Introduction to Data Science: Linear Regression

Héctor Corrada Bravo

University of Maryland, College Park, USA

2020-04-05

Linear Regression

Linear regression is a very elegant, simple, powerful and commonly used technique for data analysis.

We use it extensively in exploratory data analysis and in statistical analyses

Linear Regression

Simple Regression

The goal here is to analyze the relationship between a *continuous numerical* attribute Y and another (*numerical* or *categorical*) variable X.

We assume that in the population, the relationship between the two is given by a linear function:

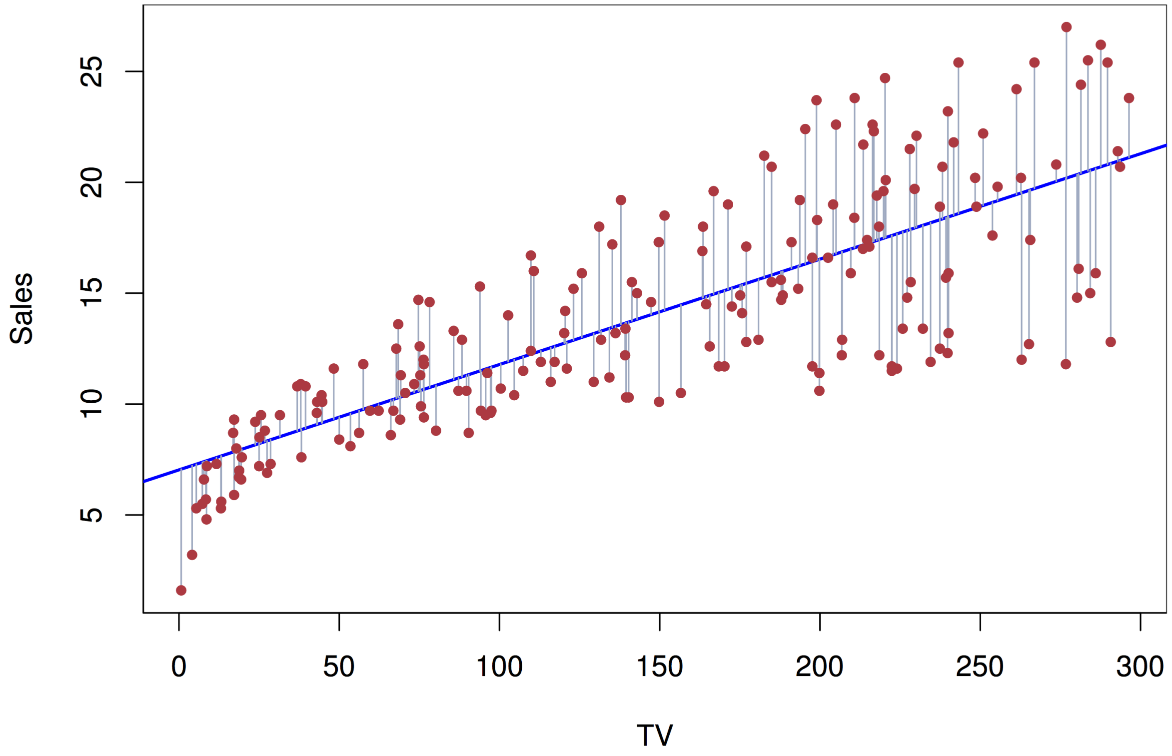
$$Y = \beta_0 + \beta_1 X$$

Linear Regression

Here is (simulated) data from an advertising campaign measuring sales and the amount spent in advertising.

$$\texttt{\textbf{sales}} \approx \beta_0 + \beta_1 \times \texttt{\textbf{TV}}$$

Linear Regression



Linear Regression

We would say that we *regress* `sales` on `TV` when we perform this regression analysis.

As before, given data we would like to estimate what this relationship is in the *population* (what is the population in this case?).

What do we need to estimate in this case? Values for β_0 and β_1. What is the criteria that we use to estimate them?

Linear Regression

We are stating mathematically:

$$\mathbb{E}[Y|X = x] = \beta_0 + \beta_1 x$$

Linear Regression

Given a dataset, the problem is then to find the values of β_0 and β_1 that minimize deviation between data and expectation

Like the estimation of central trend (mean) we use squared devation to do this.

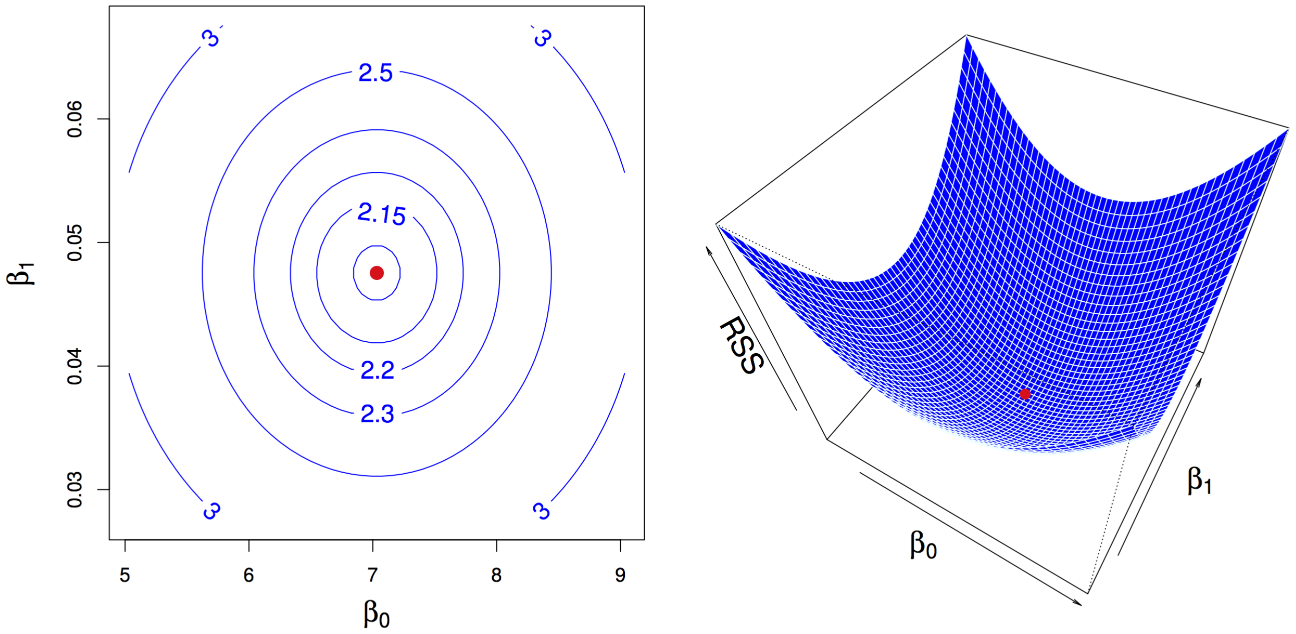
Linear Regression

The linear regression problem

Given data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, find values β_0 and β_1 that minimize *objective* or *loss* function RSS (residual sum of squares):

$$\arg\min_{\beta_0, \beta_1} RSS = \frac{1}{2} \sum_i (y_i - (\beta_0 + \beta_1 x_i))^2$$

Linear Regression



Linear Regression

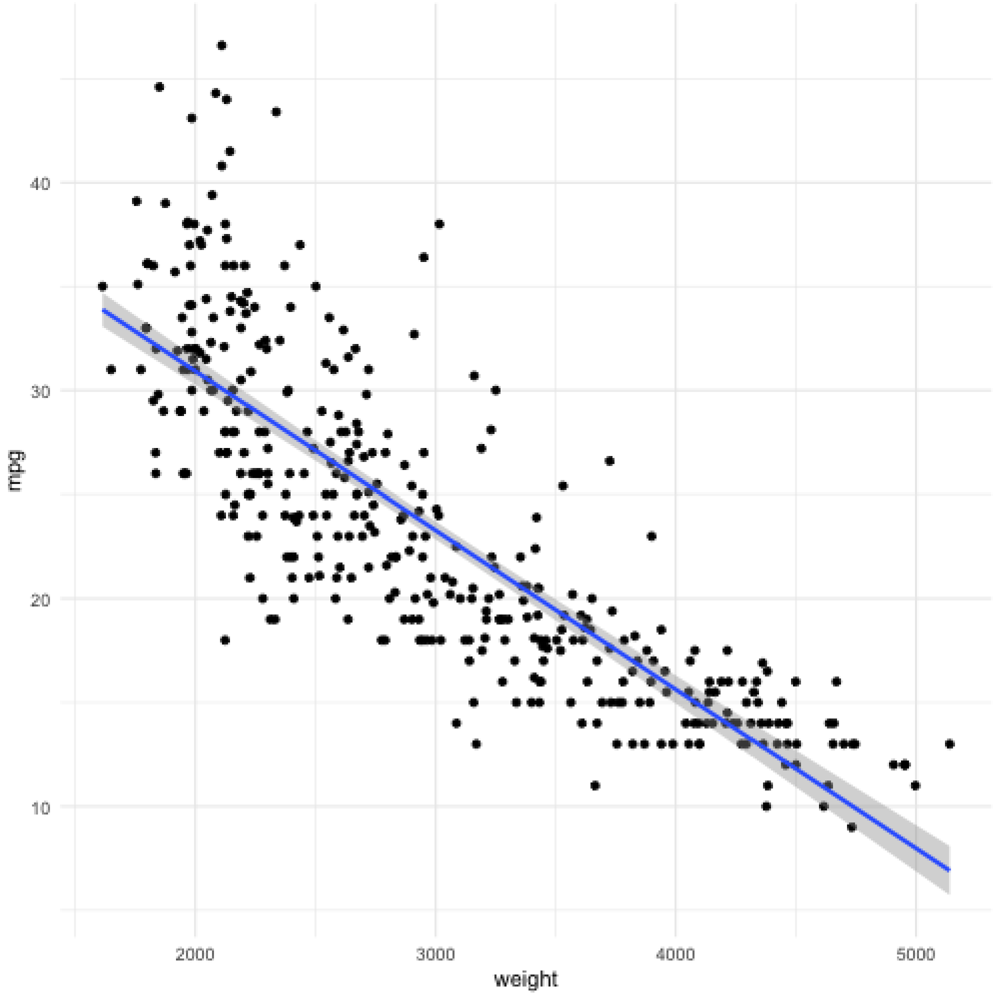
Like derivation of the mean as a measure of central tendency we can derive the values of minimizers $\hat{\beta}_0$ and $\hat{\beta}_1$.

We use the same principle, compute derivatives (partial this time) of the objective function RSS, set to zero and solve.

Linear Regression

$$\begin{aligned}\hat{\beta}_1 &= \frac{\sum_{i=1}^{n}(y_i - \overline{y})(x_i - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2} \\ &= \frac{\mathrm{cov}(y, x)}{\mathrm{var}(x)} \\ \hat{\beta}_0 &= \overline{y} - \hat{\beta}_1\overline{x}\end{aligned}$$

Linear Regression



Linear Regression

In R, linear models are built using the `lm` function

```
auto_fit <- lm(mpg~weight, data=Auto)
auto_fit
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = Auto)
##
## Coefficients:
## (Intercept)       weight
##   46.216525    -0.007647
```

Linear Regression

This states that for this dataset

$$\hat{\beta}_0 = 46.2165245 \; \hat{\beta}_1 = -0.0076473.$$

What's the interpretation?

Linear Regression

According to this model,

a weightless car `weight=0` would run ≈ 46.22 *miles per gallon* on average, and,

on average, a car would run ≈ 0.01 *miles per gallon* fewer for every extra *pound* of weight.

Units of the outcome Y and the predictor X matter for the interpretation of these values.

Linear Regression

Inference

Now that we have an estimate, we want to know its precision.

The main point is to understand that like the sample mean, the regression line we learn from a specific dataset is an estimate.

A different sample from the same population would give us a different estimate (regression line).

Linear Regression

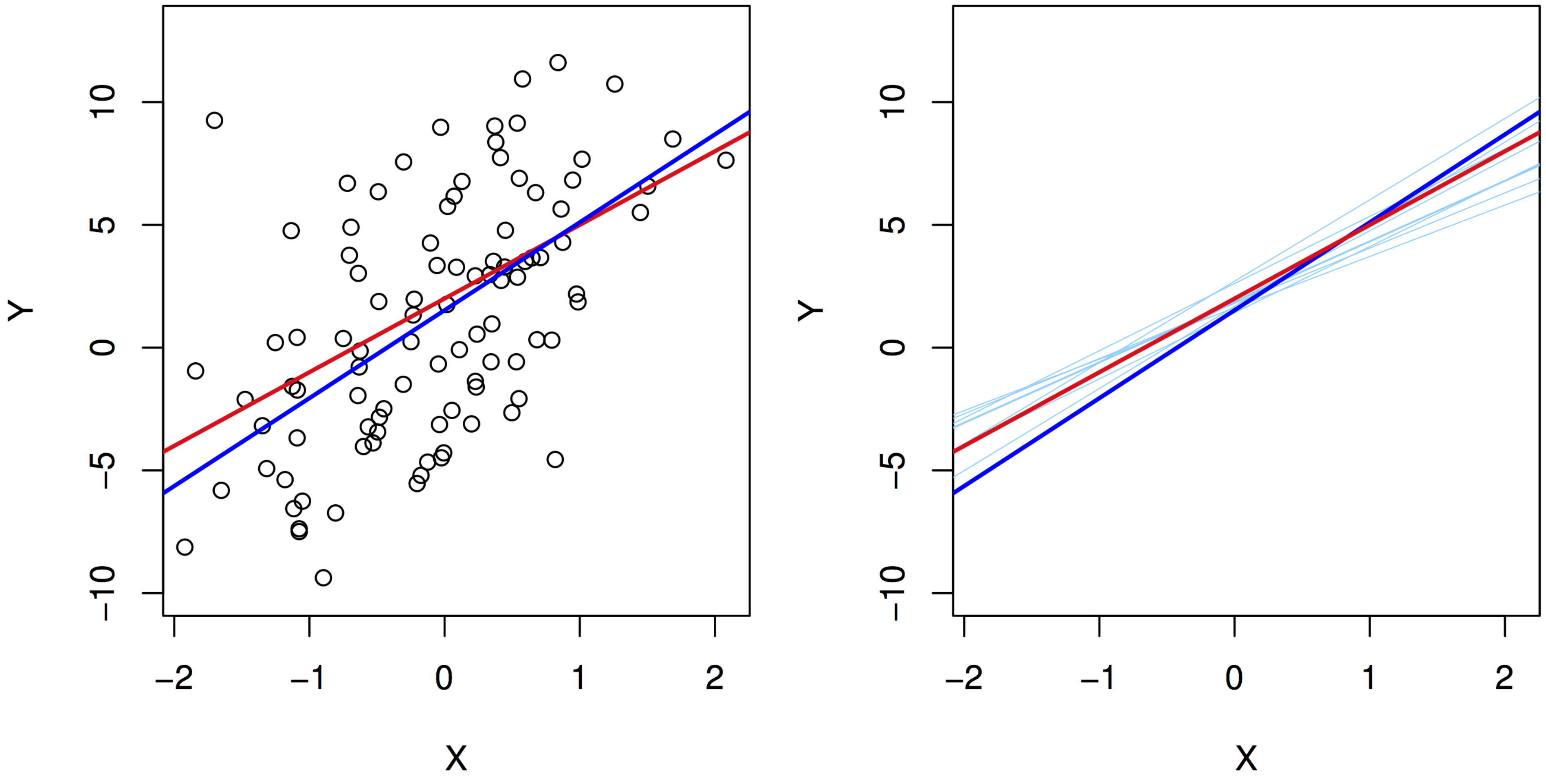
The Central Limit Theorem tells us

on average, we are close to population regression line (I.e., close to β_0 and β_1),

the spread around β_0 and β_1 is well approximated by a normal distribution and

the spread goes to zero as the sample size increases.

Linear Regression



Linear Regression

Confidence Interval

We can construct a confidence interval to say how precise we think our estimates are.

We want to see how precise our estimate of β_1 is, since that captures the relationship between the two variables.

Linear Regression

First, we calculate a standard error estimate for β_1:

$$\mathrm{se}(be\hat{t}a_1)^2 = \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (x_i - \overline{x})^2}$$

Linear Regression

and construct a 95% confidence interval

$$\beta_1 = \hat{\beta}_1 \pm 1.95 \times \mathrm{se}(\hat{beta_1})$$

Linear Regression

Going back to our example:

```
auto_fit_stats <- auto_fit %>%
  tidy() %>%
  select(term, estimate, std.error)
auto_fit_stats
```

```
## # A tibble: 2 x 3
##   term        estimate std.error
##   <chr>          <dbl>     <dbl>
## 1 (Intercept) 46.2      0.799
## 2 weight      -0.00765  0.000258
```

Linear Regression

Given the confidence interval, we would say,

"on average, a car runs ${}_{-0.0082} - 0.0076_{-0.0071}$ *miles per gallon* fewer per pound of weight.

Linear Regression

The t-statistic and the t-distribution

We can also test a null hypothesis about this relationship: "there is no relationship between weight and miles per gallon",

this translates to $\beta_1 = 0$.

Linear Regression

Again, using the same argument based on the CLT, if this hypothesis is true then the distribution of $\hat{\beta}_1$ is well approximated by $N(0, \mathrm{se}(\hat{\beta}_1))$,

if we observe the learned $\hat{\beta}_1$ is *too far* from 0 according to this distribution then we *reject* the hypothesis.

Linear Regression

The CLT states that the normal approximation is good as sample size increases, but what about moderate sample sizes (say, less than 100)?

The t distribution provides a better approximation of the sampling distribution of these estimates for moderate sample sizes, and it tends to the normal distribution as sample size increases.

Linear Regression

The t distribution is commonly used in this testing situation to obtain the probability of rejecting the null hypothesis.

It is based on the t-statistic

$$\frac{\hat{\beta}_1}{\mathrm{se}(\hat{\beta}_1)}$$

Linear Regression

You can think of this as a *signal-to-noise* ratio, or a standardizing transformation on the estimated parameter.

Linear Regression

In our example, we get a t statistic and p-value as follows:

```
auto_fit_stats <- auto_fit %>%
  tidy()
auto_fit_stats
```

```
## # A tibble: 2 x 5
##   term  estimate std.error statistic
##   <chr>    <dbl>     <dbl>     <dbl>
## 1 (Int… 46.2      0.799         57.9
## 2 weig… -0.00765  0.000258     -29.6
## # … with 1 more variable: p.value <dbl>
```

Linear Regression

We would say:

"We found a statistically significant relationship between weight and miles per gallon. On average, a car runs ${}_{-0.0082} - \mathbf{0.0076}_{-0.0071}$ *miles per gallon* fewer per pound of weight (t=-29.65, $p <$ 6.02e-102)."

Linear Regression

Global Fit

We can make *predictions* based on our conditional expectation,

that prediction should be better than a prediction of the outcome with a simple average.

We can use this comparison as a measure of how good of a job we are doing using our model to fit this data: how much of the variance of Y can we *explain* with our model.

Linear Regression

To do this we can calculate *total sum of squares*:

$$TSS = \sum_i (y_i - \overline{y})^2$$

(this is the squared error of a prediction using the sample mean of Y)

Linear Regression

and the *residual sum of squares*:

$$RSS = \sum_i (y_i - \hat{y}_i)^2$$

(which is the squared error of a prediction using the linear model we learned)

Linear Regression

The commonly used R^2 measure compares these two quantities:

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

Linear Regression

These types of global statistics for the linear model can be obtained using the `glance` function in the `broom` package. In our example

```
auto_fit %>%
  glance() %>%
  select(r.squared, sigma, statistic, df, p.value)
```

```
## # A tibble: 1 x 5
##   r.squared sigma statistic    df   p.value
##       <dbl> <dbl>     <dbl> <int>     <dbl>
## 1     0.693  4.33      879.     2 6.02e-102
```

Linear Regression

Some important technicalities

We mentioned above that predictor X could be *numeric* or *categorical*.

However, this is not precisely true. We use a transformation to represent *categorical* variables.

Linear Regression

Here is a simple example:

Suppose we have a categorical attribute `sex`. We can create a 0-1 dummy variable x as we have seen before.

and fit a model $y = \beta_0 + \beta_1 x$.

Linear Regression

What is the conditional expectation given by this model?

If the person is male, then $y = \beta_0$, if the person is female, then $y = \beta_0 + \beta_1$.

So, what is the interpretation of β_1?

Linear Regression

What is the conditional expectation given by this model?

If the person is male, then $y = \beta_0$, if the person is female, then $y = \beta_0 + \beta_1$.

So, what is the interpretation of β_1?

The average difference in credit card balance between females and males.

Linear Regression

We could do a +1/-1 different encoding as well.

Then what is the interpretation of β_1 in this case?

Linear Regression

Note, that when we call the `lm(y~x)` function and `x` is a factor with two levels, the first transformation is used by default.

What if there are more than 2 levels? We need multiple regression, which we will see shortly.

Issues with linear regression

There are some assumptions underlying the inferences and predictions we make using linear regression

We should verify are met when we use this framework.

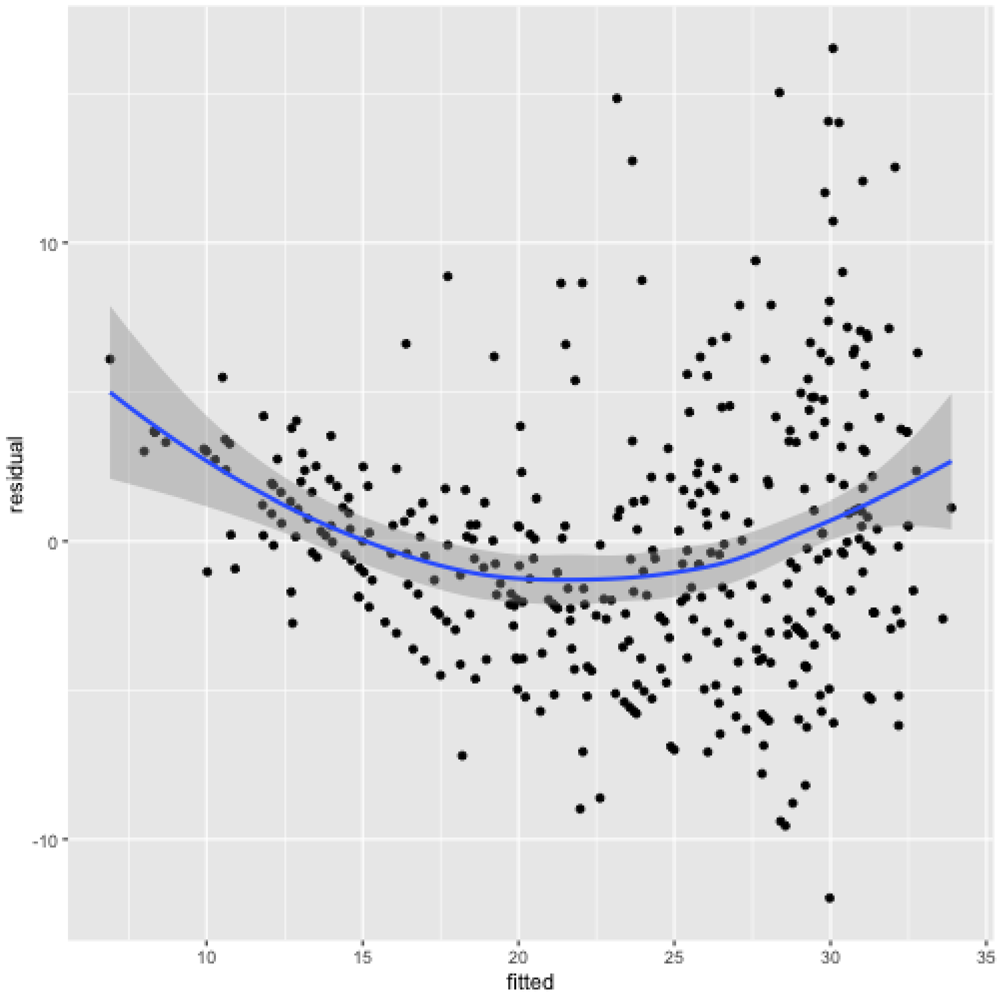
Issues with linear regression

Non-linearity of outcome-predictor relationship

What if the underlying relationship is not linear?

We can use exploratory visual analysis to do this for now by plotting residuals $(y_i - \hat{y}_i)^2$ as a function of the fitted values $\hat{y}_i$.

Issues with linear regression



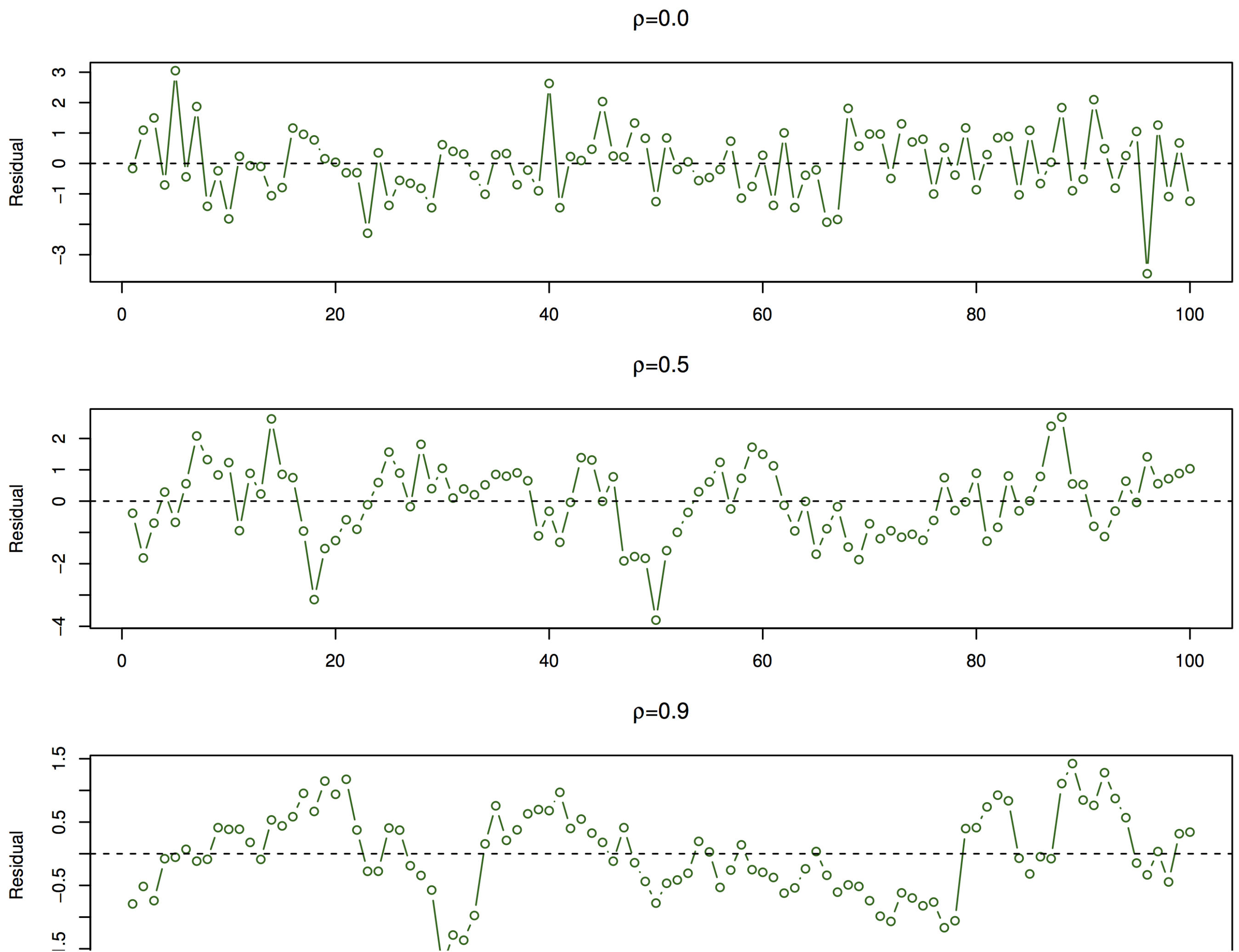
Issues with linear regression

Correlated Error

For our inferences to be valid, we need residuals to be independent and identically distributed.

We can spot non independence if we observe a trend in residuals as a function of the predictor X.

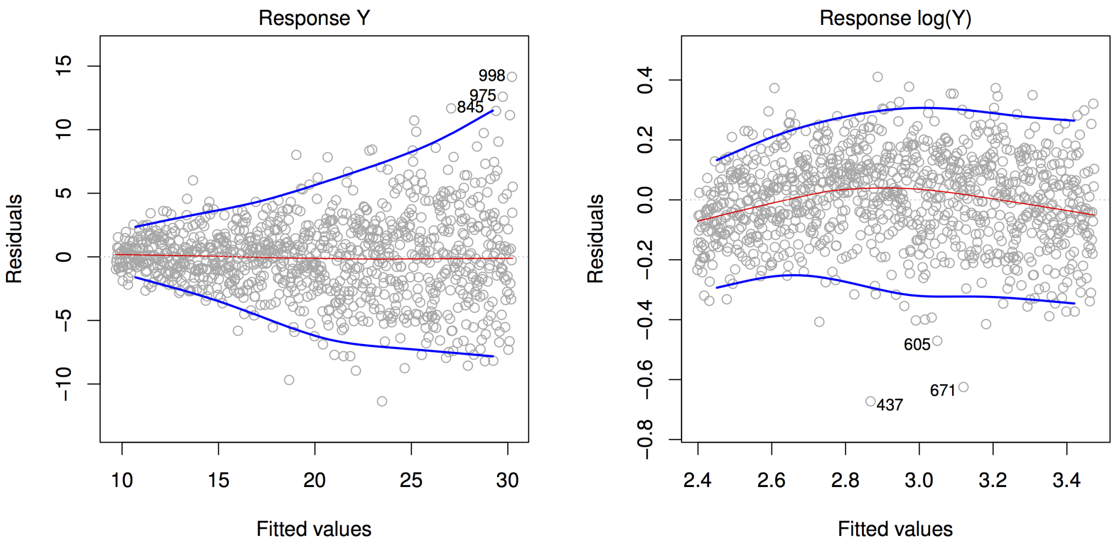
Issues with linear regression



Issues with linear regression

Non-constant variance

Here is an illustration, and a possible fix using a log transformation on the outcome Y.



Multiple linear regression

In this case, we use models of conditional expectation represented as linear functions of multiple variables:

$$\mathbb{E}[Y|X_1 = x_1, X_2 = x_2, \ldots, X_p = x_p] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_3 x_3$$

Multiple linear regression

In the case of our advertising example, this would be a model:

$$\texttt{sales} = \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{newspaper} + \beta_3 \times \texttt{facebook}$$

Multiple linear regression

These models let us make statements of the type:

"holding everything else constant, sales increased on average by 1000 per dollar spent on Facebook advertising" (this would be given by parameter β_3 in the example model).

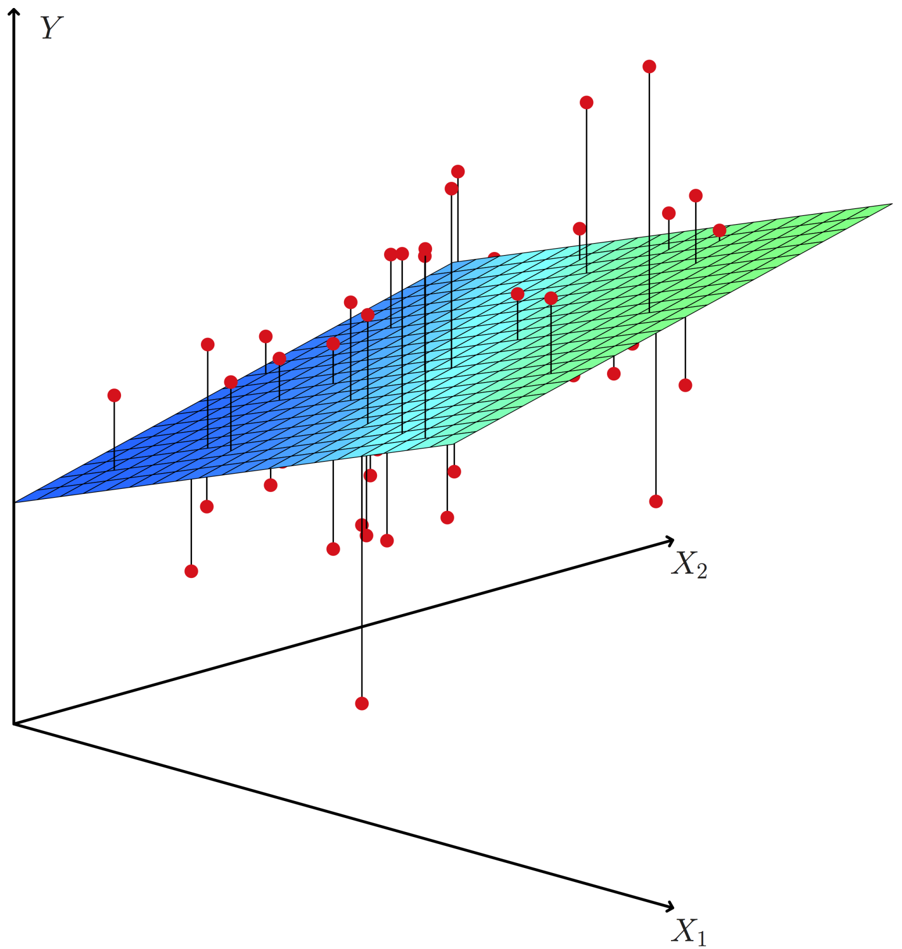
Multiple linear regression

Estimation in multivariate regression

Generalizing simple regression, we estimate β's by minimizing an objective function that represents the difference between observed data and our expectation based on the linear model:

$$
\begin{aligned}
RSS &= \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p))^2
\end{aligned}
$$

Multiple linear regression



Multiple linear regression

The minimizer is found using numerical algorithms to solve this type of *least squares* problems.

Later in the course we will look at *stochastic gradient descent*, a simple algorithm that scales to very large datasets.

Multiple linear regression

Example (cont'd)

```
auto_fit <- lm(mpg~1+weight+cylinders+horsepower+displacement+year, data=Auto)

auto_fit
```

Multiple linear regression

```
##
## Call:
## lm(formula = mpg ~ 1 + weight + cylinders + horsepower + displacement +
##     year, data = Auto)
##
## Coefficients:
##  (Intercept)        weight     cylinders
##   -12.779493     -0.006524     -0.343690
##   horsepower  displacement          year
##    -0.007715      0.006996      0.749924
```

Multiple linear regression

From this model we can make the statement:

"Holding everything else constant, cars run 0.76 miles per gallon more each year on average".

Multiple linear regression

Statistical statements (cont'd)

Like simple linear regression, we can construct confidence intervals, and test a null hypothesis of no relationship ($\beta_j = 0$) for the parameter corresponding to each predictor.

Multiple linear regression

This is again nicely managed by the `broom` package:

```
auto_fit_stats <- auto_fit %>%
  tidy()
auto_fit_stats
```

```
## # A tibble: 6 x 5
##   term  estimate std.error statistic
##   <chr>    <dbl>     <dbl>     <dbl>
## 1 (Int… -1.28e+1  4.27        -2.99
## 2 weig… -6.52e-3  0.000587   -11.1
## 3 cyli… -3.44e-1  0.332       -1.04
```

Multiple linear regression

In this case we would reject the null hypothesis of no relationship only for predictors `weight` and `year`.

We would write the statement for year as follows:

"Holding everything else constant, cars run ${}_{0.65}0.75_{0.85}$ miles per gallon more each year on average (P<1e-16)".

Multiple linear regression

The F-test

We can make additional statements for multivariate regression:

"is there a relationship between *any* of the predictors and the response?".

Mathematically, we write this as $\beta_1 = \beta_2 = \cdots = \beta_p = 0$.

Multiple linear regression

As before, we can compare total outcome variance the residual sum of squared error RSS using the F statistic:

$$\frac{(\mathrm{TSS} - \mathrm{RSS})/p}{\mathrm{RSS}/(n-p-1)}$$

Multiple linear regression

Back to our example, we use the `glance` function to compute this type of summary:

```
auto_fit %>%
  glance() %>%
  select(r.squared, sigma, statistic, df, p.value) %>%
  knitr::kable("html")
```

r.squared	sigma	statistic	df	p.value
0.8089093	3.433902	326.7965	6	0

Multiple linear regression

In comparison with the linear model only using `weight`, this multivariate model explains *more of the variance* of `mpg`, but using more predictors.

This is where the notion of *degrees of freedom* comes in: we now have a model with expanded *representational* ability.

Multiple linear regression

The bigger the model, we are conditioning more and more,

given a fixed dataset, have fewer data points to estimate conditional expectation for each value of the predictors.

estimated conditional expectation is less *precise*.

Multiple linear regression

To capture this phenomenon, we want statistics that tradeoff how well the model fits the data, and the "complexity" of the model.

Multiple linear regression

Now, we can look at the full output of the `glance` function:

```
auto_fit %>%
  glance() %>%
  knitr::kable("html")
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	
0.8089093	0.806434	3.433902	326.7965	0	6	-1036.81	2087.62	211

Multiple linear regression

Columns `AIC` and `BIC` display statistics that penalize model fit with model size.

The smaller this value, the better.

Let's now compare a model only using `weight`, a model only using `weight` and `year` and the full multiple regression model we saw before.

Multiple linear regression

```r
lm(mpg~weight, data=Auto) %>%
  glance() %>%
  knitr::kable("html")
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC
0.6926304	0.6918423	4.332712	878.8309	0	2	-1129.969	2265.939

Multiple linear regression

```r
lm(mpg~weight+year, data=Auto) %>%
  glance() %>%
  knitr::kable("html")
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	
0.8081803	0.8071941	3.427153	819.473	0	3	-1037.556	2083.113	20

Multiple linear regression

In this case, using more predictors beyond `weight` and `year` doesn't help.

Multiple linear regression

Categorical predictors (cont'd)

We saw transformations for categorical predictors with only two values.

In our example we have the `origin` predictor, corresponding to where the car was manufactured, which has multiple values

```
Auto <- Auto %>%
  mutate(origin=factor(origin))
levels(Auto$origin)
```

```
## [1] "1" "2" "3"
```

Multiple linear regression

The `lm` function in R does this transformation by default when a variable has class `factor`.

We can see what the underlying numerical predictors look like by using the `model_matrix` function and passing it the model formula we build:

```
##   (Intercept) origin2 origin3 origin
## 1           1       0       0      1
## 2           1       0       0      1
## 3           1       0       0      1
## 4           1       0       0      1
## 5           1       0       0      1
```

Multiple linear regression

```
##   (Intercept) origin2 origin3 origin
## 1           1       1       0      2
## 2           1       1       0      2
## 3           1       1       0      2
## 4           1       1       0      2
## 5           1       1       0      2
## 6           1       1       0      2
```

Multiple linear regression

```
##   (Intercept) origin2 origin3 origin
## 1           1       0       1      3
## 2           1       0       1      3
## 3           1       0       1      3
## 4           1       0       1      3
## 5           1       0       1      3
## 6           1       0       1      3
```

Interactions in linear models

The linear models so far include *additive* terms for a single predictor.

That let us made statemnts of the type "holding everything else constant...".

But what if we think that a pair of predictors *together* have a relationship with the outcome.

Interactions in linear models

We can add these *interaction* terms to our linear models as products

$$\mathbb{E}Y|X_1 = x_1, X_2 = x2 = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2$$

Interactions in linear models

Consider the advertising example:

$$\texttt{sales} = \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{facebook} + \beta_3 \times (\texttt{TV} \times \texttt{facebook})$$

If β_3 is positive, then the effect of increasing TV advertising money is increased if facebook advertising is also increased.

Interactions in linear models

When using categorical variables, interactions have an elegant interpretation.

Consider our car example, and suppose we build a model with an interaction between `weight` and `origin`.

Interactions in linear models

Let's look at what the numerical predictors look like:

```
##   (Intercept) weight origin2 origin3
## 1           1   3504       0       0
## 2           1   3693       0       0
## 3           1   3436       0       0
## 4           1   3433       0       0
## 5           1   3449       0       0
## 6           1   4341       0       0
##   weight:origin2 weight:origin3 origin
## 1              0              0      1
## 2              0              0      1
```

Interactions in linear models

```
##   (Intercept) weight origin2 origin3
## 1           1   1835       1       0
## 2           1   2672       1       0
## 3           1   2430       1       0
## 4           1   2375       1       0
## 5           1   2234       1       0
## 6           1   2123       1       0
##   weight:origin2 weight:origin3 origin
## 1           1835              0      2
## 2           2672              0      2
## 3           2430              0      2
## 4           2375              0      2
```

Interactions in linear models

```
##   (Intercept) weight origin2 origin3
## 1           1   2372       0       1
## 2           1   2130       0       1
## 3           1   2130       0       1
## 4           1   2228       0       1
## 5           1   1773       0       1
## 6           1   1613       0       1
##   weight:origin2 weight:origin3 origin
## 1              0           2372      3
## 2              0           2130      3
## 3              0           2130      3
## 4              0           2228      3
```

Interactions in linear models

So what is the expected miles per gallon for a car with `origin == 1` as a function of weight?

$$\texttt{mpg} = \beta_0 + \beta_1 \times \texttt{weight}$$

Interactions in linear models

Now how about a car with `origin == 2`?

$$\texttt{\textbf{mpg}} = \beta_0 + \beta_1 \times \texttt{\textbf{weight}} + \beta_2 + \beta_4 \times \texttt{\textbf{weight}}$$

Interactions in linear models

Now think of the graphical representation of these lines.

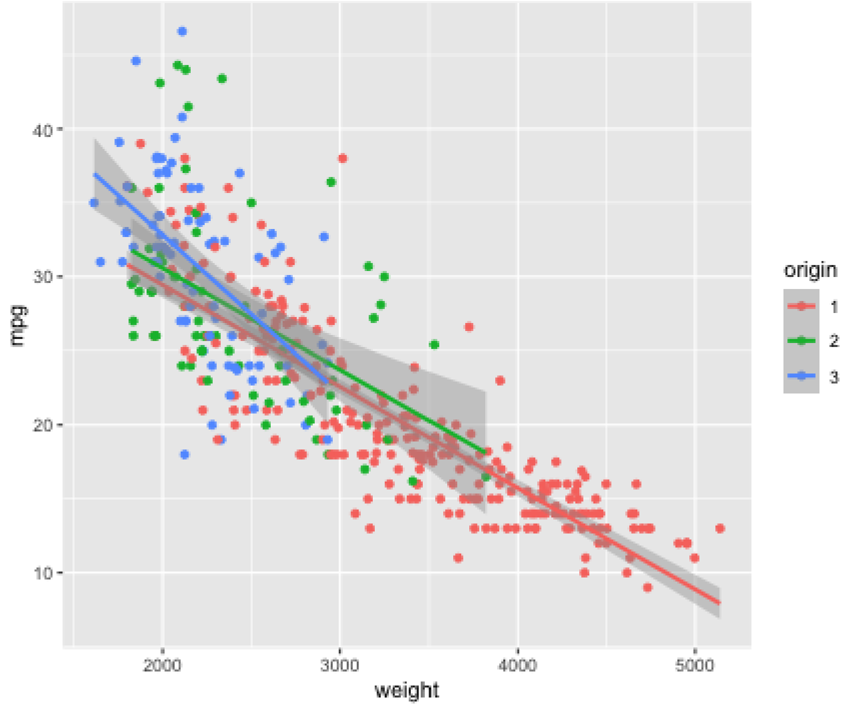
For `origin == 1` the intercept of the regression line is β_0 and its slope is β_1.

For `origin == 2` the intercept of the regression line is $\beta_0 + \beta_2$ and its slope is $\beta_1 + \beta_4$.

Interactions in linear models

`ggplot` does this when we map a factor variable to a aesthetic, say color, and use the `geom_smooth` method:

```
Auto %>%
  ggplot(aes(x=weight, y=mpg, color=origin))
    geom_point() +
    geom_smooth(method=lm)
```



Interactions in linear models

The intercept of the three lines seem to be different, but the slope of `origin == 3` looks different (decreases faster) than the slopes of `origin == 1` and `origin == 2` that look very similar to each other.

Interactions in linear models

Let's fit the model and see how much statistical confidence we can give to those observations:

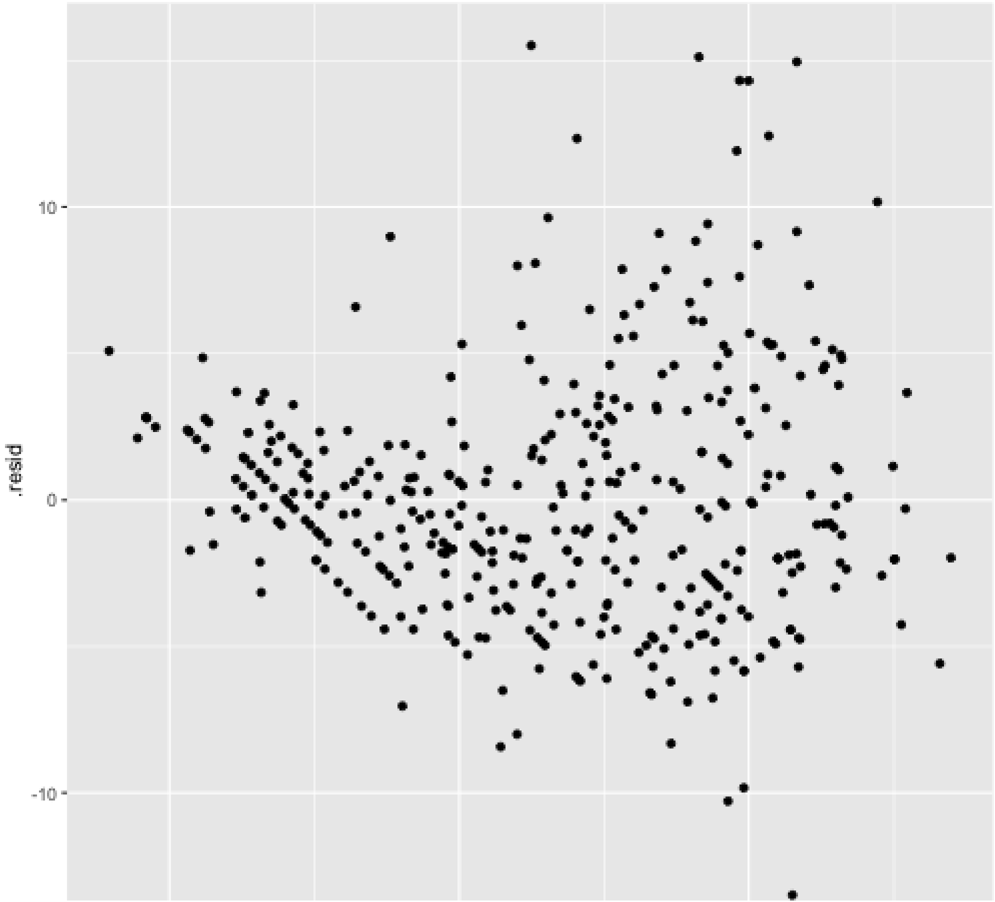
```
## # A tibble: 6 x 5
##   term  estimate std.error statistic
##   <chr>    <dbl>     <dbl>     <dbl>
## 1 (Int…  4.31e+1  1.19      36.4
## 2 weig… -6.85e-3  0.000342 -20.0
## 3 orig…  1.12e+0  2.88       0.391
## 4 orig…  1.11e+1  3.57       3.11
## 5 weig…  3.58e-6  0.00111    0.00322
## 6 weig… -3.87e-3  0.00154   -2.51
```

Interactions in linear models

There is still an issue here because this could be the result of a poor fit from a linear model, it seems none of these lines do a very good job of modeling the data we have.

Interactions in linear models

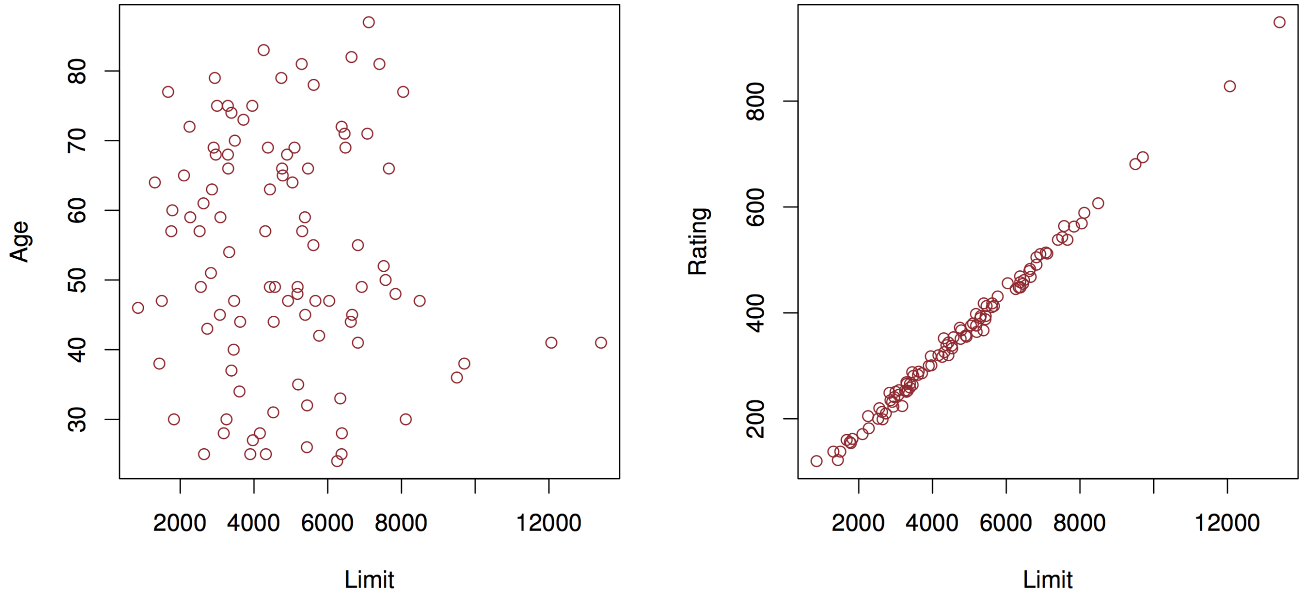
We can again check this for this model:



Additional issues with linear regression

Multiple linear regression introduces an additional issue that is extremely important to consider when interpreting the results of these analyses: collinearity.

Additional issues with linear regression



Additional issues with linear regression

In that case, the set of β's that minimize RSS may not be unique, and therefore our interpretation is invalid.

You can identify this potential problem by regressing predictors onto each other.

The usual solution is to fit models only including one of the colinear variables.

Summary

Flexible, but highly biased method for modeling relationships between variables and deriving predictions for continuous attributes.

We have seen how it is used in the context of EDA and statistical inference.

Saw important caveats to their application.