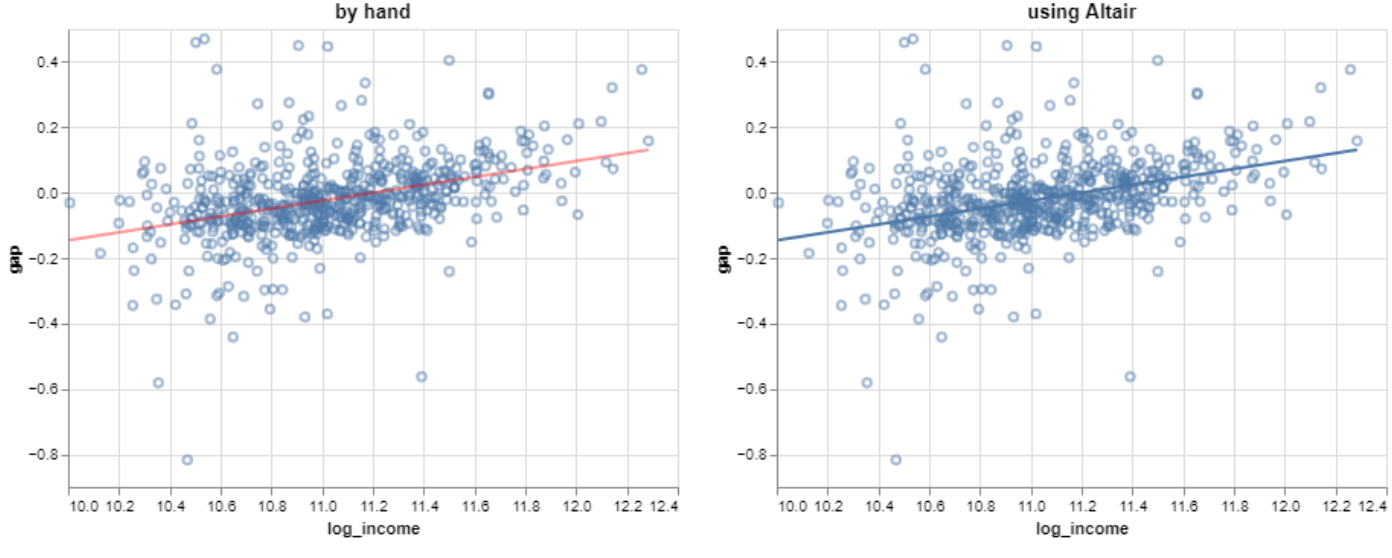
Out[5]:
```
array([-1.35616996,  0.12105696])
```

Plotting

The OLS solution can then be used to connect a grid of evenly-spaced points for plotting the line.

In [6]:
```
# grid of values for plotting
line_df = pd.DataFrame({'log_income': np.linspace(x.min(), x.max(), 500)})
line_df['gap'] = line_df.log_income*slope + intercept
```

Compare the result of layering a line plot of `line_df` on the data scatter (left) with the regression transform from Altair (right):



OLS is not a statistical model (yet)

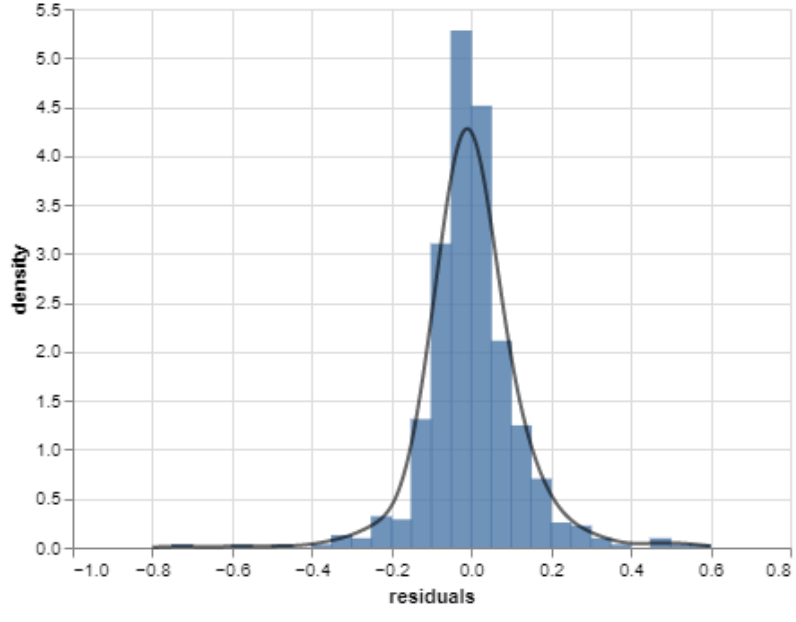
The least squares line is simply an algebraic transformation of the data.

This is not yet a statistical model, since there is no probability distribution involved.

We can change that by considering the residuals to be random.

Residual distribution

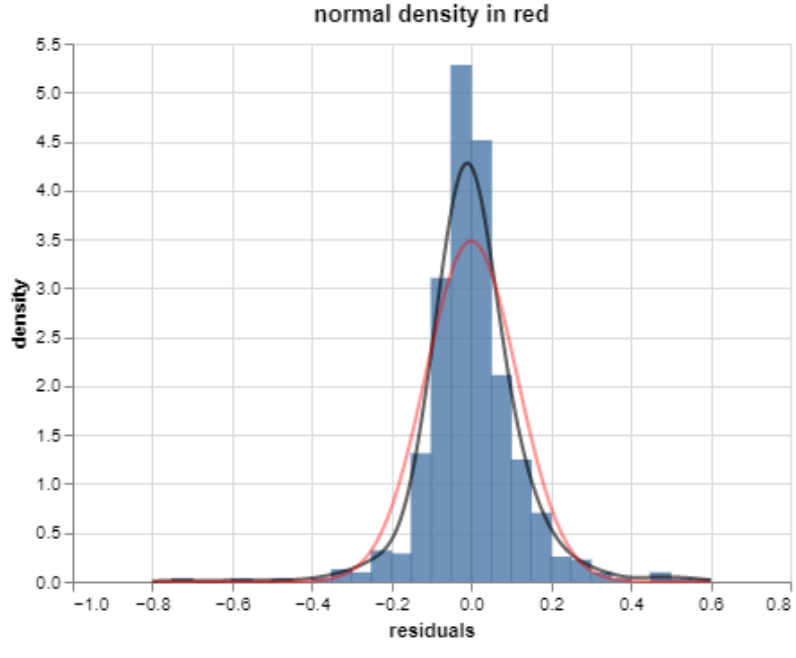
Have a look at the histogram of the residuals (with a KDE curve):



Does this look like any probability density function you encountered in 120A?

Residual distribution

The residual distribution is pretty well-approximated by the *normal* or *Gaussian* distribution:



The error model

This phenomenon -- nearly normal residuals -- is pretty robust.

So a standard **error model** for the residuals is that they are independent normal random variables. This is written as:

$$e_i \overset{iid}{\sim} N\left(0, \sigma^2\right)$$

This is an important modification because it induces a probability distribution on y_i.

In other words, it makes the linear description of Y into a statistical model.

The simple linear model

Now we're in a position to state the **simple linear model**:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \quad \begin{cases} i = 1, \ldots, n \\ \epsilon_i \sim N\left(0, \sigma^2\right) \end{cases}$$

$\Rightarrow \mathbb{E}\,\epsilon_i = 0$

$\text{var}\,\epsilon_i = \sigma^2$

Terminology

- y_i is the ***response variable***
- x_i is the ***explanatory variable***
- ϵ_i is the ***error***
- $\beta_0, \beta_1, \sigma^2$ are the ***model parameters***
    - β_0 is the ***intercept***
    - β_1 is the ***coefficient***
    - σ^2 is the ***error variance***

Model implications

Treating the error term as random has a number of implications.

- (Normality) The response is a normal random variable: $y_i \sim N\left(\beta_0 + \beta_1 x_i, \sigma^2\right)$
- (Linearity) The mean response is linear in X: $\mathbb{E}y_i = \beta_0 + \beta_1 x_i$
- (Constant variance) The response has variance: $\mathrm{var} y_i = \sigma^2$
- (Independence) The observations are independent (because the errors are): $y_i \perp y_j$

These are the **assumptions** of the linear model.

Aside: other error distributions, or conditions that don't assume a specific distribution, are possible.

$$\mathbb{E} y_i = \mathbb{E}\left[\beta_0 + \beta_1 x_i + \varepsilon_i\right]$$
$$= \mathbb{E}\beta_0 + \mathbb{E}\beta_1 x_i + \mathbb{E}\varepsilon_i \quad (\beta_0 + \beta_1 x_i \text{ not random}; \ \mathbb{E}\varepsilon_i = 0)$$
$$= \beta_0 + \beta_1 x_i$$

$$\mathrm{var}\, y_i = \mathrm{var}\left[\beta_0 + \beta_1 x_i + \varepsilon_i\right]$$
$$= \mathrm{var}\,\varepsilon_i$$
$$= \sigma^2$$

Estimates

The OLS solution has a number of optimality properties with respect to the simple linear model -- in other words, it's the best estimate of the parameters β_0, β_1 under many conditions.

You've already seen how to compute the OLS estimates. These are typically denoted by the corresponding paramater with a hat:

$$\hat{\beta} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{bmatrix} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

An estimate of the error variance is:

$$\hat{\sigma}^2 = \frac{1}{n-2}\sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i\right)^2 = \frac{1}{n-2}\left(\mathbf{y} - \mathbf{X}\hat{\beta}\right)'\left(\mathbf{y} - \mathbf{X}\hat{\beta}\right)$$

Fitted values and residuals

Once estimates are computed, the projections of the data points onto the line are known as **fitted values**: they are ***the estimated values of the response variable for each data point***.

These are typically denoted $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ and computed as:

In [7]:
```
# fitted values
fitted = slr.predict(x)
```

array of explanatory variable values
(remember: slr.fit(x, y) ; same x)

The **model residuals** are then ***the difference between observed and fitted values***: $y_i - \hat{y}_i$

.

Parameter interpretations

Let's go back to the SEDA example. The parameter estimates were:

In [8]:
```
ols
```

Out[8]:
```
array([-1.35616996,  0.12105696])
```

Since $\mathbb{E}y_i = \beta_0 + \beta_1 x_i$, these are interpreted as follows.

- (Intercept) When median district income is 1 dollar ($x_i = 0$), the mean achievement gap ($\mathbb{E}y_i$) is estimated to be 1.356 standard deviations of the national average in favor of girls.
    - (Not of particular interest here because no districts have a median income of 1 USD.)
- (Slope) Every doubling of median income is associated with an estimated increase in the mean achievement gap of 0.084 standard deviations of the national average in favor of boys.
    - $\hat{\beta}_1 \log(2x) = \hat{\beta}_1 \log x + \hat{\beta}_1 \log 2$

In [9]:
```
ols[1]*np.log(2)
```

Out[9]:
```
0.08391029103902604
```

General parameter interpretations

There is some general language for interpreting the parameter estimates:

- (Intercept) [at $x_i = 0$] the mean [response variable] is estimated to be [$\hat{\beta}_0$ units].
- (Slope) Every [one-unit increase in x_i] is associated with an estimated change in mean [response variable] of [$\hat{\beta}_1$ units].

You can use this standard language as a formulaic template for interpreting estimated parameters.

Uncertainty quantification

A great benefit of the simple linear model relative to a best-fit line is that the error variance allows for *uncertainty quantification*.

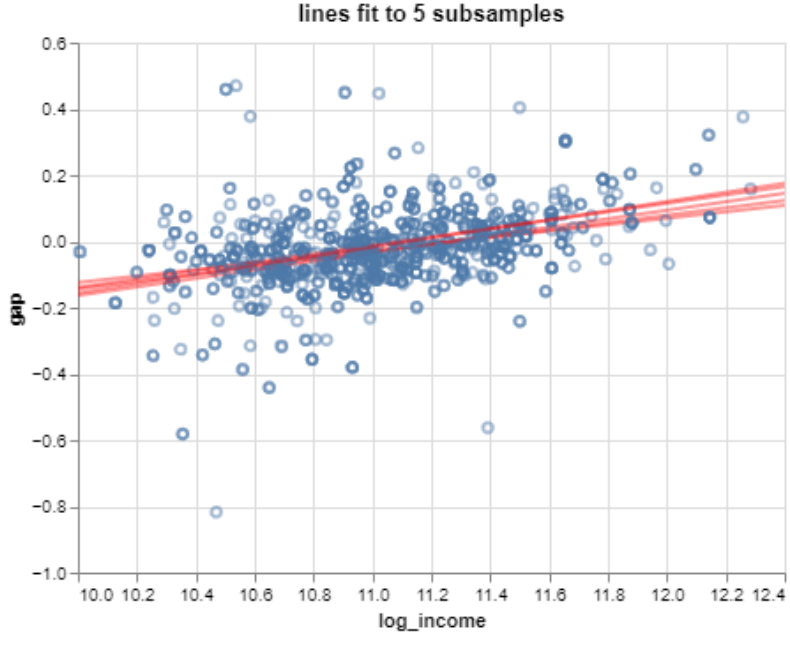
What that means is that one can describe precisely:

- variation in the estimates (*i.e.*, estimated model reliability);
- variation in predictions made using the estimated model (*i.e.*, predictive reliability).

Understanding variation in estimates

What would happen to the estimates if they were computed from a different sample?

We can explore this idea a little by calculating OLS estimates from *subsamples* of the dataset.



The lines are pretty similar, but they change a bit from subsample to subsample.

So, a useful question is: *by how much should one expect the estimates to change depending on the data they are fit to?*

Variance of parameter estimates

Under the simple linear model, the estimated parameters have calculable variances.

It can be shown that:

$$\begin{bmatrix} \operatorname{var}\hat{\beta}_0 & \operatorname{cov}\left(\hat{\beta}_0, \hat{\beta}_1\right) \\ \operatorname{cov}\left(\hat{\beta}_0, \hat{\beta}_1\right) & \operatorname{var}\hat{\beta}_1 \end{bmatrix} = \sigma^2 \left(\mathbf{X}'\mathbf{X}\right)^{-1}$$

So the variances can be *estimated* by plugging in $\hat{\sigma}^2$. The estimated standard deviations are known as *standard errors*:

$$\operatorname{SE}(\hat{\beta}_0) = \sqrt{\hat{\sigma}^2 (\mathbf{X}'\mathbf{X})^{-1}_{11}} \qquad \text{and} \qquad \operatorname{SE}(\hat{\beta}_1) = \sqrt{\hat{\sigma}^2 (\mathbf{X}'\mathbf{X})^{-1}_{22}}$$

About 95% of the time, the true values will be within 2SE of any particular estimates.

So was the gap estimate a fluke?

In [10]:

```
# residuals
resid = y - fitted

# residual SE
n = len(x)
p = 2
resid_se = np.sqrt(resid.var()*(n - 1)/(n - p))

# coefficient variances/covariances
x_mx = np.vstack([np.repeat(1, n), x[:, 0]]).transpose()
coef_vcov = np.linalg.inv(x_mx.transpose().dot(x_mx))*(resid_se**2)

# coefficient standard errors
coef_se = np.sqrt(coef_vcov.diagonal())

# coefficient intervals
np.vstack([ols + 2*coef_se, ols - 2*coef_se])
```

Out[10]:

```
array([[-1.09498451,  0.14472448],
       [-1.61735541,  0.09738944]])
```

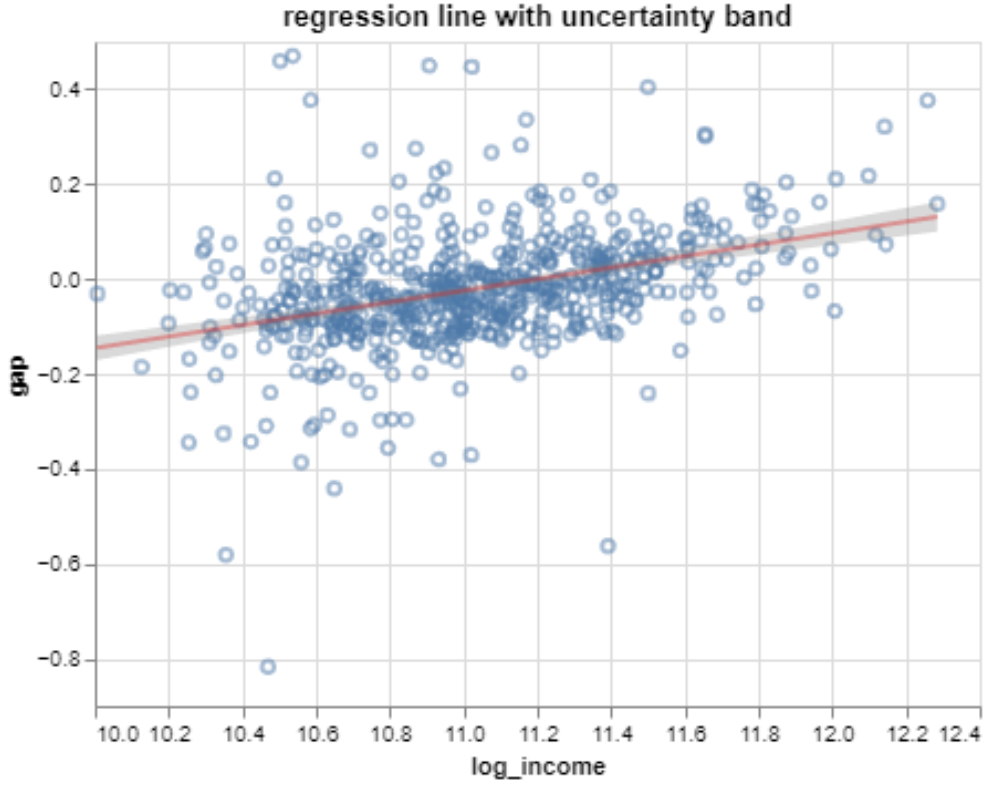
NO! not a fluke.

$\hat{\beta}_{max} = 0.0473 > 0$

Visual display of uncertainty quantification

Often it's easier to get the message across with a plot.

It's fairly common practice to add a *band* around the plotted line to indicate estimated variability.



Prediction

Predictions for a district not in the dataset can be calculated by simply plugging in the explanatory variable for the new observation.

If for example, we'd like to predict the mean achievement gap in math for a district with a median income of 86K, we'd use:

$$x_{new} = \log(86000)$$

And compute:

$$\hat{y}_{new} = \hat{\beta}_0 + \hat{\beta}_1 \log(86000) = \begin{bmatrix} 1 & \log(86000) \end{bmatrix} \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{bmatrix}$$

In [11]:
```
# prediction
newobs = np.array([1, np.log(86000)])
pred = ols.dot(newobs)
pred
```

Out[11]:
```
0.019291648586032384
```

Prediction uncertainty

The variance of a predicted observation is given by:

$$\text{var}(\hat{y}_{new}) = \sigma^2 \left(1 + \mathbf{x}'_{new}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_{new}\right)$$

So the estimated standard deviation of the prediction is:

$$SE\left(\hat{y}_{new}\right) = \sqrt{\hat{\sigma}^2 \left(1 + \mathbf{x}'_{new}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_{new}\right)}$$

In [12]:
```
pred_se = np.sqrt((resid_se**2)*(1 + newobs.dot(xtx_inv).dot(newobs)))
pred_se
```

Out[12]:
```
0.11483333147068375
```

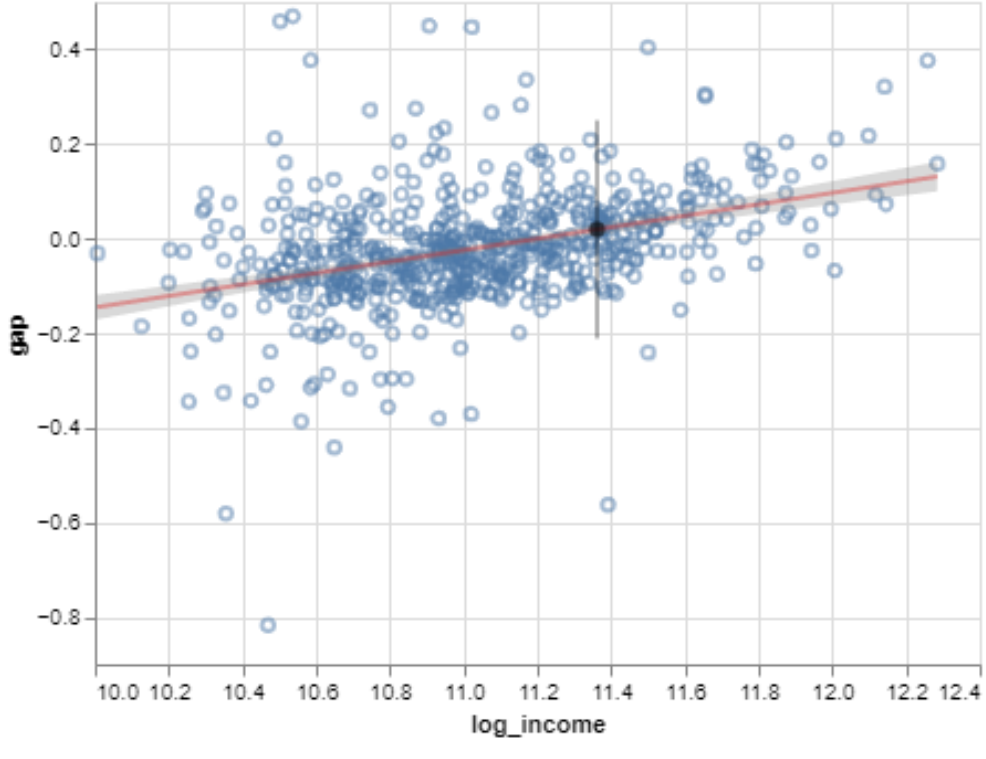
Again, about 95% of the time the true value will be within 2SE of the estimate. So our *uncertainty* about the prediction is:

In [13]:
```
[pred - 2*pred_se, pred + 2*pred_se]
```

Out[13]:
```
[-0.21037501435533512, 0.2489583115273999]
```

Prediction uncertainty

The prediction uncertainty is considerable, but consistent with the variability we see in the data.



Summary

This was our first week on statistical models.

- A statistical model is a probability distribution representing a data-generating process.
- The distributions you know and love from PSTAT120A are all simple models.

Our focus was on the **simple linear model**, according to which ***one variable of interest is a linear function of another variable and a random error***.

- Parameters are estimated by minimizing residual variance (least squares).
- The model assumes normality, linearity, constant variance, and independence.
- Useful for both prediction and inference.
- Estimated variance allows for uncertainty quantification.

Next week we'll discuss extending this model to cases with *multiple* explanatory variables.