Week 10: Multivariate Regression Improved Prediction

Modeling and Predicting Using Many Variables

Scott Schwartz

Nov 14, 2022

Returning to the Heights Data

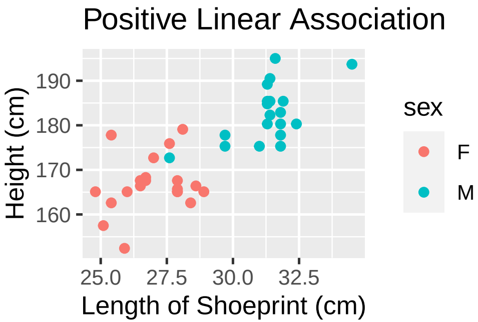
Estimation of stature from foot and shoe length: applications in forensic science

- by B. Rohren

```r
library(tidyverse)
heights <- read_csv("heights.csv")
glimpse(heights)
```

```
## Rows: 40
## Columns: 7
## $ ...1       <dbl> 1, 2, 3, 4, 5, 6, 7, 8,
## $ sex        <chr> "M", "M", "M", "M", "M",
## $ age        <dbl> 67, 47, 41, 42, 48, 34,
## $ footLength <dbl> 27.8, 25.7, 26.7, 25.9,
## $ shoePrint  <dbl> 31.3, 29.7, 31.3, 31.8,
## $ shoeSize   <dbl> 11.0, 9.0, 11.0, 10.0, 1
## $ height     <dbl> 180.3, 175.3, 184.8, 177
```

```r
# {r, fig.width=3, fig.height=2}
heights %>% ggplot(aes(
  x=shoePrint, y=height, color=sex)) +
  geom_point() + theme_gray() +
  labs(title="Positive Linear Association",
       x="Length of Shoeprint (cm)",
       y="Height (cm)")
```



As Promised Last Class...

```r
library(broom); lm(height~shoePrint, data=heights) %>% broom::tidy() # R^2 0.6608845
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    80.9     10.9        7.43 6.50e- 9
## 2 shoePrint       3.22     0.374      8.61 1.86e-10
```

```r
lm(height~sex, data=heights) %>% tidy() %>% as.matrix() # R^2 0.6283874
```

```
##      term          estimate    std.error  statistic    p.value
## [1,] "(Intercept)" "166.82381" "1.357760" "122.866909" "5.085412e-51"
## [2,] "sexM"        " 15.79198" "1.970046" "  8.016048" "1.085391e-09"
```

```r
lm(height~shoePrint+sex, data=heights) %>% tidy() %>% as.matrix() # R^2 0.6909145
```

```
##      term          estimate     std.error    statistic  p.value
## [1,] "(Intercept)" "112.734096" "19.8103430" "5.690669" "1.647767e-06"
## [2,] "shoePrint"   "  2.007926" " 0.7339256" "2.735872" "9.498622e-03"
## [3,] "sexM"        "  7.001892" " 3.6929712" "1.896005" "6.578969e-02"
```

```r
# summary(lm(height~shoePrint+sex, data=heights))$r.squared
```

What can we do with this?

The Statistical Inference Landscape

Hypothesis Testing	**Estimation***	**Model Prediction**
$H_0 : \beta_1 = 0$ reject at $\alpha = 0.05$	$\hat{\beta}_1 = 2.007926$	$\hat{y}_i \approx 112.7 + 2x_{1i} + 7 \times \mathrm{I}(x_{2i} = \mathrm{M})$
$H_0 : \beta_2 = 0$ fail to reject at $\alpha = 0.05$	$\hat{\beta}_2 = 7.001892$	R^2 0.6909145

*Parameter Estimation includes Confidence Intervals as well (we just aren't covering it)

```
lm(height~shoePrint+sex, data=heights) %>% tidy() # R^2 0.6909145
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)   113.      19.8        5.69 0.00000165
## 2 shoePrint       2.01     0.734      2.74 0.00950
## 3 sexM            7.00     3.69       1.90 0.0658
```

The Coefficient of Determination R^2

$$R^2 = \frac{\sum_{i=1}^n (y_i - \bar{y})^2 - \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = r^2$$

The "Proportion of Variation Explained" is a *Measure of Model Fit*

$$Y_i = \beta_0 + \beta_1 x_{1i} + \epsilon_i$$

```
summary(lm(height~shoePrint,
           data=heights))$r.squared
```

```
## [1] 0.6608845
```

$$Y_i = \beta_0 + \beta_1 I(x_{2i} = \text{M}) + \epsilon_i$$

```
summary(lm(height~sex,
           data=heights))$r.squared
```

```
## [1] 0.6283874
```

$$Y_i = \beta_0 + \beta_1 x_{1i} + \beta_1 I(x_{2i} = \text{M}) + \epsilon_i$$

```
summary(lm(height~shoePrint+sex,
           data=heights))$r.squared
```

```
## [1] 0.6909145
```

- What does $I(x_{2i} = \text{M})$ mean?
- Do you understand this model?
- Is the increase in R^2 expected?
- Which model best predicts the data?
- Is R^2 a good way to decide?

The Coefficient of Determination R^2

$$R^2 = \frac{\sum_{i=1}^n (y_i - \bar{y})^2 - \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = r^2$$

The "Proportion of Variation Explained" is a *Measure of Model Fit*

$$Y_i = \beta_0 + \beta_1 x_{1i} + \epsilon_i$$

```
# Same for Simple Linear Regression
cor(heights$shoePrint, heights$height)^2
```

```
## [1] 0.6608845
```

$$Y_i = \beta_0 + \beta_1 \mathrm{I}(x_{2i} = \mathrm{M}) + \epsilon_i$$

```
cor(as.numeric(as.factor(heights$sex)),
    heights$height)^2
```

```
## [1] 0.6283874
```

$$Y_i = \beta_0 + \beta_1 x_{1i} + \beta_1 \mathrm{I}(x_{2i} = \mathrm{M}) + \epsilon_i$$

```
summary(lm(height~shoePrint+sex,
           data=heights))$r.squared
```

```
## [1] 0.6909145
```

```
# `cor()` to match R^2 ?
# doesn't make sense here...
```

- Is the increase in R^2 expected?
- Which model best predicts the data?
- Is R^2 a good way to decide?

Variable Selection with Hypothesis Testing

p-values are for the “Last Variable Added”

```r
lm(height~shoePrint, data=heights) %>%
  tidy()%>%select(term,estimate,p.value)
```

```
## # A tibble: 2 x 3
##   term        estimate  p.value
##   <chr>          <dbl>    <dbl>
## 1 (Intercept)    80.9  6.50e- 9
## 2 shoePrint       3.22 1.86e-10
```

```r
lm(height~sex, data=heights) %>%
  tidy()%>%select(term,estimate,p.value)
```

```
## # A tibble: 2 x 3
##   term        estimate  p.value
##   <chr>          <dbl>    <dbl>
## 1 (Intercept)    167.  5.09e-51
## 2 sexM            15.8 1.09e- 9
```

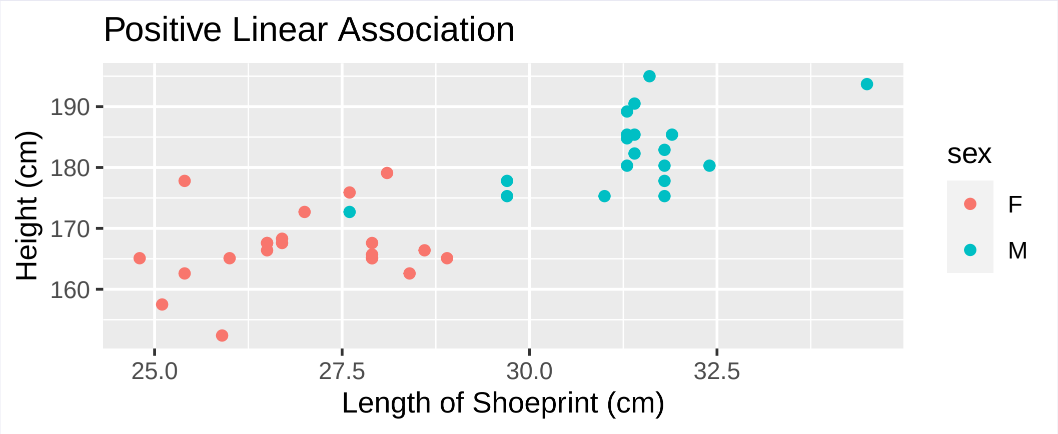
```r
lm(height~shoePrint+sex,data=heights)%>%
  tidy()%>%select(term,estimate,p.value)
```

```
## # A tibble: 3 x 3
##   term        estimate    p.value
##   <chr>          <dbl>      <dbl>
## 1 (Intercept)   113.   0.00000165
## 2 shoePrint       2.01 0.00950
## 3 sexM            7.00 0.0658
```

- Both p-values are weaker when the model includes both of the variables
- The other variable already in the model already explains the variation

Which Variable Predicts Height?

The model can't tell which variable to attribute the explanation too



Thus, the "effect" of each variable detracts from each other, increasing both p-values

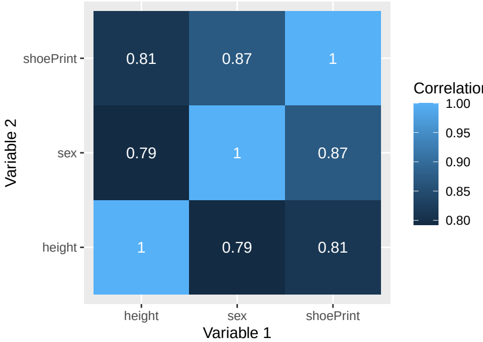
[Not Un]Observed Confounding: Multicollinearity

```r
cor(as.numeric(as.factor(heights$sex)),
    heights$shoePrint)
```

```
## [1] 0.8700046
```

```r
# {r, fig.width=5, fig.height=3.5}
heights %>% select(height,shoePrint,sex) %>%
  mutate(sex=as.numeric(as.factor(sex))) %>%
  cor() %>% as_tibble(rownames="rowname") %>%
  pivot_longer(cols=!rowname,
    names_to="Variable 1",
    values_to="Correlation") %>%
  rename("Variable 2"=rowname) %>%
  ggplot(aes(x=`Variable 1`, y=`Variable 2`,
             fill=Correlation,
             label=round(Correlation,2))) +
  geom_tile() + geom_text(color="white")
```

A multivariate linear regression model can't "de-tangle" contributions of correlated (positive or negative) variables

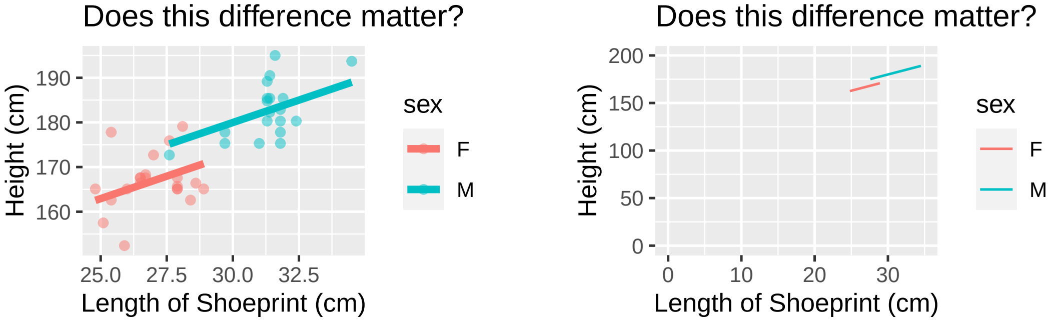


- As an extreme example, suppose x_{1i} and x_{2i} are identical, then $\hat{y}_i = x_{1i} + x_{2i} = 2x_{1i} + 0x_{2i}$ and linear regression can't tell the difference

Statistical VS Practical Significance

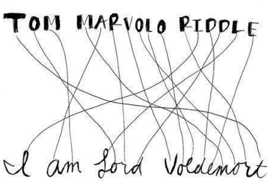
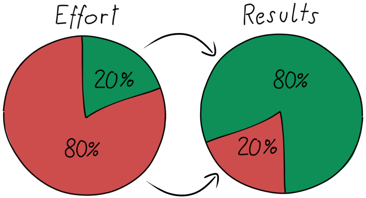
```r
least_squares_fit <- lm(height~shoePrint+sex,data=heights) # If sexM was significant
summary(least_squares_fit)$coefficients # would it actually be practically relevant?
```

```
##               Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 112.734096 19.8103430 5.690669 1.647767e-06
## shoePrint     2.007926  0.7339256 2.735872 9.498622e-03
## sexM          7.001892  3.6929712 1.896005 6.578969e-02
```



An Alternative Method: 80/20 Train-Test Split

1. R^2-based model selection is problematic since more explanatory variables means more parameters which means larger R^2 because the model can Overfit the data
   - Overfitting data menas a random chance pattern gets interpreted as a real pattern
2. Variable Selection Hypothesis Testing is problematic since testing order is arbitrary
3. Another (very good) idea (from data science) is to fit the model on some data, and then score the model on some different data. **This is called a Train-Test split**

In Sample Performance	Variable Selection	Out of Sample Performance
		

An Alternative Method: 80/20 Train-Test Split

The "80/20 rule" for a Train-Test Split analysis is

Fit a model on 80% of the data $\rightarrow$ "score" the model on the remaining 20%

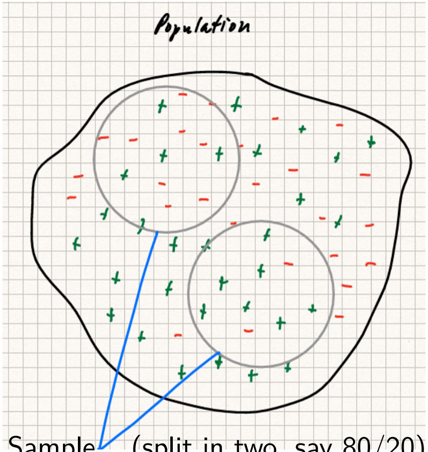
$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$ is one possible model score; but, it's simpler to just use

$$\text{Root Mean Square Error} \quad \text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{m}(y_j - \hat{y}_j)^2}$$

- $\hat{y}_j$ is the prediction from the model fit using 80% of the data
- j indexes over the m data points comprising the remaining 20%
- The square root keeps the original (rather than squared) units
- And the denominator isn't $n-1$ like when estimating variance

An Alternative Method: 80/20 Train-Test Split

- RMSE is about the fact that you're doing something (predicting) in the sample
- *So you should try to see how well you can do that thing in the population...*



Sample (split in two, say 80/20)

← **Here we split a "representative" population sample into two "representative" samples**

1. You fit the model based on a "representative" sample
2. So subsamples are "representative of the population"
3. Use 80% of the data to fit the "representative" model
4. Use 20% to see if the model's actually "representative"

This strategy shows when the model for 80% of the data doesn't work well for the remaing 20% of the data, which could happen if

- The subsamples aren't 'representative' to start with
- The model is overly specific to 80% of the data

An Alternative Method: 80/20 Train-Test Split

$$\text{Root Mean Square Error} \quad \text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{m}(y_i - \hat{y}_i)^2}$$

is based on scoring how well a created model explains new data

$\rightarrow$ How good it is at predicting new data	$\rightarrow$ How well it generalizes to new data

Prediction (RMSE) doesn't care about confounding or multicollinearity

- If x_{1i} and x_{2i} are identical, the prediction $\hat{y}_i = x_{1i} + x_{2i} = 2x_{1i} + 0x_{2i}$ is the same
- (outcome) *Prediction* and (parameter) *Inference* are related but different exercises

If predictions are not generalizing to new data well, the model may be *overfit*

- it may be representing idiosyncratic spurious patterns of the model fitting data

$\rightarrow$ If predictions could be generally improved, then the model is said to be *underfit*

An Alternative Method: 80/20 Train-Test Split

Fit a model on 80% of the data → "score" the model on the remaining 20%

```r
n <- dim(heights)[1] # nrow(heights)
n_train <- as.integer(n*0.8)
n_test <- n - n_train
set.seed(130)
training_indices <-
  sample(1:n,size=n_train,replace=FALSE)
heights <- heights %>% rowid_to_column()
train <- heights %>%
  filter(rowid %in% training_indices)
test <- heights %>%
  filter(!(rowid %in% training_indices))
```

- There was not p-value evidence at the $\alpha = 0.05$ significance level for `lm(height~shoePrint+sex)`

```r
model <- lm(height~shoePrint, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

```
## [1] 8.189324
```

```r
model <- lm(height~sex, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

```
## [1] 6.941034
```

```r
model <- lm(height~shoePrint+sex, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

```
## [1] 7.989595
```

- But there is train-test evidence

An Alternative Method: 80/20 Train-Test Split

Fit a model on 80% of the data → "score" the model on the remaining 20%

```r
n <- dim(heights)[1] # nrow(heights)
n_train <- as.integer(n*0.8)
n_test <- n - n_train
set.seed(131)
training_indices <-
  sample(1:n,size=n_train,replace=FALSE)
#heights <- heights %>% rowid_to_column()
train <- heights %>%
  filter(rowid %in% training_indices)
test <- heights %>%
  filter(!(rowid %in% training_indices))
```

- There was not p-value evidence at the $\alpha = 0.05$ significance level for `lm(height~shoePrint+sex)`

```r
model <- lm(height~shoePrint, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

```
## [1] 5.500713
```

```r
model <- lm(height~sex, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

```
## [1] 5.637513
```

```r
model <- lm(height~shoePrint+sex, data=train)
yhat_test <- predict(model, newdata=test)
sqrt(mean((test$height-yhat_test)^2))
```

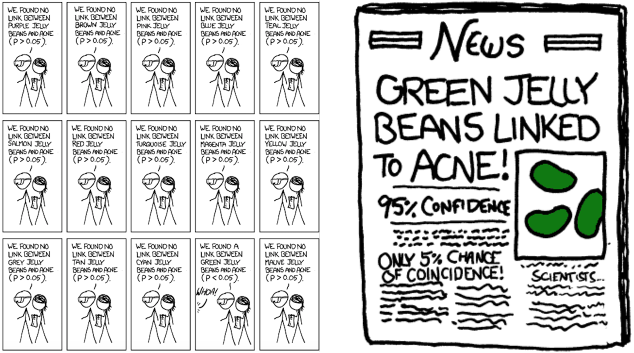
```
## [1] 5.765259
```

- But there is train-test evidence?

An Alternative Method: 80/20 Train-Test Split

Fit a model on 80% of the data → "score" the model on the remaining 20%

- The train-test method is a wonderful tool in LARGE data contexts
  - when there's enough data so the random train-test split isn't just "lucky"
- In its more advanced (data science) forms, train-test is a powerful model tuning tool



- Like Hypothesis Testing, it is subject to "random chance" (of the test-train split)
- Unlike Hypothesis Testing, it is based on observed out of sample generalizability, rather than tests based on modeling assumptions
- It's not about parameters or 'right' or 'wrong', but picking models predicting new data 'well'

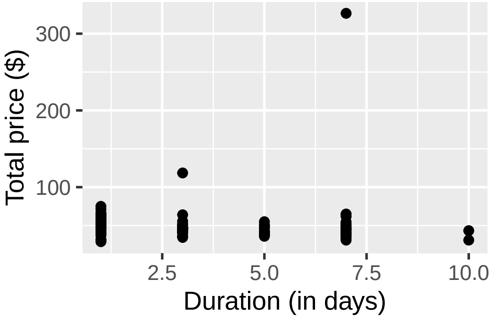
ebay Auctions of Mario Kart Games


```r
library(openintro)
glimpse(mariokart)
```

```
## Rows: 143
## Columns: 12
## $ id          <dbl> 150377422259, 26048337б
## $ duration    <int> 3, 7, 3, 3, 1, 3, 1, 1,
## $ n_bids      <int> 20, 13, 16, 18, 20, 19,
## $ cond        <fct> new, used, new, new, ne
## $ start_pr    <dbl> 0.99, 0.99, 0.99, 0.99,
## $ ship_pr     <dbl> 4.00, 3.99, 3.50, 0.00,
## $ total_pr    <dbl> 51.55, 37.04, 45.50, 44
## $ ship_sp     <fct> standard, firstClass, f
## $ seller_rate <int> 1580, 365, 998, 7, 820,
## $ stock_photo <fct> yes, yes, no, yes, yes,
## $ wheels      <int> 1, 1, 1, 1, 2, 0, 0, 2,
## $ title       <fct> "~~ Wii MARIO KART &amp
```

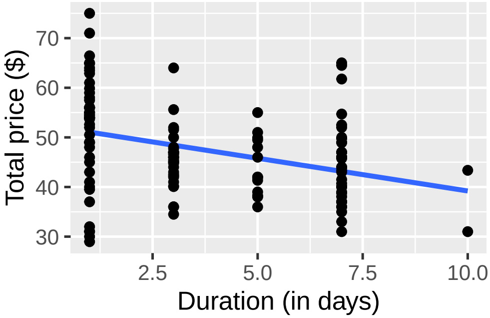
Is auction length associated with the selling price?

```r
# {r, fig.width=3, fig.height=2}
mariokart %>% ggplot(aes(x=duration,
                          y=total_pr)) +
  geom_point() + labs(x="Duration (in days)",
                       y="Total price ($)")
```




```r
# the outliers are multi-item purchases
```

```r
mariokart %>% filter(total_pr < 100) %>%
  ggplot(aes(x=duration, y=total_pr)) +
  geom_smooth(method=lm, se=FALSE) +
  geom_point() + labs(x="Duration (in days)",
                       y="Total price ($)")
```




```r
# Moderate negative linear(?) association?
```

Moderate Negative Linear(?) Association?

```r
mariokart2 <- mariokart %>%
  filter(total_pr < 100)
model1 <- lm(total_pr~duration,
             data=mariokart2)
summary(model1) %>% tidy() %>%
  select(-statistic) %>%
  rename(se=std.error)
```

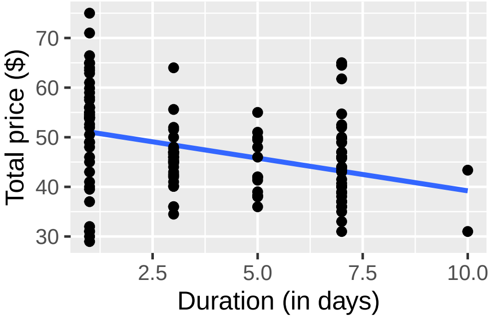
```
## # A tibble: 2 x 4
##   term        estimate    se  p.value
##   <chr>          <dbl> <dbl>    <dbl>
## 1 (Intercept)    52.4  1.26  3.01e-80
## 2 duration      -1.32 0.277  4.87e- 6
```

```r
summary(model1)$r.squared
```

```
## [1] 0.1399937
```

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i}$$
$$\approx 52.4 - 1.3 \times \text{duration}_i$$

```r
mariokart2 %>%
  ggplot(aes(x=duration, y=total_pr)) +
  geom_smooth(method=lm, se=FALSE) +
  geom_point() + labs(x="Duration (in days)",
                      y="Total price ($)")
```

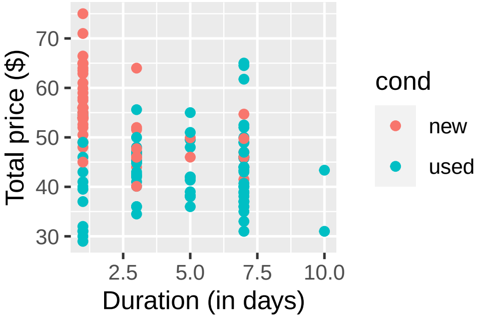


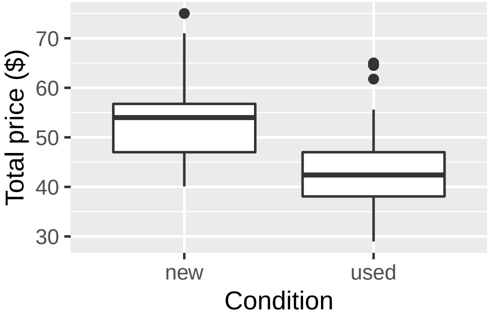

```r
# Moderate negative linear(?) association?
```

Another Variable: `cond(ition) new/used`

```r
mariokart2 %>% ggplot(aes(
  x=duration, y=total_pr, color=cond)) +
  geom_point() + labs(x="Duration (in days)",
                      y="Total price ($)")
```

```r
mariokart2 %>%
  ggplot(aes(x=cond, y=total_pr)) +
  geom_boxplot() +
  labs(x="Condition", y="Total price ($)")
```





Another Variable: `cond(ition) new/used`

```r
model2 <- lm(total_pr~cond,
             data=mariokart2)
summary(model2) %>% tidy() %>%
  select(-statistic) %>%
  rename(se=std.error)
```

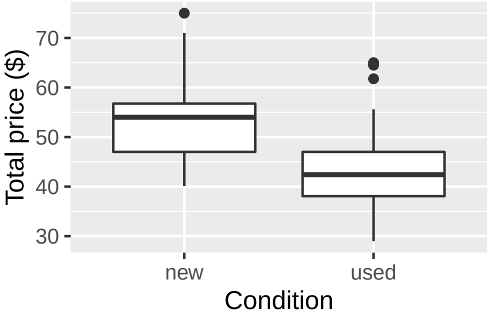
```
## # A tibble: 2 x 4
##   term        estimate    se  p.value
##   <chr>          <dbl> <dbl>    <dbl>
## 1 (Intercept)     53.8 0.960 2.73e-97
## 2 condused       -10.9 1.26  1.06e-14
```

```r
summary(model2)$r.squared
```

```
## [1] 0.3505528
```

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1(x_{2i} = \text{used})$$
$$\approx 53.7 - 10.9 \times \mathrm{I}(\text{cond}_i = \text{used})$$

```r
mariokart2 %>%
  ggplot(aes(x=cond, y=total_pr)) +
  geom_boxplot() +
  labs(x="Condition", y="Total price ($)")
```



Two Variables: `duration` + `cond(ition)` `new/used`

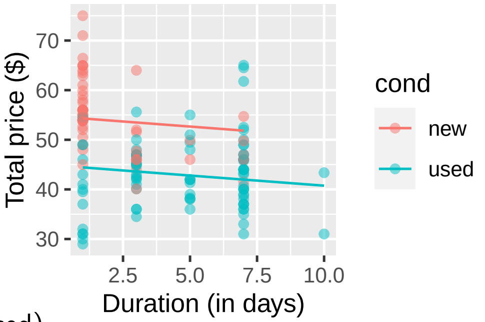
```r
model3 <- lm(total_pr~duration+cond,
             data=mariokart2)
summary(model3) %>% tidy() %>%
  select(-statistic) %>%
  rename(se=std.error)
```

```
## # A tibble: 3 x 4
##   term        estimate    se  p.value
##   <chr>          <dbl> <dbl>    <dbl>
## 1 (Intercept)   54.7   1.14  7.02e-88
## 2 duration      -0.409 0.273 1.37e- 1
## 3 condused      -9.87  1.43  1.66e-10
```

```r
summary(model3)$r.squared
```

```
## [1] 0.3609101
```

```r
library(broom); mariokart2 %>% ggplot(aes(
  x=duration, y=total_pr, color=cond)) +
  geom_point(alpha=0.5) +# <- SEE OVERLAPPING
  geom_line(data=augment(model3),     # POINTS
            aes(y=.fitted, colour=cond)) +
  labs(x="Duration (in days)", y="Total price
```



$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \hat{\beta}_2 \mathrm{I}(x_{2i} = \text{used})$$

$$\approx 53.7 - 0.4 \times \texttt{duration}_i - 9.9 \times \mathrm{I}(\texttt{cond}_i = \text{used})$$

Two Variables: `duration` + `cond(ition) new/used`

$$\hat{y}_i = \begin{cases} (\hat{\beta}_0 + \hat{\beta}_2) + x_{1i}\hat{\beta}_1 & \text{if cond}_i = \text{used} \\ \hat{\beta}_0 + x_{1i}\hat{\beta}_1 \quad \text{[baseline]} & \text{otherwise} \end{cases}$$

$$= \begin{cases} (53.7 - 9.9) - 0.4 \times \text{duration}_i & \text{if cond}_i = \text{used} \\ 53.7 - 0.4 \times \text{duration}_i \quad \texttt{[baseline]} & \text{otherwise} \end{cases}$$

```r
model3 <- lm(total_pr~duration+cond, data=mariokart2)
summary(model3)$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 54.7058693  1.1418347 47.910500 7.021551e-88
## duration    -0.4087132  0.2732979 -1.495486 1.370710e-01
## condused    -9.8709545  1.4291789 -6.906731 1.663239e-10
```

```r
model3
```

```
##
## Call:
## lm(formula = total_pr ~ duration + cond, data = mariokart2)
##
## Coefficients:
## (Intercept)     duration     condused
##     54.7059      -0.4087      -9.8710
```

```r
library(broom); augment(model3) %>% print(n=6)
```

```
## # A tibble: 141 x 9
##   total_pr duration cond  .fitted .resid   .hat .sigma   .cooksd .std.resid
##      <dbl>    <int> <fct>   <dbl>  <dbl>  <dbl>  <dbl>     <dbl>      <dbl>
## 1     51.6        3 new      53.5  -1.93 0.0177   7.36 0.000422     -0.265
## 2     37.0        7 used     42.0  -4.93 0.0189   7.35 0.00296      -0.679
## 3     45.5        3 new      53.5  -7.98 0.0177   7.33 0.00721      -1.10
## 4     44          3 new      53.5  -9.48 0.0177   7.32 0.0102       -1.30
## 5     71          1 new      54.3  16.7  0.0193   7.22 0.0346        2.30
## 6     45          3 new      53.5  -8.48 0.0177   7.33 0.00814      -1.17
## # ... with 135 more rows
```

Interactions: `duration` × `cond(ition) new/used`

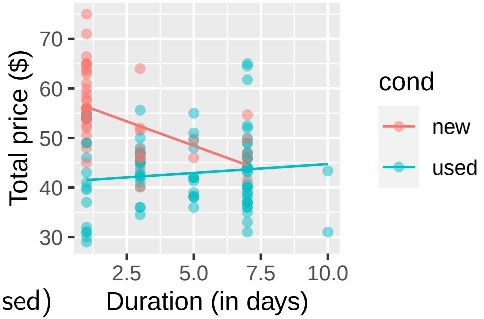
```r
model4 <- lm(total_pr~duration*cond,
             data=mariokart2)
summary(model4) %>% tidy() %>%
  select(-statistic, -std.error)
```

```
## # A tibble: 4 x 3
##   term              estimate  p.value
##   <chr>                <dbl>    <dbl>
## 1 (Intercept)          58.3  5.83e-81
## 2 duration             -1.97 2.34e- 5
## 3 condused            -17.1  1.01e-12
## 4 duration:condused     2.32 4.10e- 5
```

```r
summary(model3)$r.squared
```

```
## [1] 0.3609101
```

```r
library(broom); mariokart2 %>% ggplot(aes(
  x=duration, y=total_pr, color=cond)) +
  geom_point(alpha=0.5) +
  geom_line(data=augment(model4),
            aes(y=.fitted, colour=cond)) +
  labs(x="Duration (in days)", y="Total price
```



$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \hat{\beta}_2 \mathrm{I}(x_{2i} = \text{used}) + \hat{\beta}_3 x_{1i}\mathrm{I}(x_{2i} = \text{used})$$

$$\approx 58.3 - 2.0 \times \texttt{duration}_i - 17.1 \times \mathrm{I}(\texttt{cond}_i = \text{used}) + 2.3 \times \texttt{duration}_i \times \mathrm{I}(\texttt{cond}_i = \text{used})$$

Interactions: `duration` × `cond(ition) new/used`

$$\hat{y}_i = \begin{cases} \hat\beta_0 + x_{1i}\hat\beta_1 \quad \text{[baseline]} & \text{if } \text{cond}_i = \text{new} \\ (\hat\beta_0 + \hat\beta_2) + x_{1i}(\hat\beta_1 + \hat\beta_3) & \text{if } \text{cond}_i = \text{used} \end{cases}$$

$$= \begin{cases} 58.3 - 2.0x_{1i} \quad \text{[baseline]} & \text{if } \text{cond}_i = \text{new} \\ (58.3 - 17.1) + (2.3 - 2.0)x_{1i} & \text{if } \text{cond}_i = \text{used} \end{cases}$$

```r
model4 <- lm(total_pr~duration*cond, data=mariokart2)
summary(model4)$coefficients
```

```
##                     Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)        58.268226  1.3664729 42.641332 5.832075e-81
## duration           -1.965595  0.4487799 -4.379865 2.341705e-05
## condused          -17.121924  2.1782581 -7.860374 1.013608e-12
## duration:condused   2.324563  0.5483731  4.239016 4.101561e-05
```

Model Comparison

x_{1i}: auction diration of i^{th} item	x_{2i}: condition of i^{th} item

- Model 1: $y_i = \beta_0 + \beta_1 x_{1i} + \epsilon_i$
- Model 2: $y_i = \beta_0 + \beta_1 \mathsf{I}(x_{1i} = \text{used}) + \epsilon_i$
- Model 3: $y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 \mathsf{I}(x_{2i} = \text{used}) + \epsilon_i$
- Model 4: $y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 \mathsf{I}(x_{2i} = \text{used}) + \beta_3 x_{1i} \mathsf{I}(x_{2i} = \text{used}) + \epsilon_i$

Model Comparison

x_{1i}: auction diration of i^{th} item	x_{2i}: condition of i^{th} item

- Model 1 Fit: $\hat{y}_i \approx 52.4 - 1.32x_{1i}$
- Model 2 Fit: $\hat{y}_i \approx 53.8 - 10.9 \times \mathrm{I}(x_{1i} = \text{used})$
- Model 3 Fit: $\hat{y}_i \approx 54.7 - 0.4x_{1i} - 9.9 \times \mathrm{I}(x_{2i} = \text{used})$
- Model 4 Fit: $\hat{y}_i \approx 58.3 - 2.0x_{1i} - 17.1 \times \mathrm{I}(x_{2i} = \text{used}) + 2.3x_{1i} \times \mathrm{I}(x_{2i} = \text{used})$

```r
summary(model1)$r.squared # 0.1399937
summary(model2)$r.squared # 0.3505528
summary(model3)$r.squared # 0.3609101
summary(model4)$r.squared # 0.435015
```

What is each part of this code doing?

```
set.seed(130);
n <- nrow(mariokart2)
training_indices <- sample(1:n, size=round(0.8*n))
mariokart2 <- mariokart2 %>% rowid_to_column()
train <- mariokart2 %>% filter(rowid %in% training_indices)
y_train <- train$total_pr
test <- mariokart2 %>% filter(!(rowid %in% training_indices))
y_test <- test$total_pr
model1_train <- lm(total_pr ~ duration, data = train)
model2_train <- lm(total_pr ~ cond, data = train)
model3_train <- lm(total_pr ~ duration + cond, data=train)
model4_train <- lm(total_pr ~ duration * cond, data=train)
```

Model Comparison

```r
yhat_model1_test <- predict(model1_train, newdata=test)
yhat_model2_test <- predict(model2_train, newdata=test)
yhat_model3_test <- predict(model3_train, newdata=test)
yhat_model4_test <- predict(model4_train, newdata=test)
model1_test_RMSE <- sqrt(mean((y_test-yhat_model1_test)^2))
model2_test_RMSE <- sqrt(mean((y_test-yhat_model2_test)^2))
model3_test_RMSE <- sqrt(mean((y_test-yhat_model3_test)^2))
model4_test_RMSE <- sqrt(mean((y_test-yhat_model4_test)^2))
yhat_model1_train <- predict(model1_train, newdata=train)
yhat_model2_train <- predict(model2_train, newdata=train)
yhat_model3_train <- predict(model3_train, newdata=train)
yhat_model4_train <- predict(model4_train, newdata=train)
model1_train_RMSE <- sqrt(mean((y_train-yhat_model1_train)^2))
model2_train_RMSE <- sqrt(mean((y_train-yhat_model2_train)^2))
model3_train_RMSE <- sqrt(mean((y_train-yhat_model3_train)^2))
model4_train_RMSE <- sqrt(mean((y_train-yhat_model4_train)^2))
```

Model Comparison

```r
mytable <- tibble(Model = c("Model 1", "Model 2", "Model 3", "Model 4"),
  RMSE_testdata = c(model1_test_RMSE, model2_test_RMSE, model3_test_RMSE, model4_test_RMSE),
  RMSE_traindata = c(model1_train_RMSE, model2_train_RMSE, model3_train_RMSE, model4_train_RMSE))
mytable %>% rowid_to_column() %>% ggplot() + labs(x="Model", y="RMSE") +
  geom_point(aes(x=rowid, y=RMSE_testdata, color="Test")) +
  geom_point(aes(x=rowid, y=RMSE_traindata, color="Train"))
```