normalize

Alison Gale

2/27/2021

Testing strategies for normalizing coefficients

Start by loading the six month data:

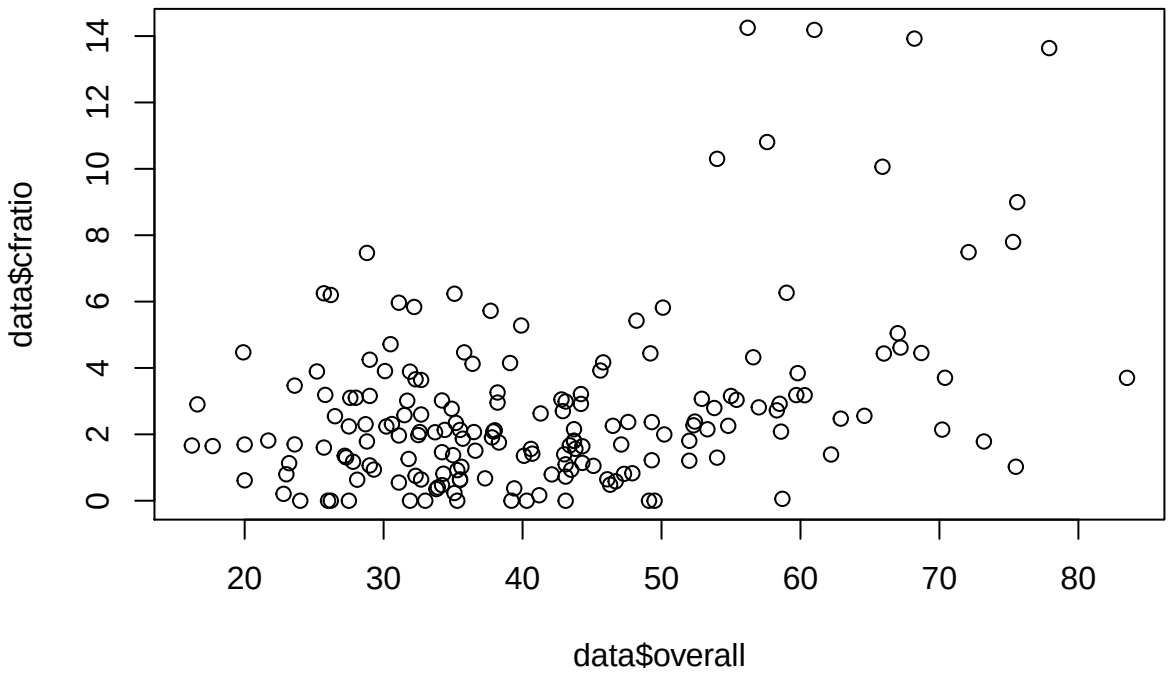
```
data <- read.csv(file = '../prepped_data/six_month_outlier_screened.csv')
```

As a baseline, run a regression on the overall GHSI score:

```
old_fit = lm(formula = cfratio ~ overall, data = data)
summary(old_fit)
```

```
##
## Call:
## lm(formula = cfratio ~ overall, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2044 -1.4438 -0.5437  0.7918 10.4436
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.33252    0.57139  -0.582    0.561
## overall      0.07363    0.01297   5.677 5.51e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.435 on 177 degrees of freedom
## Multiple R-squared:  0.154,  Adjusted R-squared:  0.1493
## F-statistic: 32.23 on 1 and 177 DF,  p-value: 5.51e-08
```

```
plot(data$overall, data$cfratio)
```



Suppose we run the following regression to get coefficients for our model:

```
fit = lm(formula = cfratio ~ prev_emergence_pathogens + early_detection + rapid_response + robust_healt
summary(fit)
```

```
##
## Call:
## lm(formula = cfratio ~ prev_emergence_pathogens + early_detection +
##     rapid_response + robust_health_sector + commitments + risk_environment,
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1533 -1.5581 -0.5003  0.8940 10.6471
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                0.3509193  1.0011137   0.351    0.726
## prev_emergence_pathogens   0.0412611  0.0227935   1.810    0.072 .
## early_detection            0.0049748  0.0129809   0.383    0.702
## rapid_response            -0.0007135  0.0210045  -0.034    0.973
## robust_health_sector       0.0206773  0.0234523   0.882    0.379
## commitments                0.0107225  0.0195402   0.549    0.584
## risk_environment          -0.0074782  0.0147077  -0.508    0.612
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.448 on 172 degrees of freedom
## Multiple R-squared:  0.169,  Adjusted R-squared:   0.14
## F-statistic: 5.829 on 6 and 172 DF,  p-value: 1.494e-05
```

Now we retrieve estimated values for the data:

```
estimates = fitted.values(fit)
```

And we normalize these values from 0 to 100:

```
norm = rescale(estimates, c(100, 0))
```

Add this back to the data frame:

```
data$new_overall = norm
```

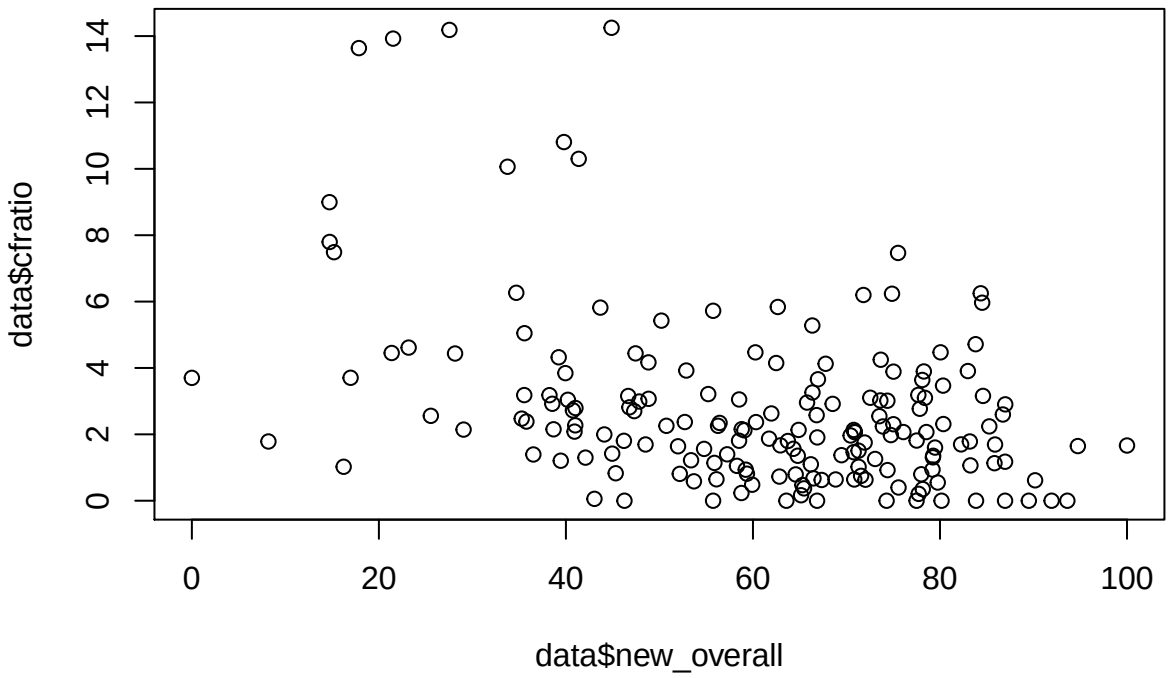
Finally, we run a new regression:

```
new_fit = lm(formula = cfratio ~ new_overall, data = data)
summary(new_fit)
```

```
##
## Call:
## lm(formula = cfratio ~ new_overall, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1533 -1.5581 -0.5003  0.8940 10.6471
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.067198   0.582816  10.410  < 2e-16 ***
## new_overall -0.054920   0.009155  -5.999 1.09e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.414 on 177 degrees of freedom
## Multiple R-squared:  0.169,  Adjusted R-squared:  0.1643
## F-statistic: 35.99 on 1 and 177 DF,  p-value: 1.091e-08
```

Plot the data:

```
plot(data$new_overall, data$cfratio)
```



In summary, we can get the correlation to go in the correct direction with the new score and the plots of the data look a little better. The improvement was more noticeable for cases-per-capita. Additionally the statistical significance of the intercept is higher. We might be able to improve this by adding in our confounding variables (possibly to replace the sub-components with weights closer to zero).