Datasets

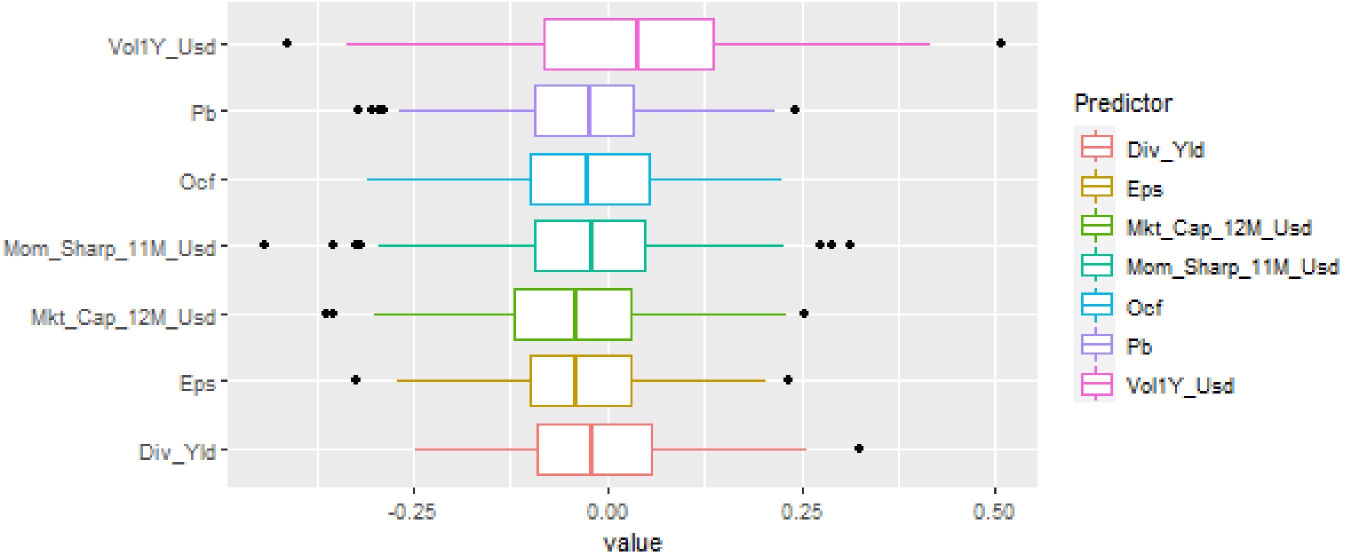
```r
load(here("Machine Learning for Factor Investing", "data_ml.RData"))
```

Know your Data

```r
features_short <- c("Div_Yld", "Eps", "Mkt_Cap_12M_Usd", "Mom_Sharp_11M_Usd", "Ocf", "Pb", "Vol

data_ml %>%
   dplyr::select(c(features_short), "R1M_Usd", "date") %>%
   group_by(date) %>%
   summarise_all(funs(cor(., R1M_Usd))) %>%
   dplyr::select(-R1M_Usd) %>%
   gather(key = Predictor, value = value, -date) %>%
   ggplot(aes(x = Predictor, y = value, color = Predictor)) +
      geom_boxplot(outlier.color = "black") + coord_flip() +
      theme(aspect.ratio = 0.6) + xlab(element_blank())
```

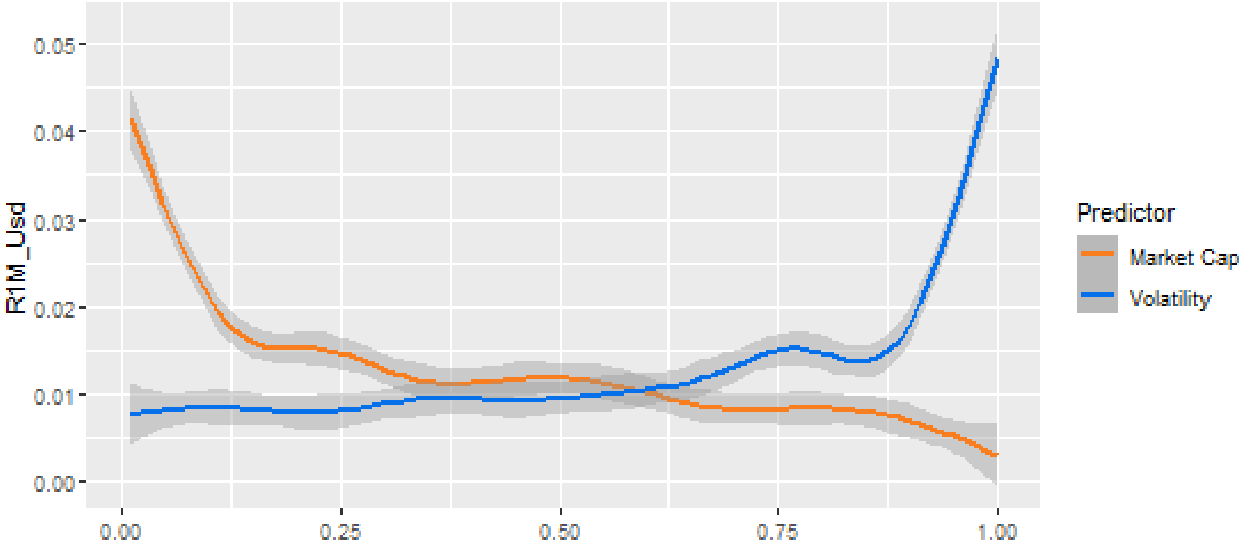
```
Note: Using an external vector in selections is ambiguous.
i Use `all_of(features_short)` instead of `features_short` to silence this message.
i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
This message is displayed once per session.
```

```
Warning: `funs()` is deprecated as of dplyr 0.8.0.
Please use a list of either functions or lambdas:

  # Simple named list:
  list(mean = mean, median = median)

  # Auto named with `tibble::lst()`:
  tibble::lst(mean, median)

  # Using lambdas
  list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
This warning is displayed once every 8 hours.
Call `lifecycle::last_warnings()` to see where this warning was generated.
```



```
data_ml %>%
   ggplot(aes(y = R1M_Usd)) +
      geom_smooth(aes(x = Mkt_Cap_12M_Usd, color = "Market Cap")) +
      geom_smooth(aes(x = Vol1Y_Usd, color = "Volatility")) +
   scale_color_manual(values = c("#F87E1F", "#0570EA")) +
   coord_fixed(10) +
   labs(color = "Predictor") + xlab(element_blank())
```

```
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Autocorrelation

```
features <- c("Advt_12M_Usd","Advt_3M_Usd","Advt_6M_Usd","Asset_Turnover","Bb_Yld","Bv","Capex_

autocorrs <- data_ml %>%
   dplyr::select(c("stock_id", features)) %>%
   gather(key = feature, value = value, -stock_id) %>%
   group_by(stock_id, feature) %>%
   summarise(acf = acf(value, lag.max = 1, plot = F)$acf[2])
```

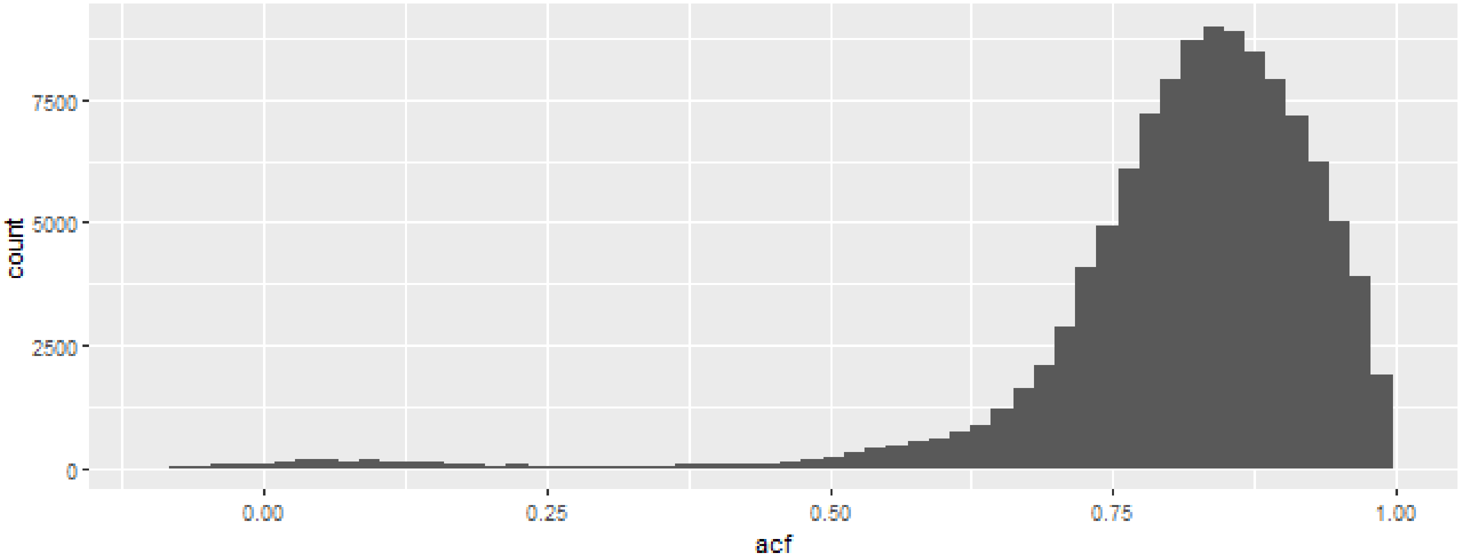
```
Note: Using an external vector in selections is ambiguous.
i Use `all_of(features)` instead of `features` to silence this message.
i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
This message is displayed once per session.
```

```
`summarise()` regrouping output by 'stock_id' (override with `.groups` argument)
```

```
autocorrs %>%
   ggplot(aes(x = acf)) + xlim(-0.1, 1) +
   geom_histogram(bins = 60)
```

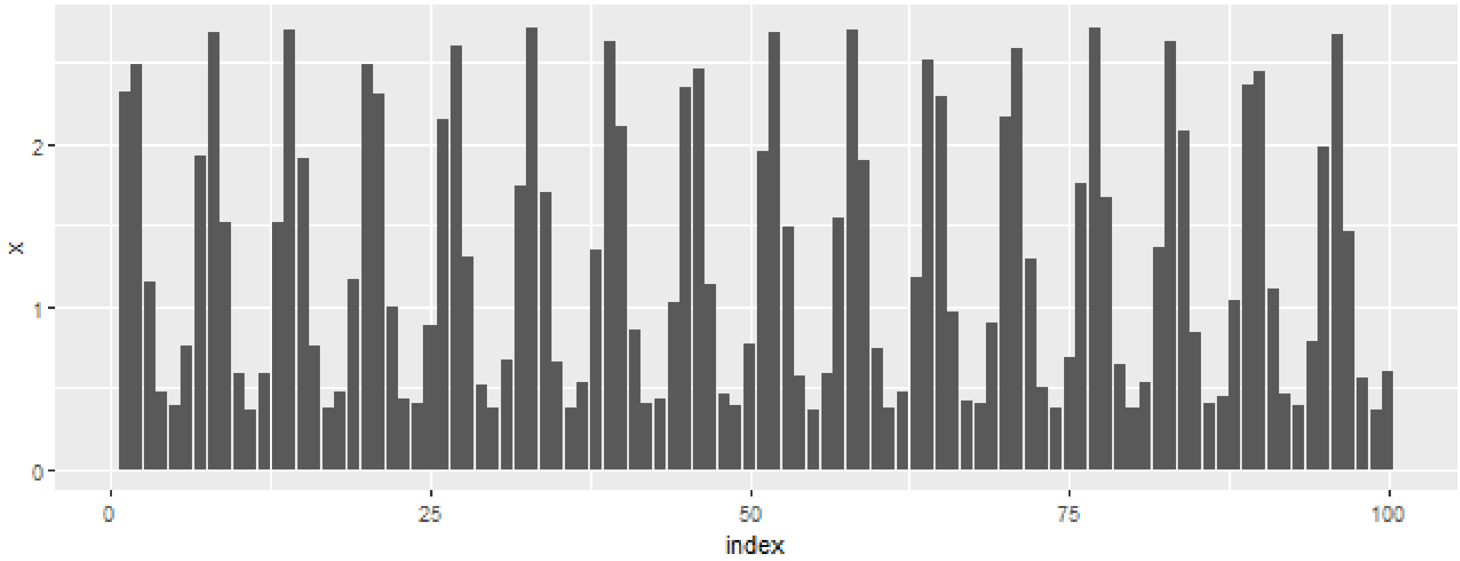
```
Warning: Removed 270 rows containing non-finite values (stat_bin).
```

```
Warning: Removed 2 rows containing missing values (geom_bar).
```



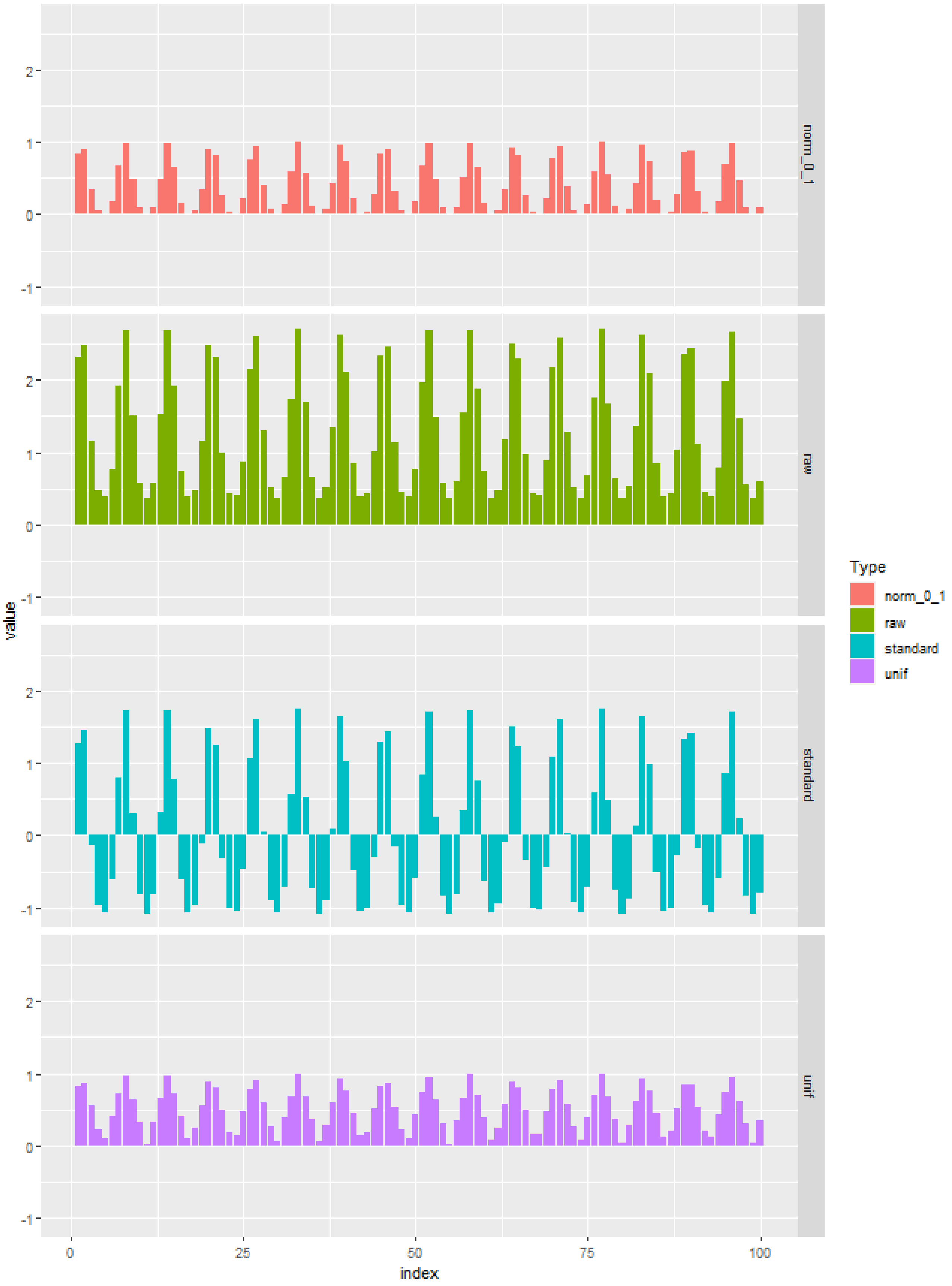
Impact of rescaling: graphical representation

```
Length <- 100                                # length of the sequence
x <- exp(sin(1:Length))                      # original data
data <- data.frame(index = 1:Length, x = x)  # convert to df

ggplot(data, aes(x = index, y = x)) + geom_bar(stat = "identity")
```



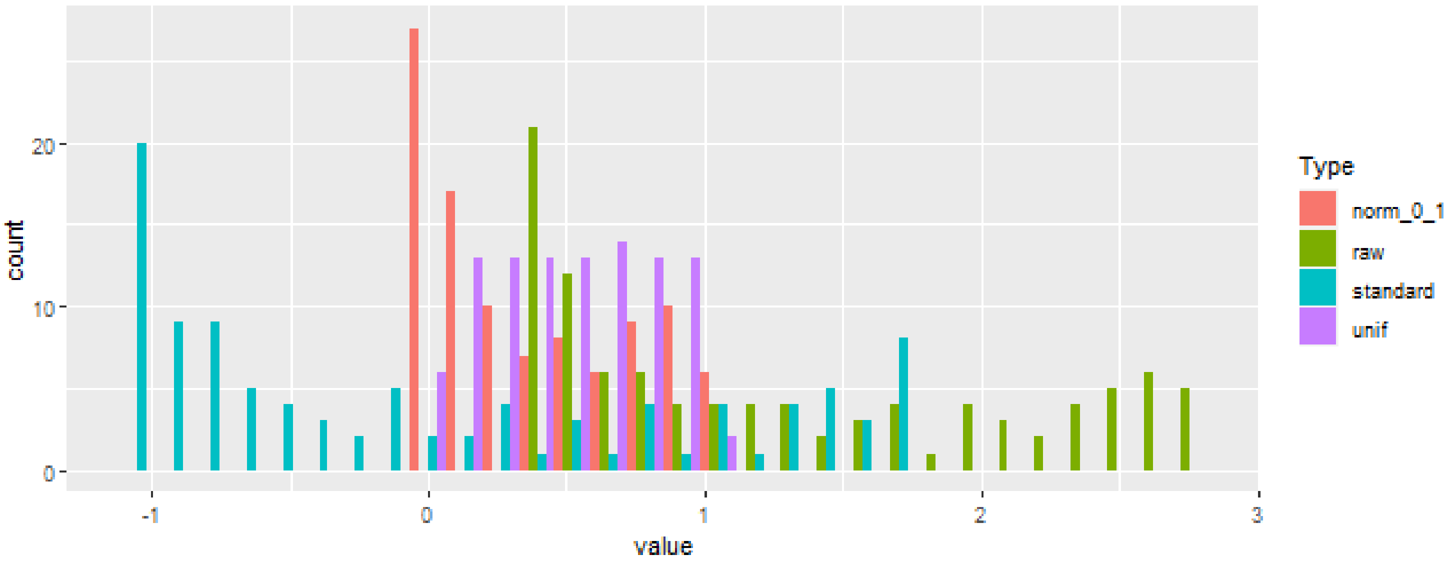
```
# uniformalises a vector
norm_unif <- function(v) {
   v <- v %>% as.matrix()
   return(ecdf(v)(v))
}

# function that uniformalises a vector
norm_0_1 <- function(v) {
   return((v-min(v))/(max(v)-min(v)))
}

data_norm <- data.frame(
   index = 1:Length,
```

```r
    raw = x,
    standard = (x - mean(x)) / sd(x),
    norm_0_1 = norm_0_1(x),
    unif = norm_unif(x)) %>%
    gather(key = Type, value = value, -index)

ggplot(data_norm, aes(x = index, y = value, fill = Type)) +
    geom_bar(stat = "identity") +
    facet_grid(Type~.)
```


value
2
1
0
-1
norm_0_1
raw
standard
unif
0
25
50
75
100
index
Type
norm_0_1
raw
standard
unif

```
ggplot(data_norm, aes(x = value, fill = Type)) +
   geom_histogram(position = "dodge")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
firm <- c(rep(1, 3), rep(2, 3), rep(3, 3))
date <- rep(c(1, 2, 3), 3)
cap <- c(10, 50, 100,
         15, 10, 15,
         200, 120, 80)

sample_data <- data.table(
   firm = firm,
   date = date,
   cap = cap
)

sample_data[, cap_0_1 := norm_0_1(cap), by = c("date")]
sample_data[, cap_u := norm_unif(cap), by = c("date")]

sample_data[, return := c(0.06, 0.01, -0.06,
                          -0.03, 0.00, 0.02,
                          -0.04, -0.02, 0.00)]
```

Impact of Rescaling

```
sample_data[date == 1]
```

```
   firm date cap    cap_0_1     cap_u return
```

Table 1: Regression output when the independent var. comes from min-max rescaling.

term	estimate	std.error	statistic	p.value
(Intercept)	0.0162778	0.0137351	1.185121	0.2746390
cap_0_1	-0.0497032	0.0213706	-2.325777	0.0529421

Table 2: Regression output when the independent var. comes from uniformization rescaling.

term	estimate	std.error	statistic	p.value
(Intercept)	0.06	0.0198139	3.028170	0.0191640
cap_u	-0.10	0.0275162	-3.634219	0.0083509

```
1:    1    1  10 0.00000000 0.3333333   0.06
2:    2    1  15 0.02631579 0.6666667  -0.03
3:    3    1 200 1.00000000 1.0000000  -0.04
```

```r
sample_data[date == 2]
```

```
   firm date cap   cap_0_1     cap_u return
1:    1    2  50 0.3636364 0.6666667   0.01
2:    2    2  10 0.0000000 0.3333333   0.00
3:    3    2 120 1.0000000 1.0000000  -0.02
```

```r
sample_data[date == 3]
```

```
   firm date cap   cap_0_1     cap_u return
1:    1    3 100 1.0000000 1.0000000  -0.06
2:    2    3  15 0.0000000 0.3333333   0.02
3:    3    3  80 0.7647059 0.6666667   0.00
```

```r
lm(return ~ cap_0_1, data = sample_data) %>%
   broom::tidy() %>%
   knitr::kable(caption = "Regression output when the independent var.
                comes from min-max rescaling.", booktabs = T)
```

```r
lm(return ~ cap_u, data = sample_data) %>%
   broom::tidy() %>%
   knitr::kable(caption = "Regression output when the independent var.
                comes from uniformization rescaling.", booktabs = T)
```