

# multiple\_polynomial\_regression

September 9, 2019

## 0.1 Multiple regression

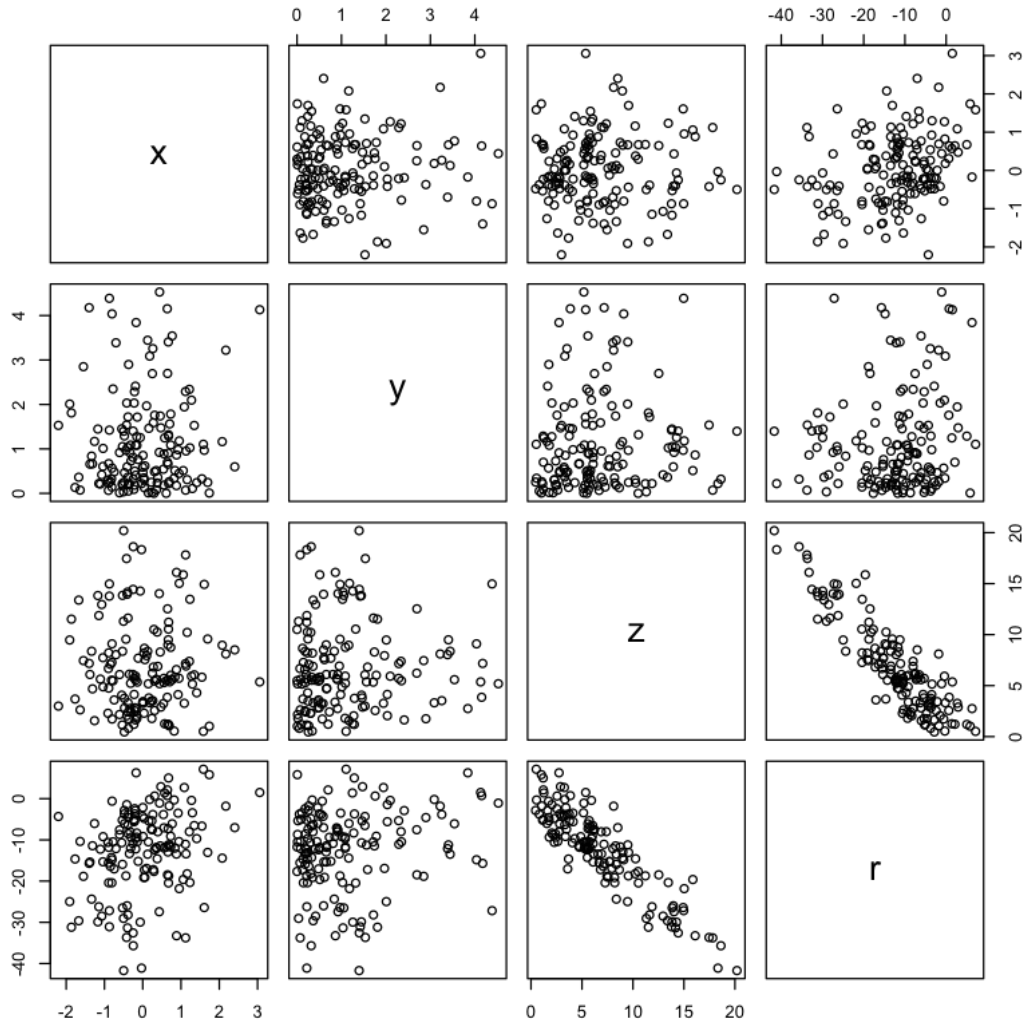
Lets look at a dataset made up of 4 variables:  $x$ ,  $y$ ,  $z$ , and  $r$ .

```
[2]: dta <- read.csv('multipleData.csv')  
      head(dta)
```

	x <dbl>	y <dbl>	z <dbl>	r <dbl>
A data.frame: 6 × 4	-0.9082393	0.30143444	2.287881	-10.61071
	-0.2554406	0.80946626	6.093306	-10.86466
	1.2185192	2.33952550	7.383286	-8.01159
	-1.3996301	4.17670312	7.174199	-15.69763
	0.9467331	0.44573692	5.719590	-10.40901
	-0.2311548	0.08942789	1.201217	-5.03375

Just like simple linear regression, we can gain intuition by plotting all variables in a multiple linear regression to one another.

```
[3]: plot(dta)
```



Sometimes this plot is called a **draftsman plot**. We notice a few interesting relationships \*  $r$  and  $z$  are related negatively. Increasing values of  $z$  correspond to decreasing values of  $r$  \*  $x$  is modestly related to  $r$ ,  $y$ , and  $z$  positively. Increasing values of  $x$  correspond to increasing values of  $y$ ,  $z$ , and  $r$

A **multiple linear regression** (sometimes called a **multivariate regression**) relates changes in  $y$  variable (the right-hand side of the equals sign) to a response variable, or variable to predict, or variable to explain (the left hand side of the equals sign). We can write this down in model form as

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_n x_{in} + \epsilon_i \quad (1)$$

$$\epsilon \sim N(0, \sigma^2), \quad (2)$$

and using matrices and vectors,

$$y = X\beta + \epsilon \quad (3)$$

$$\epsilon \sim N(0, \sigma^2), \quad (4)$$

where

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}; X = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{i,1} & x_{i,2} & \cdots & x_{i,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{M,1} & x_{M,2} & \cdots & x_{M,n} \end{bmatrix}$$

The parameters for each variable ( $\beta$ ) are stacked into a single vector.

The Matrix  $X$  assigns one column per variable, and a column of 1s to represent the intercept.

We can also write the above multiple regression in probabilistic form

$$y \sim N(X\beta, \sigma^2)$$

Notice the same equation is used to represent simple linear regression and multiple linear regression. The differences between simple and multiple regression are folded into  $X$  and  $\beta$ .

## 0.2 Polynomial regression

We've discussed simple and multiple linear regression. Now lets discuss how linear regression can apply to nonlinear relationships.

**Polynomial regression** supposes the functional form relating  $y$  and  $x$  is a polynomial

$$\begin{aligned} y_i &= f(x_i|\beta) \\ &= \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots \beta_n x_i^n + \epsilon_i \\ \epsilon_i &\sim N(0, \sigma^2) \end{aligned}$$

We can rewrite this equation in matrix form

$$\begin{aligned} y_i &= f(x_i|\beta) \\ &= X\beta + \epsilon \\ \epsilon &\sim N(0, \sigma^2) \end{aligned}$$

where

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}; X = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_i & x_i^2 & \cdots & x_i^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_M & x_M^2 & \cdots & x_M^n \end{bmatrix}$$

Here the matrix  $X$  looks slightly different than in multiple regression. The first column of 1s is a place-holder for an intercept. The second column is the linear term  $x_i$ , the next column a quadratic term  $x_i^2$ , next column a cubic term  $x_i^3$  and so on. The same observation  $x$  is raised to higher powers as we move across columns.

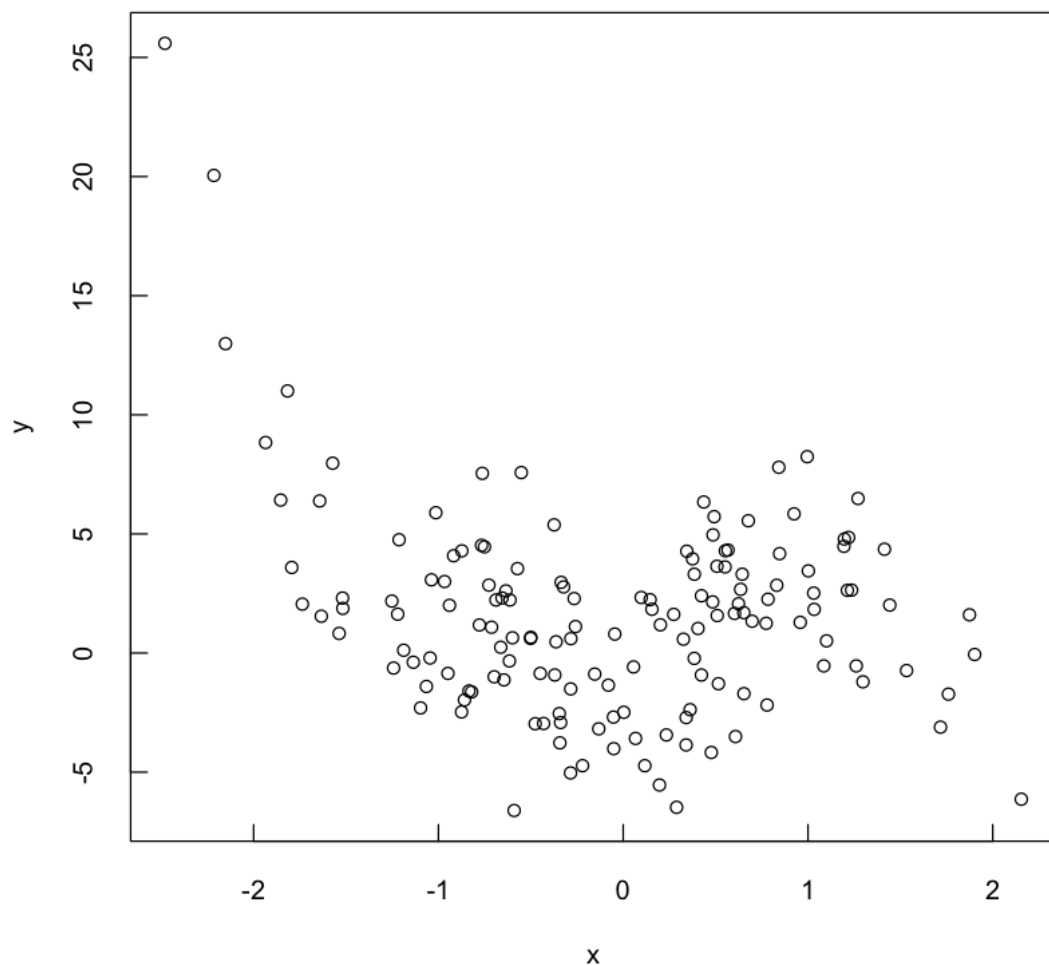
Now lets look at an example.

```
[4]: data <- read.csv('polynomialData.csv')
      head(data)
```

	x	y
	<dbl>	<dbl>
	0.9958723	8.2420054
	-0.6556163	2.3114202
A data.frame: 6 × 2	-0.9176787	4.0842076
	0.1963727	-5.5386897
	1.0309346	2.5166174
	1.2610719	-0.5388713

Here we just have  $x$  values and  $y$  values. The difference, they're not related to each other linearly.

```
[5]: plot(data$x,data$y
          ,xlab="x"
          ,ylab="y"
          ,tck=0.02
          )
```



The variable  $x$  does not relate to  $y$  linearly; fitting a straight line to this data may under-represent the more complicated relationship.

Lets fit three different models: a linear model, a quadratic model, and a cubic model. We can plot each model fit and evaluate which model looks like it best represents the relationship between  $x$  and  $y$ .

```
[6]: simpleLinearRegression <- lm(y~x,data=data)
      print(simpleLinearRegression)

      quadraticRegression <- lm(y~x+I(x^2),data=data)
      print(quadraticRegression)

      cubicRegression <- lm(y~x+I(x^2)+I(x^3),data=data)
```

```
print(cubicRegression)
```

Call:

```
lm(formula = y ~ x, data = data)
```

Coefficients:

(Intercept)	x
1.453	-1.253

Call:

```
lm(formula = y ~ x + I(x^2), data = data)
```

Coefficients:

(Intercept)	x	I(x^2)
-0.08514	-0.81522	1.72492

Call:

```
lm(formula = y ~ x + I(x^2) + I(x^3), data = data)
```

Coefficients:

(Intercept)	x	I(x^2)	I(x^3)
0.2994	2.2877	1.0962	-1.4120

```
[7]: plot(data$x,data$y
      ,xlab="x"
      ,ylab="y"
      ,tck=0.02
      )

minX <- min(data$x)
maxX <- max(data$x)

#linear regression
betas <- coef(simpleLinearRegression)
beta0 <- betas[1]
beta1 <- betas[2]

xVals <- seq(minX,maxX,0.01)
linearYPredictions <- beta0 + beta1*xVals
lines(xVals,linearYPredictions,col='red')

#quadratic regression
betas <- coef(quadraticRegression)
```

```

beta0 <- betas[1]
beta1 <- betas[2]
beta2 <- betas[3]

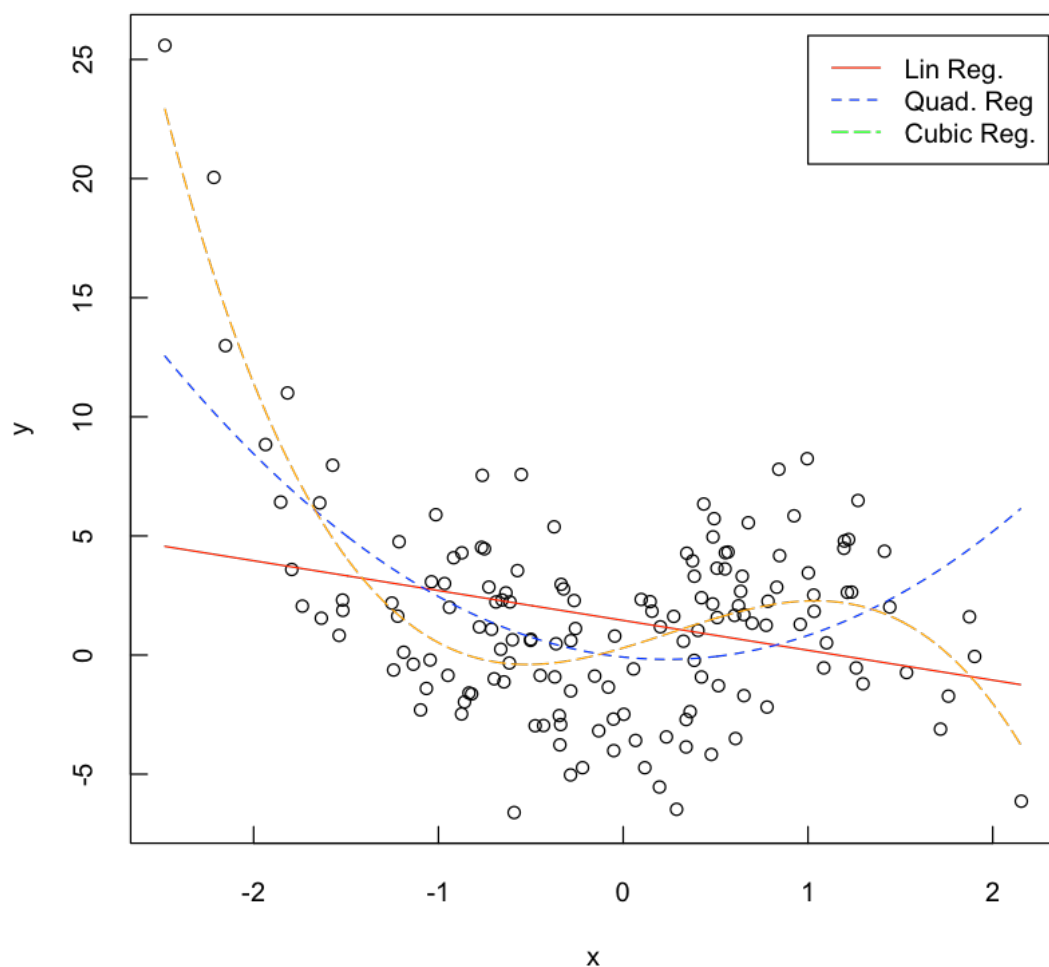
xVals <- seq(minX,maxX,0.01)
quadraticYPredictions <- beta0 + beta1*xVals + beta2*xVals^2
lines(xVals,quadraticYPredictions,col='blue', lty=2)

#cubic regression
betas <- coef(cubicRegression)
beta0 <- betas[1]
beta1 <- betas[2]
beta2 <- betas[3]
beta3 <- betas[4]

xVals <- seq(minX,maxX,0.01)
cubicYPredictions <- beta0 + beta1*xVals + beta2*xVals^2 + beta3*xVals^3
lines(xVals,cubicYPredictions,col='orange',lty=5)

legend(1,26,legend=c("Lin Reg.", "Quad. Reg", "Cubic Reg.")
      ,col=c('red', 'blue', 'green')
      ,lty=c(1,2,5)
)

```



What fit looks best?

### 0.3 Optimization and assessing error

We can look at the three model fits to the data and visually assess fit, but a quantitative method for evaluating model fit is likely more convincing. A quantitative fit allows us to numerically compare model fits.

The most common metric for evaluating model fit is the **sum squares error** (SSE). Given a data set, the **SSE** sums over all observations  $(y_i, x_{i1}, x_{i2}, \dots, x_{im})$  the squared difference between  $(y_i)$  and model predictions  $[\hat{y}_i = f(x_i|\beta)]$



$$\text{SSE}(y, \hat{y}) = \sum_{i=1}^N (y_i - \hat{y}_i)^2,$$

or in vector form,

$$\text{SSE}(y, \hat{y}) = (y - \hat{y})'(y - \hat{y})$$

where

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad (5)$$

and

$$y' = \begin{bmatrix} y'_1 \\ y'_2 \\ \vdots \\ y'_n \end{bmatrix} \quad (6)$$

Intuitively, SSE is a measure of the distance between your data and model. We can compute the SSE for the three above models fit to our non-linear data.

```
[10]: plot(data$x,data$y
          ,xlab="x"
          ,ylab="y"
          ,tck=0.02
        )

minX <- min(data$x)
maxX <- max(data$x)

#linear regression
betas <- coef(simpleLinearRegression)
beta0 <- betas[1]
beta1 <- betas[2]

xVals <- seq(minX,maxX,0.01)
linearYPredictions <- beta0 + beta1*xVals
lines(xVals,linearYPredictions,col='red')

ssr_SLR = sum(((fitted(simpleLinearRegression) - data$y)^2) #NEW
```

```

#quadratic regression
betas <- coef(quadraticRegression)
beta0 <- betas[1]
beta1 <- betas[2]
beta2 <- betas[3]

xVals <- seq(minX,maxX,0.01)
quadraticYPredictions <- beta0 + beta1*xVals + beta2*xVals^2
lines(xVals,quadraticYPredictions,col='blue', lty=2)

ssr_QR = sum((fitted(quadraticRegression) - data$y)^2) #NEW

#cubic regression
betas <- coef(cubicRegression)
beta0 <- betas[1]
beta1 <- betas[2]
beta2 <- betas[3]
beta3 <- betas[4]

xVals <- seq(minX,maxX,0.01)
cubicYPredictions <- beta0 + beta1*xVals + beta2*xVals^2 + beta3*xVals^3
lines(xVals,cubicYPredictions,col='orange',lty=5)

ssr_CR = sum((fitted(cubicRegression) - data$y)^2) #NEW

legend(1,26,legend=c("Lin Reg.", "Quad. Reg", "Cubic Reg.")
      ,col=c('red', 'blue', 'green')
      ,lty=c(1,2,5)
)

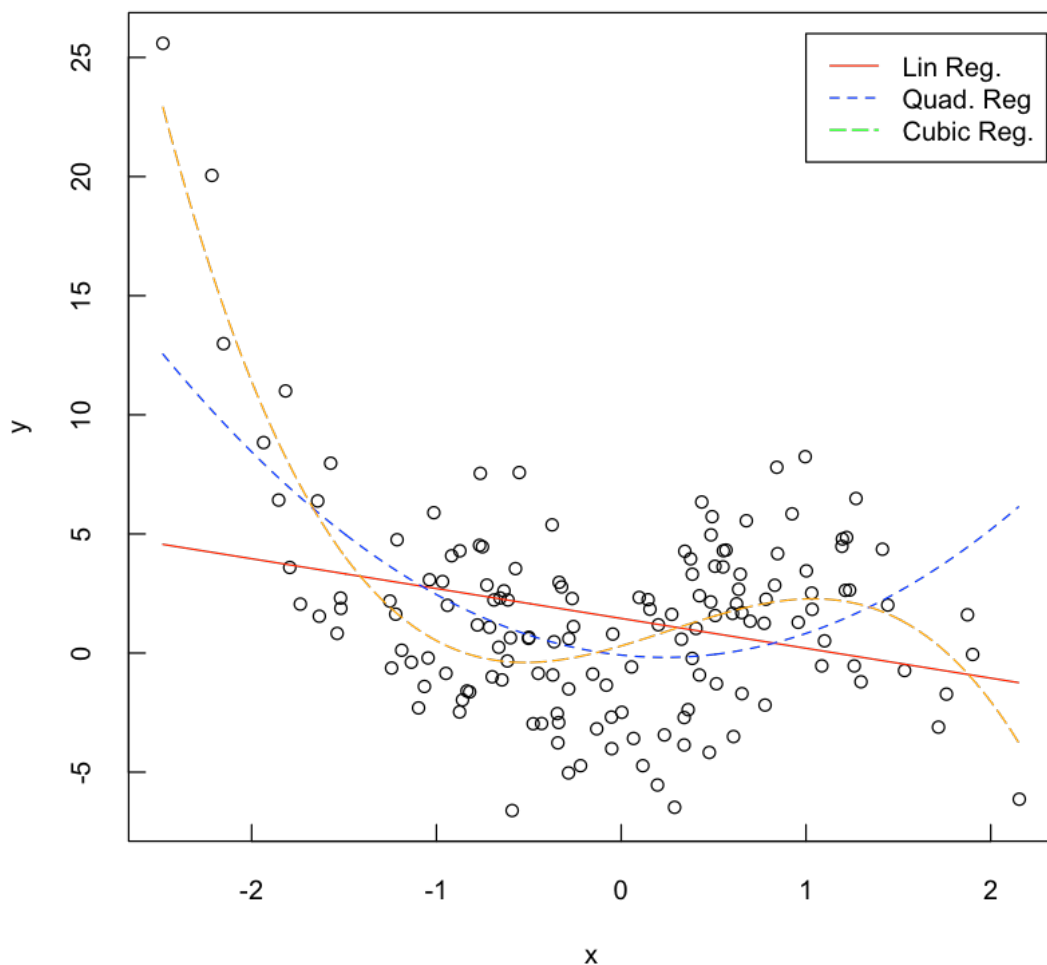
sprintf("SSR Simple Linear Regression = %.2f",ssr_SLR)
sprintf("SSR Quadratic Regression = %.2f",ssr_QR)
sprintf("SSR Cubic Regression = %.2f",ssr_CR)

```

'SSR Simple Linear Regression = 2519.69'

'SSR Quadratic Regression = 1972.54'

'SSR Cubic Regression = 1362.07'



We see the the model with smallest sum square residual is the cubic model. The difference between model predictions made by the cubic model and the empirical data is smallest. In fact, the cubic model's SSR is  $1.0 - (1362.07/2519.69)$  45% smaller than simple linear regression, and  $1.0 - (1972.54/2519.69)$  21% smaller than quadratic regression.

## 0.4 Minimizing SSE

### 0.4.1 optimal within a model

We can take the idea that a smaller SSE suggests a better model fit further. Instead of using SSE to compare different models, we can use the SSE to evaluate different parameter values inside the same model.

Consider the same dataset as above and suppose we're fitting a simple linear regression. Then our SSE becomes

$$\text{SSE}(y, \hat{y}_i) = \sum_{i=1}^N [y_i - \hat{y}_i]^2 \quad (7)$$

$$\text{SSE}(y, x, \beta_0, \beta_1) = \sum_{i=1}^N [y_i - (\beta_0 + \beta_1 x_i)]^2 \quad (8)$$

$$(9)$$

where  $y$  and  $x$  are vectors of data. Step two in above equation replaced the predicted value  $\hat{y}$  with the linear model used to make this prediction  $\beta_0 + \beta_1 x$ .

Now our SSE is a function of the data, that cannot be changed, and the parameters of our model  $\beta_0$  and  $\beta_1$ . Changing  $\beta_0$  or  $\beta_1$  will change the value of the SSE. One way to find a best fit model is to find those parameters value that make the SSE as small as possible.

#### 0.4.2 derivative

SSE is a function of  $\beta_0$  and  $\beta_1$ , and can be optimized by taking the derivative with respect to both parameters and finding the point where the derivative of these two equations equals zero simultaneously.

We take the derivative with respect to  $\beta_0$

$$\frac{d\text{SSE}(\beta_0, \beta_1)}{d\beta_0} = \sum_{i=1}^N [y_i - (\beta_0 + \beta_1 x_i)]^2 \quad (10)$$

$$\frac{d\text{SSE}(\beta_0, \beta_1)}{d\beta_0} = \sum_{i=1}^N \frac{d}{d\beta_0} [y_i - (\beta_0 + \beta_1 x_i)]^2 \quad (11)$$

$$\frac{d\text{SSE}(\beta_0, \beta_1)}{d\beta_0} = \sum_{i=1}^N -2[y_i - (\beta_0 + \beta_1 x_i)] \quad (12)$$

$$(13)$$

The above derivative can be set to zero and solved for  $\beta_0$ , our variable.

$$\sum_{i=1}^N -2[y_i - (\beta_0 + \beta_1 x_i)] = 0 \quad (14)$$

$$\sum_{i=1}^N y_i - N\beta_0 - \beta_1 \sum_{i=1}^N x_i = 0 \quad (15)$$

$$N\beta_0 = \sum_{i=1}^N y_i - \beta_1 \sum_{i=1}^N x_i \quad (16)$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x} \quad (17)$$

$$(18)$$

The value for  $\beta_0$  that optimizes the SSE is the average of our  $y$  values minus the optimal  $\beta_1$  times the average of our  $x$  values.

We must also take the derivative with respect to  $\beta_1$ .

$$\frac{d\text{SSE}(\beta_0, \beta_1)}{d\beta_1} = \sum_{i=1}^N [y_i - (\beta_0 + \beta_1 x_i)]^2 \quad (19)$$

$$\frac{d\text{SSE}(\beta_0, \beta_1)}{d\beta_1} = \sum_{i=1}^N -2x_i [y_i - (\beta_0 + \beta_1 x_i)] \quad (20)$$

$$(21)$$

The above equation can also be set to zero and solved for  $\beta_1$ .

$$\sum_{i=1}^N -2x_i [y_i - (\beta_0 + \beta_1 x_i)] = 0 \quad (22)$$

$$\sum_{i=1}^N x_i y_i - x_i \beta_0 - \beta_1 x_i^2 = 0 \quad (23)$$

$$(24)$$

At this point we can substitute the optimal value for  $\beta_0$  we derived.

$$\sum_{i=1}^N x_i y_i - x_i(\bar{y} - \beta_1 \bar{x}) - \beta_1 x_i^2 = 0 \quad (25)$$

$$\sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \bar{y} + \sum_{i=1}^N x_i \beta_1 \bar{x} - \beta_1 x_i^2 = 0 \quad (26)$$

$$\beta_1 \left( x_i^2 - \sum_{i=1}^N x_i \bar{x} \right) = \sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \bar{y} \quad (27)$$

$$\beta_1 = \frac{\sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \bar{y}}{\left( x_i^2 - \sum_{i=1}^N x_i \bar{x} \right)} \quad (28)$$

This equation for  $\beta_1$  doesn't look like anything we can recognize, but we can change the SSE we optimized to make this equation look more familiar. The equation we optimized was a function of  $\beta_0$  and  $\beta_1$ , and so adding a constant value that does not include  $\beta_0$  or  $\beta_1$  would not change the optimal  $\beta$ .

From each data point, let's subtract  $\bar{x}$  and  $\bar{y}$ , called centering our data. Then the above equation becomes

$$\beta_1 = \frac{\sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \bar{y}}{\left( x_i^2 - \sum_{i=1}^N x_i \bar{x} \right)} \quad (29)$$

$$= \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y}) - \sum_{i=1}^N (x_i - \bar{x})\bar{y}}{\sum_{i=1}^N (x_i - \bar{x})^2 - \bar{x} \sum_{i=1}^N (x_i - \bar{x})} \quad (30)$$

$$= \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^N (x_i - \bar{x})^2} \quad (31)$$

$$= \frac{Cov(X, Y)}{Var(X)} \quad (32)$$

Centering our data, we see the optimal  $\beta_1$  is the covariance between  $y$  and  $x$  divided by the variance of  $x$ .

We can also write the above in matrix form. The covariance between  $X$  and  $Y$  is written

$$Cov(X, Y) = X'Y$$

where  $X = x - \bar{x}$  and  $Y = y - \bar{y}$ , and the variance of  $X$  is written

$$Var(X) = X'X.$$

Then the expression for  $\beta_1$  is

$$\beta_1 = (X'X)^{-1}(X'Y)$$

But by adding a column of 1s to  $X$ , we can see that the above expression works for both  $\beta_1$  and  $\beta_0$ . In fact, this expression will work for any design matrix  $X$ .

So we can write

$$\beta = (X'X)^{-1}(X'y)$$

To see this more clearly, let's generalize our derivations of  $\beta_0$  and  $\beta_1$  to multiple  $\beta$ s.

We first form our SSE for multiple linear regression

$$\text{SSE}(y, X, \beta_0, \beta_1) = \sum_{i=1}^N [y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in})]^2$$

where  $x_{ij}$  is observation  $i$  for variable  $j$ . Taking the derivative for every  $\beta$  and setting equal to 0 we have

For  $\beta_0$

$$\sum_{i=1}^N 1[y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in})] = 0 \quad (33)$$

$$\sum_{i=1}^N (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in}) = \sum_{i=1}^N 1y_i \quad (34)$$

$$(35)$$

For  $\beta_1$

$$\sum_{i=1}^N x_{i1}[y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in})] = 0 \quad (36)$$

$$\sum_{i=1}^N x_{i1}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in}) = \sum_{i=1}^N x_{i1}y_i \quad (37)$$

$$(38)$$

For  $\beta_2$

$$\sum_{i=1}^N x_{i2}[y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in})] = 0 \quad (39)$$

$$\sum_{i=1}^N x_{i2}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in}) = \sum_{i=1}^N x_{i2}y_i \quad (40)$$

$$(41)$$

For  $\beta_n$

$$\sum_{i=1}^N x_{in}[y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in})] = 0 \quad (42)$$

$$\sum_{i=1}^N x_{in}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_n x_{in}) = \sum_{i=1}^N x_{in} y_i \quad (43)$$

$$(44)$$

The right hand side of this system of equations can be rewritten as

$$X'y,$$

and the left hand side can be rewritten as

$$(X'X)\beta.$$

Our system of equations is then

$$(X'X)\beta = X'y.$$

We can solve for  $\beta$  by left multiplying each side by  $(X'X)^{-1}$

$$\beta = (X'X)^{-1}X'y$$

and arriving at the same solution for multiple linear regression that we found with simple linear regression.

### 0.4.3 the linear in linear regression

Note that all the relationships linking  $x$  and  $y$  could be expressed using the same equation

$$y \sim N(X\beta, \sigma^2)$$

Why then is this called **linear** regression?

The **linear** in linear regression refers to the parameters. Lets look at the model form of a cubic regression.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 \quad (45)$$

$$\epsilon \sim N(0, \sigma^2) \quad (46)$$



We can rewrite the above equation, so that it looks a bit more like multiple regression.

$$y = \beta_0 + \beta_1 x + \beta_2 q + \beta_3 r \quad (47)$$

$$q = x^2 \quad (48)$$

$$r = x^3 \quad (49)$$

$$\epsilon \sim N(0, \sigma^2) \quad (50)$$

Our equation now is linear in  $\beta$  and in reference to three variables:  $x$ ,  $q$ , and  $r$ . We can do this with any functional form for  $x$ . The “linear” refers to the parameters.

[ ]: