SDS358: Applied Regression Analysis

Day 12: Multiple Regression PtII

Dr. Michael J. Mahometa

Agenda for Today:

- MLR recap
  - Interpretation (overall model and individual coefficients)
  - What's the same and what's different
- More differences
- A "new" model…
- Standardized betas
- Breaking down our R^2
- Adjusted R^2
- Tolerance and V.I.F. (Multicollinearity)
- Prediction
- Procedural steps of the MLR
  - Diagnostics
  - Reporting

What ELSE changes?

- Our individual parameter estimates (inference)
- Our estimate assumptions of b_k still holds (its' an unbiased estimator).
- *But* our $s.e.(b_k)$ changes…to incorporate the effect of the *other* independent variables.

$$s.e.(b_k) = \frac{\hat{\sigma}_e}{\sqrt{\Sigma(x_k - \bar{x})^2 \times (1 - R_k^2)}}$$

Where R_k^2 is the proportion of variance when x_k is regressed on the k-1 other predictors from the model. (k = 1, 2, …, k)

What ELSE changes?

```
summary(lm(HoursStudied ~ Absences, data=exams))$r.squared
```

```
## [1] 0.3089373
```

```
tol_hs <- (1 - summary(lm(HoursStudied ~ Absences, data=exams))$r.squared)
tol_hs
```

```
## [1] 0.6910627
```

```
SSX <- sum((exams$HoursStudied - mean(exams$HoursStudied, na.rm=TRUE))^2)
summary(e_mod)$sigma / sqrt(SSX * tol_hs)
```

```
## [1] 1.366619
```

What ELSE changes?

```
summary(e_mod)
```

```
##
## Call:
## lm(formula = ExamTotal ~ Absences + HoursStudied, data = exams)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.5937  -8.5403   0.4312   9.0768  25.0234
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   89.1497    16.9747   5.252  5.4e-05 ***
## Absences      -0.9733     1.2275  -0.793    0.438
## HoursStudied   2.9448     1.3666   2.155    0.045 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.85 on 18 degrees of freedom
## Multiple R-squared:  0.3657, Adjusted R-squared:  0.2952
## F-statistic: 5.189 on 2 and 18 DF,  p-value: 0.01662
```


And our Confidence of the Slope

```
confint(e_mod)
```

```
##                     2.5 %     97.5 %
## (Intercept)  53.48726987 124.812130
## Absences     -3.55211392   1.605495
## HoursStudied  0.07360786   5.815928
```

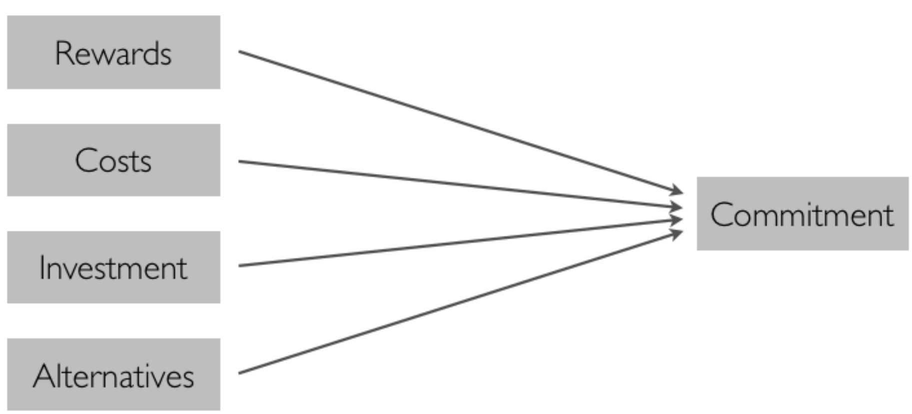
But…

- How can we establish "predictor importance"?

```
summary(e_mod)
```

```
##
## Call:
## lm(formula = ExamTotal ~ Absences + HoursStudied, data = exams)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.5937  -8.5403   0.4312   9.0768  25.0234
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   89.1497    16.9747   5.252  5.4e-05 ***
## Absences      -0.9733     1.2275  -0.793    0.438
## HoursStudied   2.9448     1.3666   2.155    0.045 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.85 on 18 degrees of freedom
## Multiple R-squared:  0.3657, Adjusted R-squared:  0.2952
## F-statistic: 5.189 on 2 and 18 DF,  p-value: 0.01662
```


What if…



What if…

- But what if…

```
##
## Call:
## lm(formula = commitment ~ reward + cost + investment + alternative,
##     data = commit)
##
## Coefficients:
## (Intercept)       reward         cost   investment  alternative
##    22.94662      0.28571     -0.05836      0.43767     -0.21741
```


What if…

```
summary(c_mod)
```

```
##
## Call:
## lm(formula = commitment ~ reward + cost + investment + alternative,
##     data = commit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.251  -3.284   1.895   4.038  11.238
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 22.94662    8.80940   2.605   0.0126 *
## reward       0.28571    0.27516   1.038   0.3049
## cost        -0.05836    0.12512  -0.466   0.6432
## investment   0.43767    0.18122   2.415   0.0201 *
## alternative -0.21741    0.04635  -4.690 2.77e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.351 on 43 degrees of freedom
## Multiple R-squared:  0.645,  Adjusted R-squared:  0.612
## F-statistic: 19.53 on 4 and 43 DF,  p-value: 3.179e-09
```

Why is this so?

- Independents and the Dependent are on *different* scales

```
library(psych)
vars <- c("commitment", "reward", "cost", "investment", "alternative")
describe(select(commit, one_of(vars)))
```

```
##             vars  n  mean    sd median trimmed   mad min max range  skew
## commitment     1 48 27.71 10.20   32.0   29.18  5.93   4  36    32 -1.21
## reward         2 48 26.65  5.05   27.5   27.18  5.19   9  33    24 -1.14
## cost           3 48 29.33  9.21   31.0   29.77  8.90  10  45    35 -0.44
## investment     4 48 19.75  7.10   22.0   20.23  5.93   3  30    27 -0.59
## alternative    5 48 45.00 23.75   43.0   44.12 19.27   5 100    95  0.40
```


How could you compare the coefficients?

- How could you fix this *scale* issue?

How could you compare the coefficients?

- Remove the scale…

```
commit$s_commitment <- scale(commit$commitment)
commit$s_reward <- scale(commit$reward)
commit$s_cost <- scale(commit$cost)
commit$s_investment <- scale(commit$investment)
commit$s_alternative <- scale(commit$alternative)
```


How could you compare the coefficients?

- Remove the scale…

```
##               vars  n mean sd median trimmed  mad   min  max range  skew
## s_commitment     1 48    0  1   0.42    0.14 0.58 -2.33 0.81  3.14 -1.21
## s_reward         2 48    0  1   0.17    0.10 1.03 -3.49 1.26  4.75 -1.14
## s_cost           3 48    0  1   0.18    0.05 0.97 -2.10 1.70  3.80 -0.44
## s_investment     4 48    0  1   0.32    0.07 0.84 -2.36 1.44  3.80 -0.59
## s_alternative    5 48    0  1  -0.08   -0.04 0.81 -1.68 2.32  4.00  0.40
```

How could you compare the coefficients?

- Remove the scale…

```
c_mod_std <- lm(s_commitment ~ s_reward + s_cost + s_investment + s_alternative,
                data = commit)
summary(c_mod_std)
```


How could you compare the coefficients?

- Remove the scale…

```
##
## Call:
## lm(formula = s_commitment ~ s_reward + s_cost + s_investment +
##     s_alternative, data = commit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3977 -0.3221  0.1859  0.3961  1.1022
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.236e-16  8.991e-02   0.000   1.0000
## s_reward       1.415e-01  1.363e-01   1.038   0.3049
## s_cost        -5.273e-02  1.130e-01  -0.466   0.6432
## s_investment   3.048e-01  1.262e-01   2.415   0.0201 *
## s_alternative -5.064e-01  1.080e-01  -4.690 2.77e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6229 on 43 degrees of freedom
## Multiple R-squared:  0.645,  Adjusted R-squared:  0.612
## F-statistic: 19.53 on 4 and 43 DF,  p-value: 3.179e-09
```

How could you compare the coefficients?

```
round(summary(c_mod_std)$coef, 4)
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.0000     0.0899  0.0000   1.0000
## s_reward        0.1415     0.1363  1.0383   0.3049
## s_cost         -0.0527     0.1130 -0.4665   0.6432
## s_investment    0.3048     0.1262  2.4151   0.0201
## s_alternative  -0.5064     0.1080 -4.6904   0.0000
```


How could you compare the coefficients?

- Remove the scale…
- These results are called "*Standardized*" betas (b'_k).

$$b'_k = b_k \times \frac{S_{xk}}{S_y}$$

```
lmBeta(c_mod)
```

```
##      reward        cost  investment alternative
##   0.1415330  -0.0527312   0.3047586  -0.5064127
```

```
round(c_mod_std$coef, 4)
```

```
##   (Intercept)      s_reward        s_cost  s_investment s_alternative
##        0.0000        0.1415       -0.0527        0.3048       -0.5064
```

Wait!

- We like `lmBeta()`.
- It's easier…
- *And* it's based on the model data, not the dataset *submitted* to the model.

- What's the difference?

```
head(commit)[1:6]
```

```
## # A tibble: 6 x 6
##   SimpleID commitment reward  cost investment alternative
##      <int>      <int>  <int> <int>      <int>       <int>
## 1        1         34     25    23         19          30
## 2        2         32     27    25         28          34
## 3        3         34     24    37         25          37
## 4        4         26     22    32         22          52
## 5        5          4     25    18          3         100
## 6        6         31     30    38         26          34
```


Our Comittment Model:

I made a little change to the data…

```
commit2 <- commit %>%
  mutate(alternative = case_when(SimpleID %in% c(1:3) ~ NA_integer_,
                                 TRUE ~ alternative))
head(commit2)[1:6]
```

```
## # A tibble: 6 x 6
##   SimpleID commitment reward  cost investment alternative
##      <int>      <int>  <int> <int>      <int>       <int>
## 1        1         34     25    23         19          NA
## 2        2         32     27    25         28          NA
## 3        3         34     24    37         25          NA
## 4        4         26     22    32         22          52
## 5        5          4     25    18          3         100
## 6        6         31     30    38         26          34
```

```
new_c_mod <- lm(commitment ~ reward + cost + investment + alternative, data = commit2) #new data
```

Our Comittment Model:

```
##
## Call:
## lm(formula = commitment ~ reward + cost + investment + alternative,
##     data = commit2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.119  -3.857   1.790   4.123  11.441
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 21.94963    9.36098   2.345   0.0241 *
## reward       0.31466    0.29405   1.070   0.2910
## cost        -0.05751    0.13181  -0.436   0.6649
## investment   0.43184    0.19447   2.221   0.0321 *
## alternative -0.21296    0.04800  -4.436    7e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.529 on 40 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.6434, Adjusted R-squared:  0.6078
## F-statistic: 18.05 on 4 and 40 DF,  p-value: 1.53e-08
```


Multiple Regression Side Step

- "Listwise deletion": when multiple independents are in the model, if a row is missing one of those independents, then the *whole row* is **not** included in the analysis.

Standardized Betas

- Effectively remove the scale of all variables.
- Good to add to the "story" of the model…
- Remove the scale…
- These results are called "*Standardized*" betas b'_k

$$b'_k = b_k \times \frac{S_{xk}}{S_y}$$

- `lmBeta()` does this for the *model data*, not the data *submitted* to the model.

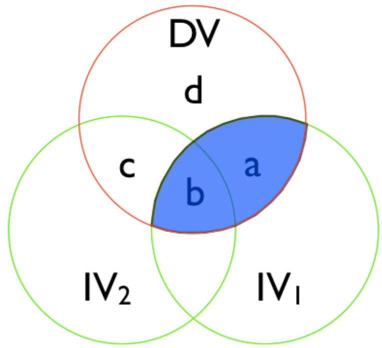
```
lmBeta(new_c_mod)
```

```
##      reward        cost  investment alternative
##  0.15688861 -0.05174359  0.29748052 -0.49726886
```


Breaking down our R^2

- R^2 is great. It tells us about the Overall Model.
- But what about x_1 and x_2?
- How much of R^2 is due to x_1 and how much is due to x_2?

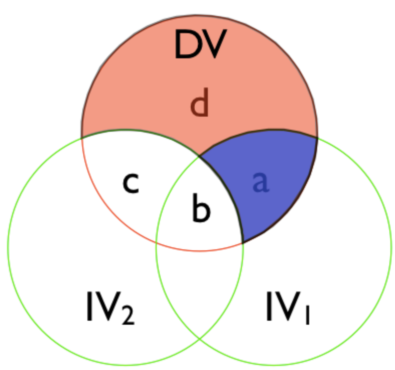
Our "Standard" Correlation:



$$r^2 = \frac{a+b}{a+b+c+d}$$

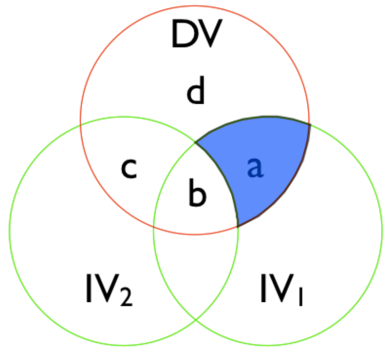
Our "Partial" Correlation:

- Partial: How much of the Y variance *which is not estimated by the other IVs in the equation* is estimated by this variable?



$$pr^2 = \frac{a}{a+d}$$

Our "Part" (Semipartial) Correlation:



$$sr^2 = \frac{a}{a + b + c + d}$$

So, how do we do (calculate) these?

Semipartial

$$sr_k = b'_k \times \sqrt{(1 - R_k^2)}$$

- Where R_k^2 is the proportion of variance when X_k is regressed on the k-1 other predictors from the model. (k = 1, 2, …, k)

Our "Part" (Semipartial) Correlation:

In R

```
pCorr(e_mod)[3]
```

```
##               Part_Corr
## Absences     -0.1488533
## HoursStudied  0.4045010
```


Our "Part" (Semipartial) Correlation:

In R

- To interpret, square…

```
pCorr(e_mod)[3]^2
```

```
##               Part_Corr
## Absences     0.02215731
## HoursStudied 0.16362104
```

```
pCorr(e_mod)[3:4]
```

```
##               Part_Corr Part_Corr_sq
## Absences     -0.1488533   0.02215731
## HoursStudied  0.4045010   0.16362104
```

Our "Part" (Semipartial) Correlation:

In R

- To interpret, square…

$$sr_k^2 = R_y^2 - R_{y-k}^2$$

Our "Part" (Semipartial) Correlation:

In R

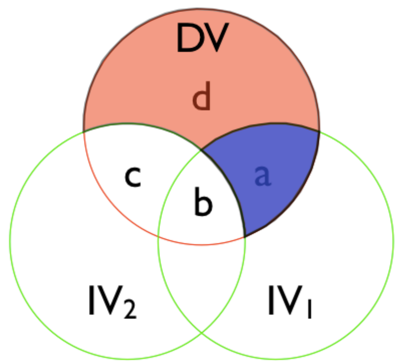
- To interpret, square…

```
Ry <- summary(e_mod)$r.squared
Ryk <- summary(lm(ExamTotal ~ HoursStudied, exams))$r.squared
Ry - Ryk
```

```
## [1] 0.02215731
```

Our "Partial" Correlation:

Visually:



So, how do we do (calculate) these?

Partial

$$pr_k = \frac{sr_k}{\sqrt{1 - R^2_{y-k}}}$$

Our "Partial" Correlation:

In R

```
pCorr(e_mod)
```

```
##              Partial_Corr Partial_Corr_sq  Part_Corr Part_Corr_sq
## Absences       -0.1837175      0.03375213 -0.1488533   0.02215731
## HoursStudied    0.4528306      0.20505554  0.4045010   0.16362104
```


Adjusted R^2

- Our R^2 for the model is an *overall* proportion of variance accounted for…
- Could it ever be 100%?
- uh…yeah
- If the sample size n equals the number of b's in the model…

Adjusted R^2

- How do we fix this?

$$R^2_{\text{adj}} = R^2 - \frac{k(1 - R^2)}{n - k - 1}$$

or

$$R^2_{\text{adj}} = R^2 - \frac{k(1 - R^2)}{n - p}$$

Adjusted R^2

```
summary(e_mod)
```

```
##
## Call:
## lm(formula = ExamTotal ~ Absences + HoursStudied, data = exams)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.5937  -8.5403   0.4312   9.0768  25.0234
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   89.1497    16.9747   5.252  5.4e-05 ***
## Absences      -0.9733     1.2275  -0.793    0.438
## HoursStudied   2.9448     1.3666   2.155    0.045 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.85 on 18 degrees of freedom
## Multiple R-squared:  0.3657, Adjusted R-squared:  0.2952
## F-statistic: 5.189 on 2 and 18 DF,  p-value: 0.01662
```

Adjusted R^2

```
R_sq <- summary(e_mod)$r.squared
R_sq - ((2 * (1 - R_sq)) / (18))
```

```
## [1] 0.2952064
```


Multicolinearity

Assumptions:
- Independence: Error associated with each data point is independent of every other value.
- The population mean of e is 0.
- For a given value of x, the population variance of e is:

$$\sigma_e^2$$

- Homoscedasticity
- For a given value of x, e has a normal distribution.
- **NEW for MLR:** No *multicolinearity* of independents. (Tolerance and V.I.F.)

What's this "Tolerance" thing?

- Multiple regression models take into account the effect of *multiple* independent variables on a single outcome.
- But everything's related. It stands to reason that the *multiple* independent variables are also related to one another.
  - Our correlation matrix tells us this:

```
vars <- c("ExamTotal", "Absences", "HoursStudied")
cor(select(exams, one_of(vars)))
```

```
##              ExamTotal   Absences HoursStudied
## ExamTotal    1.0000000 -0.4495161    0.5861130
## Absences    -0.4495161  1.0000000   -0.5558213
## HoursStudied 0.5861130 -0.5558213    1.0000000
```


What's this "Tolerance" thing?

- We know R^2 is the proportion of variance accounted for…
- What if we knew the R^2 for a model that has b_k as the outcome (with all the *other* independents as predictors)?

```
summary(lm(Absences ~ HoursStudied, data=exams))$r.squared
```

```
## [1] 0.3089373
```

What's this "Tolerance" thing?

- *Tolerance* is simply defined as the remaining variance (left unexplained in b_k).

$$TOL_k = (1 - R_k^2)$$

Where R_k^2 is the proportion of variance when X_k is regressed on the k-1 other predictors from the model. (k = 1, 2, …, k)

```
1 - summary(lm(Absences ~ HoursStudied, data=exams))$r.squared
```

```
## [1] 0.6910627
```


What's this "Tolerance" thing?

- Closely related (*very* closely) is the Variance Inflation Factor (V.I.F.):

```
1 / (1 - summary(lm(Absences ~ HoursStudied, data=exams))$r.squared)
```

```
## [1] 1.447047
```

What's this "Tolerance" thing?

- Closely related (*very* closely) is the Variance Inflation Factor (V.I.F.):
- Think about our s.e.(b) for our SLR:

$$s.e.(b) = \frac{\hat{\sigma}_e}{\sqrt{\Sigma(x_i - \bar{x})^2}}$$

What's this "Tolerance" thing?

- Closely related (*very* closely) is the Variance Inflation Factor (V.I.F.):
- Think about our s.e.(b) for our SLR:

$$s.e.(b) = \frac{\hat{\sigma}_e}{\sqrt{\Sigma(x_i - \bar{x})^2}}$$

$$Var(b) = \frac{\hat{\sigma}_e^2}{\Sigma(x_i - \bar{x})^2}$$

What's this "Tolerance" thing?

$$Var(b_k) = \frac{\hat{\sigma}_e^2}{\Sigma(x_i - \bar{x})^2} \times \frac{1}{TOL_k}$$

$$Var(b_k) = \frac{\hat{\sigma}_e^2}{\Sigma(x_i - \bar{x})^2} \times \frac{1}{(1 - R_k^2)}$$

What's this "Tolerance" thing?

- Right *back* to the s.e.(b_k) for Multiple Regression

$$s.e.(b_k) = \frac{\hat{\sigma}_e}{\sqrt{\Sigma(x_i - \bar{x})^2 \times TOL_k}}$$

$$s.e.(b_k) = \frac{\hat{\sigma}_e}{\sqrt{\Sigma(x_i - \bar{x})^2 \times (1 - R_k^2)}}$$

- So, our s.e.(b_k)) is *directly* impacted by any collinearity of k_i with the other independents.

Multicollinearity

- Literally, how much the other independents in the model effect k_i .
- Cut offs or suggestions:
  - TOL < 0.2
  - V.I.F. > 5

Multicollinearity

In R

```
library(car)
vif(e_mod)
```

```
##     Absences HoursStudied
##     1.447047     1.447047
```

Multicollinearity

In R

```
1 / vif(e_mod)
```

```
##     Absences HoursStudied
##    0.6910627    0.6910627
```


Multicollinearity

In R: Comittment

```
##
## Call:
## lm(formula = commitment ~ reward + cost + investment + alternative,
##     data = commit2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.119  -3.857   1.790   4.123  11.441
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 21.94963    9.36098   2.345   0.0241 *
## reward       0.31466    0.29405   1.070   0.2910
## cost        -0.05751    0.13181  -0.436   0.6649
## investment   0.43184    0.19447   2.221   0.0321 *
## alternative -0.21296    0.04800  -4.436    7e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.529 on 40 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.6434, Adjusted R-squared:  0.6078
## F-statistic: 18.05 on 4 and 40 DF,  p-value: 1.53e-08
```

Multicollinearity

In R: Comittment

```
vif(new_c_mod)
```

```
##      reward        cost  investment alternative
##    2.411358    1.577576    2.013320    1.409589
```


Multicollinearity

In R: Comittment

- Left over…(tolerance)

```
1 / vif(new_c_mod)
```

```
##      reward        cost  investment alternative
##   0.4147041   0.6338841   0.4966920   0.7094266
```

Multicollinearity

In R: Comittment

- Each R^2_k

```
1 - (1 / vif(new_c_mod))
```

```
##      reward        cost  investment alternative 
##   0.5852959   0.3661159   0.5033080   0.2905734
```

```
#Really, this is happening:
r_mod <- lm(reward ~ cost + investment + alternative, data=commit2)
summary(r_mod)$r.squared
```

```
## [1] 0.5852959
```


Predictions with Multiple Regression

- Just like with SLR, we can make CI around the slope (using our new s.e.(b_k)):

```
confint(new_c_mod)
```

```
##                   2.5 %     97.5 %
## (Intercept)  3.03039114 40.8688703
## reward      -0.27963554  0.9089559
## cost        -0.32390414  0.2088795
## investment   0.03879548  0.8248926
## alternative -0.30997693 -0.1159348
```

Predictions with Multiple Regression

- *AND* we can also get both prediction and confidence intervals for the model as well.
- *BUT* we need to use values that *make sense* (why your text suggests centering).
- We **could** do this…and use `predict()`

```
mn_reward <- mean(commit2$reward, na.rm=TRUE)
mn_cost <- mean(commit2$cost, na.rm=TRUE)
mn_investment <- mean(commit2$investment, na.rm=TRUE)
new <- data.frame(reward = mn_reward, cost=mn_cost,
                  investment=mn_investment, alternative=30)
new
```

```
##     reward     cost investment alternative
## 1 26.64583 29.33333      19.75          30
```


BUT!!!

- In reality, you'll *most often* (as in 99% of the time) use the Confidence Interval—as we're most often interested in predicting the means.
- **AND**, what's wrong with using the `mean()` function here? (Think about our data…)

Predictions with Multiple Regression

- An alternative:

```
#remember OUR created data:
new
```

```
##     reward     cost investment alternative
## 1 26.64583 29.33333      19.75          30
```

```
#a better way
library(emmeans)
ref_grid(new_c_mod)
```

```
## 'emmGrid' object with variables:
##     reward = 26.733
##     cost = 29.4
##     investment = 19.467
##     alternative = 45.756
```


Predictions with Multiple Regression

- An alternative:

```
library(emmeans)
emmeans(new_c_mod, "alternative", at=list(alternative=30))
```

```
##  alternative   emmean       SE df lower.CL upper.CL
##           30 30.68857 1.232608 40 28.19738 33.17977
##
## Confidence level used: 0.95
```

To Sum Up Multiple Regression…

- MLR is just like SLR, but with more independents
  - Still have the same assumptions and outlier concerns
- These additional predictors force us to make actual adjustments:
  - How we interpret the parameters in the model
  - How t is calculated (with a new s.e.)
  - "Largest" effect on the outcome
  - An effect on R^2
    - New partitions
    - An adjustment for the number of predictors
  - Need to check "tolerance"
  - Prediction changes (taking others into account)
- Conclusions based on the models