STA130 W10

Monday, November 18, 2019 2:09 PM

Linear regression for a **numerical response** variable

- Describe the association between one or more explanatory variable(s) and a numerical response
- Make predictions for a numerical response based on one or more predictor(s)

Example: Predicting height from the length of a footprint
Goal 1: How much does foot length explain the variability in height
Goal 2: Predict height based on foot length
Describing association between 2 numerical variables
heights %>% ggplot(aes(x=footLength, y=height)) + geom_point()

- Strength of association(strong/moderate/weak)
- Directions: positive, negative
- Form: linear, quadratic, non-linear

Quantifying association
The **correlation** summarises the strength and direction of a linear relationship between two numerical variables. The **sample correlation** between x and y for n observations (x_1, y_1)…(x_n, y_n)

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Sign of **r** gives direction

- r>0 : positive association r<0 : negative

Magnitude of **r** gives strength

- |r| close to 0 -> weak association close to 1 -> strong

- r is between -1 and 1 (r= +-1 means perfect linear relationship with positive or negative slope, respectively)
- Correlation between x and y is **same** as correlation between y and x
- r only describes the **linear relationship** between two **numerical** variables

Calculating in R
Calculate the correlation:
cor(heights$footLength, heights$height)
-> corr. between foot length and height
Cor(x, y)calculates the correlation r between 2 numerical vectors

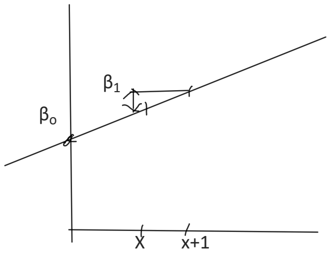
Correlation
If two variables have **strong positive** correlation (r close to 1)
high values of one variable tend to co-occur with high values of the other
If two variables have **strong negative** correlation (r close to -1)
high values of one variable tend to co-occur with low values of the other
Limitations

- Large |r| means that x is a good candidate predictor for y
- |r| close to 0 -> **we don't know** automatically if x would be a good predictor or not

Modeling linear associations
Draw a line through these points that captures the relationship between foot length and height.
Whether our goal is to use foot length to (1) explain the variation in height OR (2) predict height, we need to build a **model**.
One option is a **simple linear regression model**, which assumes that there is a "best" straight line that explains the real relationship between x and y, and that the values we observed randomly deviate from this line

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

- Y_i: response variable (or dependent variable, target variable, …) for i^{th} observation
- x_i: independent variable (or predictor, covariate, feature, input, …) for i^{th} observation
- β_0: intercept parameter
- β_1: slope parameter
- ϵ_i: random error term for i^{th} observation



"beta" (Greek letter)

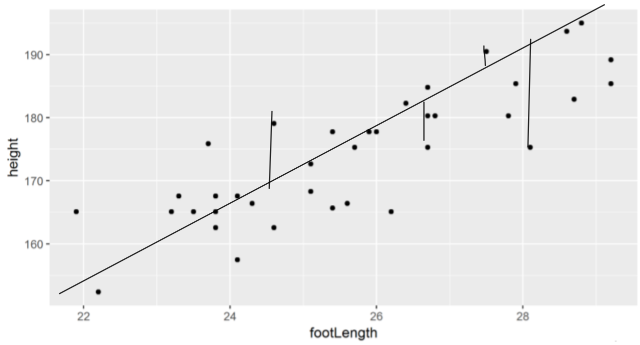
If we have data for the entire population

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

- There is a **true line** with slope β_1 and intercept β_0 which describes the overall relationship between x and y
- There are **random deviations** between this line and the individual observations Y_i

The best line

When we have a data from a sample, we want to find a line which is **as close as possible** to as many points as possible.
Find the line that **minimizes the sum of squared distances** between points and the line



Least squares Regression Line

Sum of vertical distances between the points and the line (i.e. squared differences between height of point and line) is minimized

Interactive Applet: http://mathlets.org/mathlets/linearregression/

Simple linear regression: Estimating the coefficients using R

Use the **lm()** function to fit a linear regression model
Same syntax as we used to fit classification trees(rpart()) to specify the response and predictors

```
mod_height <- lm(height ~ footLength, data = heights)
summary(mod_height)$coefficients
```

Pvalue-next class

*lm -> linear model
*height -> y, footLength -> x

$\hat{\beta}_0 = \bar{y}_f$

```
##                  Estimate     Std.  Error      t value    Pr(>|t|)
## (Intercept) 64.125614 11.4850534 5.583397 2.122303e-06 ->β0
## footLength   4.291253  0.4459507 9.622708 9.833229e-12 ->β1
```

β_1

$H_0: \beta_1 = 0$

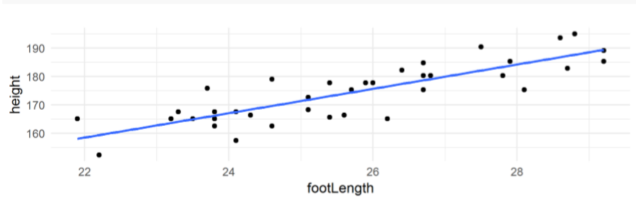
$H_a: \beta_1 \neq 0$

Foot length -> name of x variable
(Intercept) is the **estimate** of β_0 (i.e. β-hat$_0$) ->64.13
footLength is the **estimate** of β_1 (i.e. β-hat$_1$) ->4.29
Std. Error is the estimated standard deviation of β based on the one sample - no simulation (called standard error or SE)
T value is the number of SEs β is away from 0
Pr(>|t|) is the p-value for a hypothesis test of H:β0 vs H:β0 .

Add the fitted regression line to the plot

```
##            Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 64.125614 11.4850534 5.583397 2.122303e-06
## footLength   4.291253  0.4459507 9.622708 9.833229e-12
heights %>% ggplot(aes(x=footLength, y=height)) + geom_point() +
   geom_smooth(method="lm", se=FALSE) + theme_minimal()
```

*Geom_smooth -> adds fitted regression line to a plot



The blue line is the estimated regression line, with intercept β-hat$_0$ = 64.13 and slope β-hat$_1$ = 4.29

Deriving Estimators of Least Squares Regression Coefficients

Find the line (i.e. find β_0 and β_1) which minimizes the sum of squared errors for the n observations: $\sum_{i=1}^{n} \epsilon_i^2$

$$f(\beta_0, \beta_1) = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

Goal: Find the values of the intercept β_0 and slope β_1 that minimize the function $f(\beta_0, \beta_1)$. We treat the data $(x_1, y_1), \ldots, (x_n, y_n)$ as constants to do this.

The derivative of $f(\beta_0, \beta_1)$ with respect to β_0 treats β_1 as a constant. This is also called the *partial derivative* as is denoted as $\frac{\partial f}{\partial \beta_0}$

To find the values of β_0 and β_1 which minimize $f(\beta_0, \beta_1)$ we set both partial derivatives $\frac{\partial f}{\partial \beta_0}$ and $\frac{\partial f}{\partial \beta_1}$ to 0 and solve:

$$\frac{\partial f}{\partial \beta_0} = -2\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0$$

$$\frac{\partial f}{\partial \beta_1} = -2\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)x_i = 0$$

The values of β_0 and β_1 which solve the above equations are denoted β-hat$_0$ and β-hat$_1$ respectively.

Simple linear regression: Estimating the coefficients

Setting the equations to 0 and solving for β_0 and β_1 give estimates:

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = r\frac{s_y}{s_x}$$

where $\bar{x}$ and $\bar{y}$ are sample means and s_x and s_y are sample standard deviations for the x and y observations, respectively.

β-hat$_0$ and **β-hat$_1$** are the **least squares estimators** of β_0 and β_1

Interpreting the slope and intercept with a numerical predictor (the "true" model e.g. is the previous one)

The estimated simple linear regression of height on footLength (i.e. the fitted line) is:

*"hat" notation for **estimates** of population parameters

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x,$$

where $\hat{y}$ (called the **fitted** or **predicted value**) is the estimated average value of y when the predictor is equal to x.

- The **slope** $\hat{\beta}_1$ is the average change in y for a 1-unit change in x
- The **intercept** $\hat{\beta}_0$ is the average of y when $x_i = 0$ (often this doesn't make sense, but tells us the height of the line).

The difference between the observed and predicted value of y for the i^{th} observation is called the **residual** $e_i = y_i - \hat{y}_i$

We can calculate **residual** for any fitted regression line

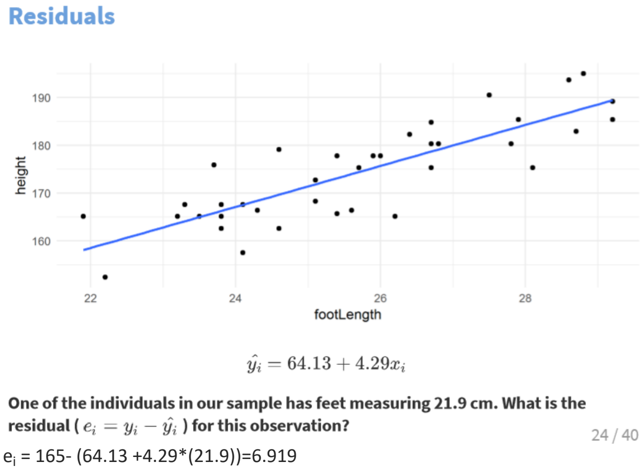
Statement1: As the length of feet increases by 1cm, height increase by 4.29cm

- Not exactly, β-hat$_1$ tells us about the average change in y.
- It is not one "causes" other

Statement2: A 1cm increase in foot length causes a 4.29cm increase in height

- We can't make causal conclusions bases on theses data

y_2-y_1 =(β-hat$_0$ +β-hat$_1$x$_2$) - (β-hat$_1$ $_+$ β-hat$_1$x$_1$)
= β-hat$_1$ (x$_2$-x$_1$)=β-hat$_1$ (1) = 4.2
x$_1$-x$_2$=x$_1$+1



Residuals

$$\hat{y}_i = 64.13 + 4.29x_i$$

One of the individuals in our sample has feet measuring 21.9 cm. What is the residual ($e_i = y_i - \hat{y}_i$) for this observation?

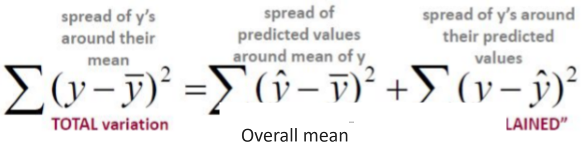
24 / 40

e_i = 165- (64.13 +4.29*(21.9))=6.919

How well does the fitted regression line "explain" the variation in heights (y) ?

mean(heights$height)
## [1] 174.325
*overall average height of sample

How well does the fitted regression line "explain" the variation in (y) heights?



spread of y's around their mean

spread of predicted values around mean of y

spread of y's around their predicted values

$$\sum(y-\bar{y})^2 = \sum(\hat{y}-\bar{y})^2 + \sum(y-\hat{y})^2$$

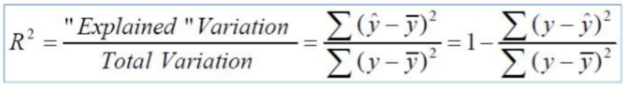
TOTAL variation

LAINED"

Overall mean

Fitted values
From own fitted negative line

Coefficient of determination



$$R^2 = \frac{\text{"Explained" Variation}}{\text{Total Variation}} = \frac{\sum(\hat{y}-\bar{y})^2}{\sum(y-\bar{y})^2} = 1 - \frac{\sum(y-\hat{y})^2}{\sum(y-\bar{y})^2}$$

The coefficient of determination R^2 is the proportion of the variability in y which is explained by our fitted regression line

- R^2 close to 1 indicates that most of the variability in t is explained by x
- R^2 close to 0 indicates that x does not explain much of the variability in y

Note: R^2 is **the square of the correlation r**

Small if model is good

```
summary(mod_height)$r.squared
```

->fitted model in R.-=

```
## [1] 0.7090274
cor(heights$footLength, heights$height)^2
## [1] 0.7090274
```

How well does this fitted regression line perform as a predictive model?

Divide our sample into **training / testing datasets** to measure the performance of our prediction model.

Strategy:

1. Divide sample into training dataset and testing dataset
   Same as last class
2. Fit a prediction model (today: a linear regression model) using the training data
   Involves using the LM function and to get β-hat$_0$ and β-hat$_1$
3. Make predictions for the testing data and compare the predictions to the true responses

Validate prediction model using the training/testing approach

```
set.seed(3)
n <- nrow(heights)
training_indices <- sample(1:n, size=round(0.8*n))
train <- heights[training_indices,]
test <- heights[-training_indices,]
```

- Here we have 80% in training and 20% in testing data

What we want to make prediction for

```
# Fit model to training data
mod_height_train <- lm(height ~ footLength, data=train)
```

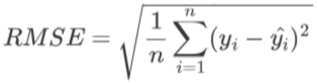
Fitted model

- Build our model using training data

```
# Make prediction on testing data using training model
yhat_test <- predict(mod_height_train, newdata = test)
yhat_test
##        8       11       18       25       26       35       37       39
## 185.2271 182.6276 184.3606 173.5295 158.3660 166.5976 165.2979 167.8974
```

The **root mean squared error (RMSE)** measures the prediction error

Y-hat, testing data
Y, true value



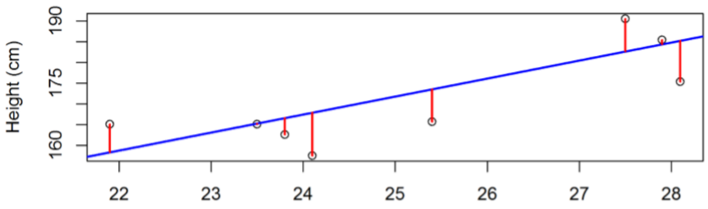
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$$

RMSE can be used to compare different sizes of datasets (must all be in the same units though) and to compare different models on the same dataset.

Calculate RMSE

Taking a square root means that RMSE is in the same units (and scale) as y

*The smaller, the better the model



Comparing the accuracy of predictions for training and testing datasets

We know that our predictions will be closer to the true values for the training data (because we used the training data to fit the model) than for the testing data

If there is a big difference between RMSE for predictions based on training and testing data, this suggest our model "learned" the training data "too well": this is called **overfitting**.

Recall: the goal of building a linear model for prediction is to generalize the pattern to make predictions for **new** observations, so this is not a good thing

- We want a prediction model that works well for **new** data

```
# Fit model to training data
mod_height_train <- lm(height ~ footLength, data=train)
```

->Fitted model

RMSE for predictions of individuals in the **testing** dataset

```
# Make predictions for testing data using training model
yhat_test <- predict(mod_height_train, newdata = test)
y_test <- test$height;                n_test <- nrow(test)
```

y-test is a vector containing all values of price in the test data set

```
# RMSE for predictions of individuals in the testing dataset
sqrt(sum((y_test - yhat_test)^2) / n_test)
## [1] 7.003343
```

->RMSE for test data

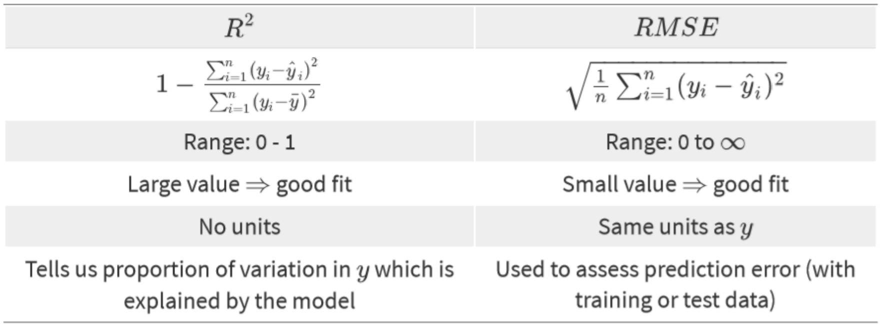
RMSE for predictions of individuals in the **training** dataset

```
# Make predictions for training data using training model
yhat_train <- predict(mod_height_train, newdata = train)
y_train <- train$height;                n_train <- nrow(train)
# RMSE for predictions of individuals in the training dataset
sqrt(sum((y_train - yhat_train)^2) / n_train)
## [1] 4.894985
```

->RMSE for training data

Here there is a relatively big difference: the RMSE is over 40% larger for predictions on the testing dataset than on the training dataset
This suggests that our predictive model may be overfitting the training data and is not very useful for to make predictions for new observations

Comparing R^2 and RMSE



R^2	$RMSE$
$1-\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}$	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$
Range: 0 - 1	Range: 0 to ∞
Large value $\Rightarrow$ good fit	Small value $\Rightarrow$ good fit
No units	Same units as y
Tells us proportion of variation in y which is explained by the model	Used to assess prediction error (with training or test data)