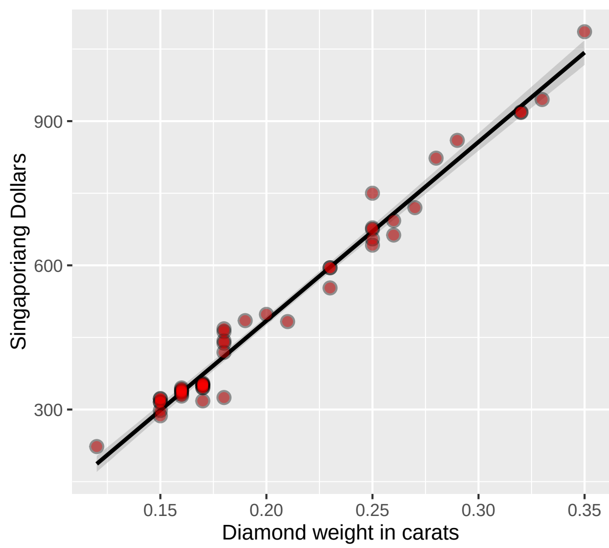
Summary Regression Models

Desiré De Waele

17 maart 2016

Used data summary

The data used throughout this summary is the diamond dataset of the UsingR package. These plots show the relation between diamond weight (carat) and price in Dollars, the linear model fit and its residuals.



Formulas

These are the formulas for variance, covariance and correlation.

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} X_i^2 - n\bar{X}^2\right)$$

$$Cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y}) = \frac{1}{n-1}\left(\sum_{i=1}^{n} X_i Y_i - n\bar{X}\bar{Y}\right)$$

$$Cor(X,Y) = \frac{Cov(X,Y)}{S_x S_y}$$

These are the formulas for linear modelfit, slope and intercept.

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i | \epsilon_i \sim N(0, \sigma^2)$$

$$\hat{\beta}_1 = Cor(Y,X)\frac{Sd(Y)}{Sd(X)}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$$

Code for coefficients, prediction outcomes and residuals

Calculating coefficients of the linear fit by hand and with lm function.

```
y <- diamond$price; x <- diamond$carat; n <- length(y)
beta1 <- cor(y, x) * sd(y) / sd(x)
beta0 <- mean(y) - beta1 * mean(x)
fit <- lm(y ~ x)
rbind(c(beta0, beta1), coef(fit))
```

```
##      (Intercept)        x
## [1,]   -259.6259 3721.025
## [2,]   -259.6259 3721.025
```

Predicting outcomes with a linear model fit by hand and with predict function

```
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x)
newx <- c(0.16, 0.27, 0.34)
byhand <- coef(fit)[1] + coef(fit)[2] * newx
byfunction <- predict(fit, newdata = data.frame(x = newx))
rbind(byhand, byfunction)
```

```
##                   1        2        3
## byhand     335.7381 745.0508 1005.523
## byfunction 335.7381 745.0508 1005.523
```

Calculating residuals by hand and with resid function.

```
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x)
byhand <- y - predict(fit)
byfunction <- resid(fit)
rbind(sort(byhand)[1:3], sort(byfunction)[1:3])
```

```
##              4        27        18
## [1,] -85.15857 -54.94832 -44.84055
## [2,] -85.15857 -54.94832 -44.84055
```

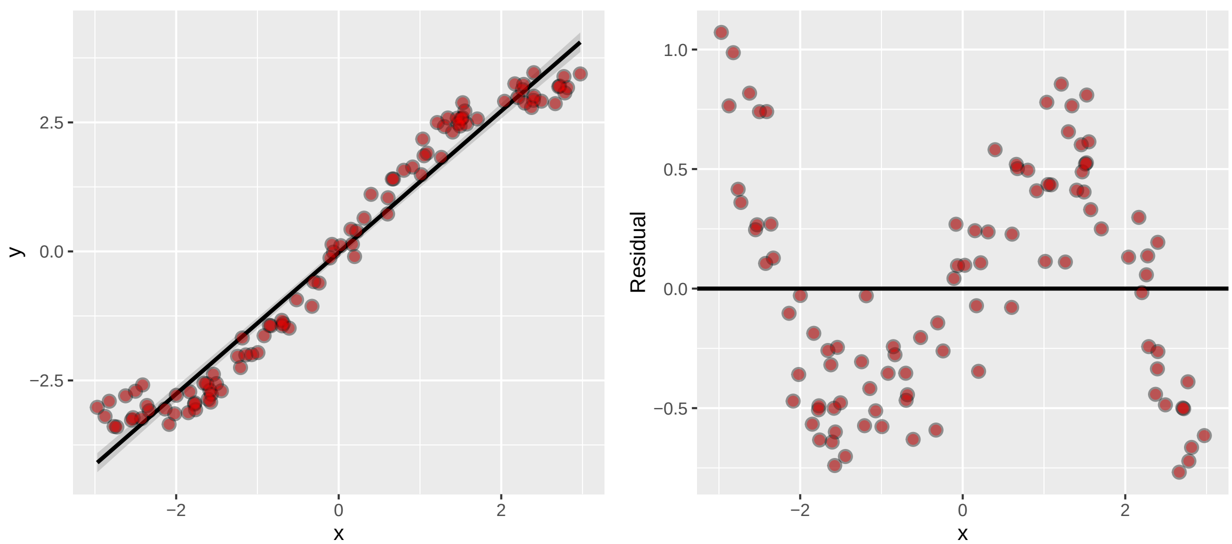
Code for plotting a linear model fit and its residuals

```
# Generating the data
x = runif(100, -3, 3); y = x + sin(x) + rnorm(100, sd = .2);

# Plotting data with their linea fit
g1 = ggplot(data.frame(x = x, y = y), aes(x = x, y = y))
g1 = g1 + geom_smooth(method = "lm", colour = "black")
g1 = g1 + geom_point(size = 3, colour = "black", alpha = 0.4)
g1 = g1 + geom_point(size = 2, colour = "red", alpha = 0.4)
```

```r
# Plotting the residuals of the data
g2 = ggplot(data.frame(x = x, y = resid(lm(y ~ x))), aes(x = x, y = y))
g2 = g2 + geom_hline(yintercept = 0, size = 1);
g2 = g2 + geom_point(size = 3, colour = "black", alpha = 0.4)
g2 = g2 + geom_point(size = 2, colour = "red", alpha = 0.4)
g2 = g2 + xlab("x") + ylab("Residual")

grid.arrange(g1, g2, ncol=2)
```



Residual variation

Formula and code for calculating the residual variation, by hand and with formula.

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^n e_i^2}{n-2}$$

```r
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x)
byhand <- sqrt(sum(resid(fit)^2) / (n - 2))
byformula <- summary(fit)$sigma
rbind(byhand, byformula)
```

```
##               [,1]
## byhand    31.84052
## byformula 31.84052
```

The total variation sums the residual variation and the regression variation, R squared is the regression variation divided by the total variation.

$$\sum_{i=1}^n (Y_i - \bar{Y})^2 = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 + \sum_{i=1}^n (\hat{Y}_i - \bar{Y})^2$$
$$R^2 = \frac{\sum_{i=1}^n (\hat{Y}_i - \bar{Y})^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2}$$

Standard error of slope and intercept

$$\sigma^2_{\hat\beta_1} = Var(\hat\beta_1) = \frac{\sigma^2}{\sum_{i=1}^n (X_i - \bar X)^2}$$

$$\sigma^2_{\hat\beta_0} = Var(\hat\beta_0) = \left(\frac{1}{n} + \frac{\bar X^2}{\sum_{i=1}^n (X_i - \bar X)^2}\right)\sigma^2$$

```r
y <- diamond$price; x <- diamond$carat; n <- length(y)
beta1 <- cor(y, x) * sd(y) / sd(x)
beta0 <- mean(y) - beta1 * mean(x)
e <- y - beta0 - beta1 * x # Residuals
sigma <- sqrt(sum(e^2) / (n-2)) # Residual variation
ssx <- sum((x - mean(x))^2) # Denominator of coefficient standard errors

# Standard errors coefficients
seBeta0 <- (1 / n + mean(x) ^ 2 / ssx) ^ .5 * sigma
seBeta1 <- sigma / sqrt(ssx)
# t-statistics when H0: beta mean = 0
tBeta0 <- beta0 / seBeta0; tBeta1 <- beta1 / seBeta1
# Calculating p-values
pBeta0 <- 2 * pt(abs(tBeta0), df = n - 2, lower.tail = FALSE)
pBeta1 <- 2 * pt(abs(tBeta1), df = n - 2, lower.tail = FALSE)

# Setting up a table
table <- rbind(c(beta0, seBeta0, tBeta0, pBeta0), c(beta1, seBeta1, tBeta1, pBeta1))
colnames(table) <- c("Estimate", "Std. Error", "t value", "Pr(>|t|)")
rownames(table) <- c("(Intercept)", "x"); table
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -259.6259   17.31886 -14.99094 2.523271e-19
## x           3721.0249   81.78588  45.49715 6.751260e-40
```

```r
# The easy way
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x);
summary(fit)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -259.6259   17.31886 -14.99094 2.523271e-19
## x           3721.0249   81.78588  45.49715 6.751260e-40
```

Confidence interval for slope

```r
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x)
sumCoef <- summary(fit)$coefficients
# Calculating 95% interval for the price increase per 0.1 carat
(sumCoef[2,1] + c(-1, 1) * qt(.975, df = fit$df) * sumCoef[2, 2]) / 10
```

```
## [1] 355.6398 388.5651
```

Confidence and prediction intervals at given point

$$SE_{confidence} \text{ at } x_0 = \hat{\sigma}\sqrt{\frac{1}{n} + \frac{(x_0 - \bar{X})^2}{\sum_{i=1}^{n}(X_i - \bar{X})^2}}$$

$$SE_{prediction} \text{ at } x_0 = \hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{X})^2}{\sum_{i=1}^{n}(X_i - \bar{X})^2}}$$

Calculating the confidence interval (for the regression line) and prediction interval (for the data) at the mean of carat weight (x), by hand and with predict function with interval arguments.

```
yhat <- fit$coef[1] + fit$coef[2] * mean(x)

confByHand <- yhat + c(-1, 1) * qt(.975, df = fit$df) * summary(fit)$sigma / sqrt(length(y))
confFormula <- predict(fit, newdata = data.frame(x = mean(x)), interval = ("confidence"))

predByHand <- yhat + c(-1, 1) * qt(.975, df = fit$df) * summary(fit)$sigma * sqrt(1 + 1/length(y))
predFormula <- predict(fit, newdata = data.frame(x = mean(x)), interval = ("prediction"))

table <- rbind(c(yhat, confByHand), confFormula, c(yhat, predByHand), predFormula)
rownames(table) <- c("Confidence by hand", "Confidence by formula",
                     "Prediction by hand", "Prediction by formula")
colnames(table) <- c("Estimate", "Lower limit", "Upper limit"); table
```

```
##                       Estimate Lower limit Upper limit
## Confidence by hand    500.0833    490.8325    509.3342
## Confidence by formula 500.0833    490.8325    509.3342
## Prediction by hand    500.0833    435.3275    564.8392
## Prediction by formula 500.0833    435.3275    564.8392
```

Example plot with confidence and prediction intervals, to help interpret them.