

# Forecasting and Forensic Analytics

## Session 2: Forecasting with Linear Regression

Dr. Wang Jiwei

# Preface

# Learning objectives

## Foundations

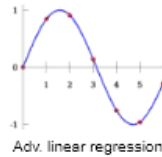
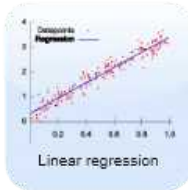


Intro to R

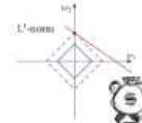
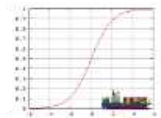


Data in R

## Forecasting



## Binary classification



## Advanced methods



## ■ Theory:

- Develop a logical approach to problem solving with data
  - Statistical hypothesis testing
    - Null vs. Alternative hypothesis
    - Test statistics
    - p-value

## ■ Application:

- Forecasting revenue for real estate firms in Singapore

## ■ Methodology:

- Univariate statistics
- Multivariate linear regression
- Visualization using R

# Assignments for this course


- Tentatively, individual assignments will be posted online after the following seminars:

1. Session 2, on forecasting analytics
2. Session 5, on a mix of linear and logit models
3. Session 7, on forensic analytics
4. Session 9, on other methods

- Individual assignments will be in R Markdown (.rmd) file format

For each assignment, you will have until the following Monday at 11:59pm to finish it (7 days). All submissions and feedback are on eLearn.

# R Markdown: A quick guide

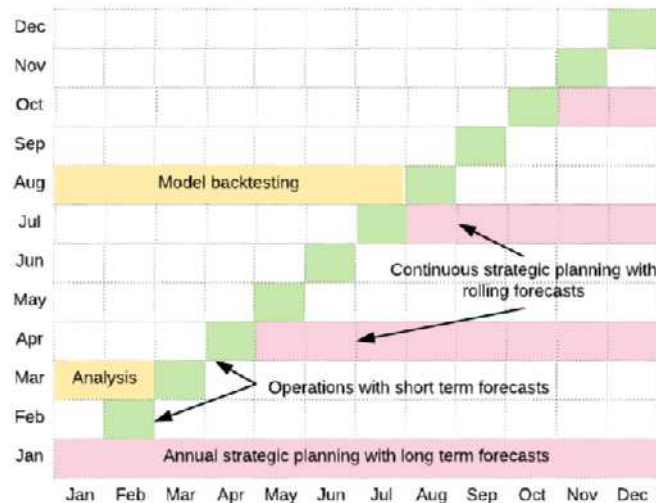
- Headers and subheaders start with # and ##, respectively
- Code blocks starts with ````{r}` and end with ````` (backticks or grave accent)
  - By default, all code and figures will show up in the document
- Inline code goes in a block starting with ``r`` and ending with ```
- Italic font can be used by putting `*` or `_` around text
- Bold font can be used by putting `**` around text
  - E.g.: `**bold text**` becomes **bold text**
- To render the document, click  Knit
- Math can be placed between `$` to use LaTeX notation
  - E.g. `$$\frac{revt}{at}$$` becomes  $\frac{revt}{at}$
- Full equations (on their own line) can be placed between `$$`
- A block quote is prefixed with `>`
- For a complete guide, see R Studio's [R Markdown::Cheat Sheet](#)

# **Application: Revenue prediction**

# The question

What factors can help us to forecast revenue of a company for budgeting, reporting, valuation, and other purposes?

- Case: Uber's financial forecasting with DS and ML



Can your company do the same as Uber?

# Weather data? Satellite images?

- Case: Weather as a Commodity
  - "Hedge Funds now employ a variety of techniques to track weather fluctuation in order to get a cutting edge over competitors and to increase their profit margins."
  - "A legacy has been created by RS Metrics LLC., a provider of satellite imagery and quantitative analysis, ever since they forecasted Walmart's second quarter customer traffic in 2011 using satellite images of parking lot traffic measurements."
- What other creative data might there be?



# Forecasting application

- Forecast sales of a real estate company in Singapore
- using
  - company's own data
  - other companies' data
  - macro economic data



# Linear models

# What is a linear model?

- Revist the following model

$$\hat{y} = \alpha + \beta\hat{x} + \varepsilon$$

- This simplest model is trying to predict some outcome  $\hat{y}$  as a function of an input  $\hat{x}$ 
  - $\hat{y}$  in our case is a firm's revenue in a given year
  - $\hat{x}$  could be a firm's assets in a given year or any other factors we can identify
  - $\alpha$  and  $\beta$  are coefficients solved for
  - $\varepsilon$  is the error in the measurement

■ This is an *OLS* model -- **Ordinary Least Square** regression

# Example

Let's predict UOL's revenue



Residential



Commercial



Hospitality

- Compustat has data for UOL since 1989 (till 2019 for this example)
  - more missing data before 1994
  - numbers in Millions

```
# revt: Revenue, at: Assets  
summary(uol[, c("revt", "at")])
```

##	revt	at
##	Min. : 94.78	Min. : 1218
##	1st Qu.: 213.05	1st Qu.: 3052
##	Median : 464.99	Median : 3520
##	Mean : 774.38	Mean : 6510
##	3rd Qu.: 1212.26	3rd Qu.: 9044
##	Max. : 2397.34	Max. : 20664

# Linear models in R

- To run a linear model, use `lm()`
  - The first argument is a formula for your model, where tilde `~` is used in place of an equals sign
    - The left side is what you want to predict
    - The right side is inputs for prediction, separated by `+`
  - The second argument is the data to use
- Additional variations for the formula:
  - Functions transforming inputs (as vectors), such as `log()`
  - Fully interacting variables using asterisk/star `*`
    - i.e., `A*B` includes, `A`, `B`, and `A times B` in the model
  - Interactions using colon `:`
    - i.e., `A:B` just includes `A times B` in the model

*# Example:*

```
lm(revt ~ at, data = uol)
```

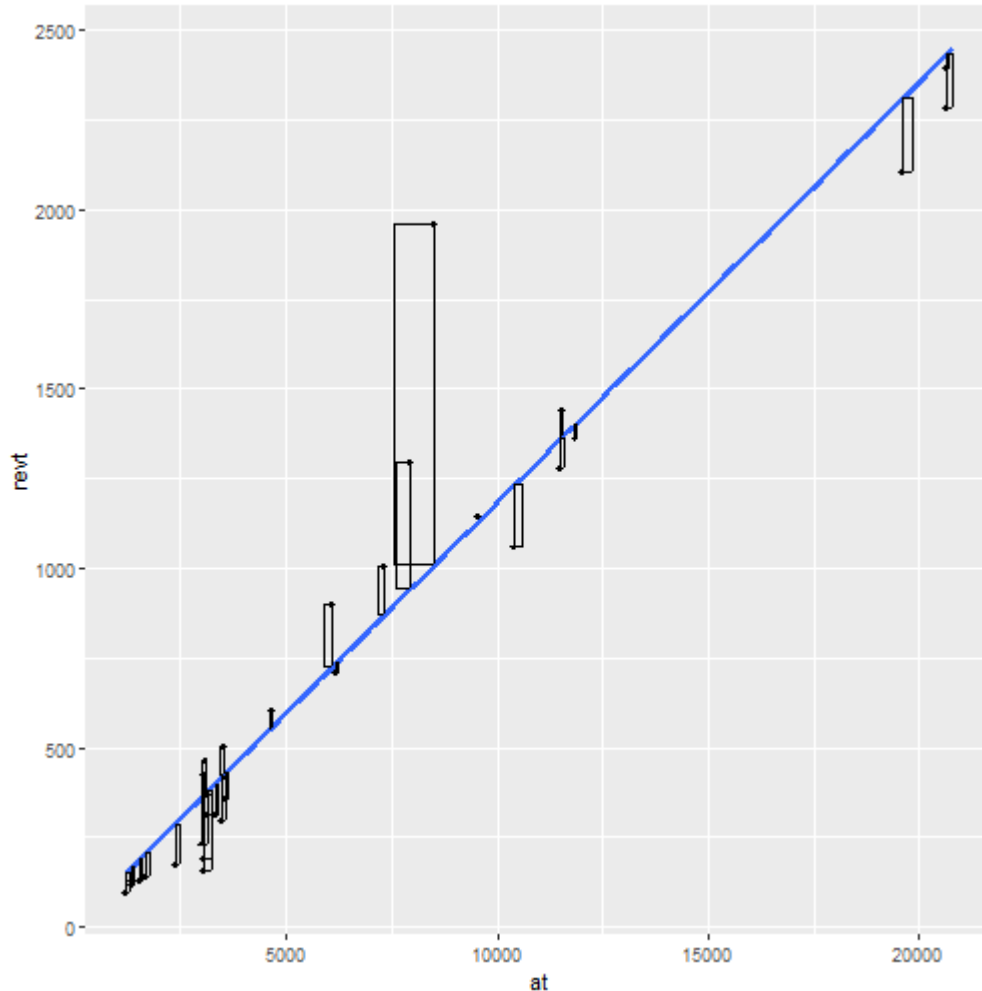
# Example: UOL

```
mod1 <- lm(revt ~ at, data = uol)
summary(mod1)
```

```
##
## Call:
## lm(formula = revt ~ at, data = uol)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -212.45  -98.13  -48.29   53.50  949.34
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.101598  60.085716   0.168   0.868
## at          0.117403   0.007031  16.698 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 216.7 on 29 degrees of freedom
## Multiple R-squared:  0.9058,    Adjusted R-squared:  0.9025
## F-statistic: 278.8 on 1 and 29 DF,  p-value: < 2.2e-16
```

■ \$1 more in assets leads to \$0.12 more in revenue

# Why called Ordinary Least Squares?



# Example: UOL

- This model wasn't so interesting...
  - Bigger firms have more revenue -- this is a given
- How about... revenue *growth*?
- And *change* in assets
  - i.e., Asset growth

$$\Delta x_t = \frac{x_t}{x_{t-1}} - 1$$



# Calculating changes in R

- The easiest way is using `package:tidyverse`'s `package:dplyr`
  - `lag()` function along with `mutate()`
- The default way to do it is to create a vector manually

```
# tidyverse with pipe %>%
uol <- uol %>%
  mutate(revt_growth1 = revt / lag(revt, order_by = fyear) - 1)

# which is equivalent to
uol <- mutate(uol, revt_growth2 = revt / lag(revt) - 1)

# Base R way, [-n] to index vector less the nth element
uol$revt_growth3 = uol$revt / c(NA, uol$revt[-length(uol$revt)]) - 1
identical(uol$revt_growth1, uol$revt_growth3)
```

```
## [1] TRUE
```

```
# magrittr %<>% to combine <- and %>%
library(magrittr)
uol %<>% mutate(revt_growth4 = revt / lag(revt) - 1)
identical(uol$revt_growth1, uol$revt_growth4)
```

```
## [1] TRUE
```

# A note on lag() and lead()

- `lag()` or `lead()` finds the "previous" or "next" values in a vector.
- Very useful for comparing values ahead of or behind the current values.
- The dataset must be sorted by the key (eg, time for time series data)

```
# Use order_by if data not already ordered
dff <- data.frame(year = 2001:2003, value = (1:3) ^ 2)
scrambled <- dff[sample(nrow(dff)), ]

wrong <- mutate(scrambled, prev = lag(value))
arrange(wrong, year)
```

```
##   year value prev
## 1 2001     1    4
## 2 2002     4   NA
## 3 2003     9    1
```

```
right <- mutate(scrambled, prev = lag(value, order_by = year))
arrange(right, year)
```

```
##   year value prev
## 1 2001     1   NA
## 2 2002     4    1
## 3 2003     9    4
```

# A note on mutate()

- `mutate()` adds variables to an existing data frame
  - Also `mutate_all()`, `mutate_at()`, `mutate_if()`
    - `mutate_all()` applies a transformation to all values in a data frame and adds these to the data frame
    - `mutate_at()` does this for a set of specified variables
    - `mutate_if()` transforms all variables matching a condition
      - Such as `is.numeric`
- Mutate can be very powerful when making more complex variables
  - For instance: Calculating growth within company in a multi-company data frame (cross-sectional with time series data, ie, panel data)
- Do Exercise 1 in the **R Practice**

# Example: UOL with changes

```
# Make the other needed change
uol <- uol %>%
  mutate(at_growth = at / lag(at) - 1) # From dplyr
# Rename our revenue growth variable
uol <- rename(uol, revt_growth = revt_growth1) # From dplyr
# Run the OLS model
mod2 <- lm(revt_growth ~ at_growth, data = uol)
summary(mod2)
```

```
##
## Call:
## lm(formula = revt_growth ~ at_growth, data = uol)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56897 -0.12016 -0.01099  0.15012  0.42991
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.08443    0.05215   1.619   0.1167
## at_growth     0.55576    0.26591   2.090   0.0458 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.237 on 28 degrees of freedom
```

# Example: UOL with changes

- $\Delta$ Assets doesn't capture  $\Delta$ Revenue so well
- Perhaps change in total assets is a bad choice?
- Or perhaps we need to expand our model?

# Scaling up!

$$\hat{y} = \alpha + \beta_1 \hat{x}_1 + \beta_2 \hat{x}_2 + \dots + \varepsilon$$

- OLS doesn't need to be restricted to just 1 input!
  - Not unlimited though (yet)
    - Number of inputs must be less than the number of observations minus 1
- Each  $\hat{x}_i$  is an input in our model
- Each  $\beta_i$  is something we will solve for
- $\hat{y}$ ,  $\alpha$ , and  $\varepsilon$  are the same as before

■ We have... 823 variables from Compustat Global alone!

- Let's just add them all?
  - This is a very machine-learning mindset
- We only have 31 observations...
  - $31 \ll 823$ ...

■ Now what?

# Scaling up our model

| Building a model requires careful thought!

- What makes sense to add to our model?

| This is where having accounting and business knowledge comes in!

- Some potential sources to consider:
  - Direct accounting relations
    - Financing? Capex? R&D? Other expenditures?
  - Business management and corporate structure
    - Some management characteristics may matter
    - Corporate governance may also matter
  - Economics
    - Macro econ: trade, economic growth, population, weather
    - Micro econ: Other related firms like suppliers and customers
  - Legal factors
    - Any changes in law? Favorable or not?
  - Market factors
    - Interest rates, cost of capital, foreign exchange?

| Any other factors?

# Scaling up our model

- One possible improvement:

```
# lct: short term liabilities, che: cash and equivalents, ebit: EBIT
uol <- uol %>%
  mutate_at(vars(lct, che, ebit), funs(growth = . / lag(.) - 1))
mod3 <- lm(revt_growth ~ lct_growth+che_growth+ebit_growth, data=uol)
summary(mod3)
```

```
##
## Call:
## lm(formula = revt_growth ~ lct_growth + che_growth + ebit_growth,
##     data = uol)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45761 -0.13091 -0.00649  0.17247  0.32014
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.06853    0.04569   1.500  0.14566
## lct_growth     0.23717    0.06988   3.394  0.00222 **
## che_growth    -0.11357    0.08822  -1.287  0.20931
## ebit_growth    0.03861    0.02128   1.814  0.08117 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



# Formalizing testing

# Why formalize?

- Our current approach has been ad hoc
  - What is our goal?
  - How will we know if we have achieved it?
- Formalization provides more rigor



# Scientific method

## 1. Question

- What are we trying to determine?
- Fundamentally, the question is asked/answered to solve your business problems

## 2. Hypothesis

- What do we think will happen? Make a statement
  - "If X, then Y"
  - e.g., "If capital expenditures increase, revenue will increase."
  - A good hypothesis based on information in prior research, ie, hypothesis typically follows a thorough *literature review*
- Null hypothesis, a.k.a.  $H_0$ 
  - Typically: The statement *doesn't* work
- Alternative hypothesis, a.k.a.  $H_1$  or  $H_A$ 
  - The statement *does* work (and perhaps how it works)

## 3. Research design

- What exactly will we test? How to measure X and Y?
- Formalize a statistical model

## 4. Testing

- Test the model

## 5. Analysis

- Did it work?

# Test statistics

- Testing a coefficient:
    - Use a  $t$  (less assumption on normality, unknown population s.d., more commonly used) or  $z$  test (known population s.d.)
  - Testing a model as a whole
    - $F$ -test, check *adjusted* R squared as well
  - Testing across models
    - Chi squared ( $\chi^2$ ) test
    - Vuong test (comparing  $R^2$ )
    - **Akaike Information Criterion** (AIC) (Comparing MLEs, lower is better)
- All of these have p-values, except for AIC

# Revisiting the previous problem

# Formalizing our last test

## 1. Question

- 

## 2. Hypotheses

- $H_0$ :

- $H_1$ :

## 3. Research design

- Individual variables:

- Model:

## 4. Testing:

-

# Formalizing our last test

## 1. Question

- Can we predict changes in revenue using a firm's accounting information?

## 2. Hypotheses

- $H_0$ : Our variables do not predict UOL's change in revenue
- $H_1$ : Our variables are help to predict UOL's change in revenue

## 3. Research design

- Individual variables
  - Growth in current liabilities (+)
  - Growth in cash and cash equivalent (+)
  - Growth in EBIT (+)
- Model: OLS

## 4. Testing:

- t-test for coefficients and F-test for model

# Is this model better?

```
anova(mod2, mod3, test = "Chisq")
```

```
## Analysis of Variance Table
##
## Model 1: revt_growth ~ at_growth
## Model 2: revt_growth ~ lct_growth + che_growth + ebit_growth
##   Res.Df    RSS Df Sum of Sq Pr(>Chi)
## 1      28 1.5721
## 2      26 1.2035  2    0.36861 0.01865 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

■ A bit better at  $p < 0.05$

- This means our model with change in current liabilities, cash, and EBIT appears to be better than the model with change in assets.

■ Note: p-value tells the prob that the two models are the same (in terms of variance explained). If the first model is better, the Sum of Sq number will be significantly negative. RSS also speaks.



# Panel data

# Expanding our methodology

- Why should we limit ourselves to 1 firm's data?
- The nature of data analysis is such:
  - Adding more data usually helps improve predictions
- Assuming:
  - The data isn't of low quality (too noisy)
  - The data is relevant
  - Any differences can be reasonably controlled for

# Expanding our question

- Previously: Can we predict revenue using a firm's accounting information?
  - This is simultaneous, and thus is not forecasting
- Now: Can we predict *future* revenue using a firm's accounting information?
  - By trying to predict ahead, we are now in the realm of forecasting
  - What do we need to change?
    - $\hat{y}$  will need to be 1 year in the future

# First things first

- When using a lot of data, it is important to make sure the data is clean
- In our case, we may want to remove any very small firms

```
# Ensure firms have at least $1M (local currency), and have revenue  
# df contains all real estate companies excluding North America  
df_clean <- filter(df, df$at > 1, df$revt > 0)
```

```
# We cleaned out 2,177 observations!  
print(c(nrow(df), nrow(df_clean)))
```

```
## [1] 34156 31979
```

```
# Another useful cleaning function:  
# Replaces NaN, Inf, and -Inf with NA for all numeric variables!  
df_clean <- df_clean %>%  
  mutate_if(is.numeric, funs(replace(., !is.finite(.), NA)))
```

- `is.finite()` returns a vector of the same length as `x` the `j`th element of which is TRUE if `x[j]` is finite (i.e., it is not one of the values NA, NaN, Inf or -Inf) and FALSE otherwise. Click [here](#) for the R function.

# Looking back at the prior models

```
uol <- uol %>% mutate(revt_lead = lead(revt)) # From dplyr
forecast1 <-
  lm(revt_lead ~ lct + che + ebit, data=uol)
library(broom) # To display regression outputs in a tidy fashion
tidy(forecast1) # present regression output
```

```
## # A tibble: 4 x 5
##   term          estimate std.error statistic p.value
##   <chr>         <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)   64.0      127.        0.505   0.618
## 2 lct           0.392      0.237        1.65    0.111
## 3 che           0.141      0.330        0.425   0.674
## 4 ebit          2.03       1.04        1.96    0.0613
```

```
glance(forecast1) # present regression statistics
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic      p.value    df logLik   AIC    BIC
##   <dbl>         <dbl> <dbl>     <dbl>        <dbl> <int>  <dbl> <dbl> <dbl>
## 1    0.746         0.717  369.     25.5 0.0000000663     4  -218.  446.  453.
```

**|** The model is significant but not the coefficients. We can do better.

# Expanding the prior model

```
forecast2 <-  
  lm(revt_lead ~ revt + act + che + lct + dp + ebit , data=uol)  
tidy(forecast2)
```

```
## # A tibble: 7 x 5  
##   term          estimate std.error statistic    p.value  
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>  
## 1 (Intercept)    75.2      97.1      0.775  0.446  
## 2 revt           1.63       0.318     5.11  0.0000356  
## 3 act            0.212      0.168     1.26  0.219  
## 4 che            0.264      0.290     0.912 0.371  
## 5 lct           -0.238      0.190    -1.25  0.223  
## 6 dp            -1.45       4.42    -0.328 0.746  
## 7 ebit          -3.28       1.12    -2.91  0.00780
```

- Revenue (revt) to capture stickiness of revenue
- Current assest (act) & Cash (che) to capture asset base
- Current liabilities (lct) to capture payments due
- Depreciation (dp) to capture decrease in real estate asset values
- EBIT to capture operational performance

# Expanding the prior model

```
glance(forecast2)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC    BIC devi
##   <dbl>      <dbl> <dbl>     <dbl>    <dbl> <int> <dbl> <dbl> <dbl> <dbl>
## 1    0.923        0.903   216.      46.1 1.11e-11     7  -200.  416.  427. 1074
```

```
anova(forecast1, forecast2, test="Chisq")
```

```
## Analysis of Variance Table
##
## Model 1: revt_lead ~ lct + che + ebit
## Model 2: revt_lead ~ revt + act + che + lct + dp + ebit
##   Res.Df    RSS Df Sum of Sq Pr(>Chi)
## 1      26 3548955
## 2      23 1074135  3   2474820 1.84e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

■ This is better (Adj.  $R^2$ ,  $\chi^2$ , AIC).

# Panel data

- Panel data refers to data with the following characteristics:
  - There is a time dimension
  - There is at least 1 other dimension to the data (firm, country, etc.)
- Special cases:
  - A panel where all dimensions have the same number of observations is called *balanced*
    - Otherwise we call it *unbalanced*
  - A panel missing the time dimension is *cross-sectional*
  - A panel missing the other dimension(s) is a *time series*
- Format:
  - Long: Indexed by all dimensions (e.g., country-year)
  - Wide: Indexed only by other dimensions (e.g., country only)



# dplyr makes transpose easy

- Depending on data source, you may need to transform the data format from wide to long (or long to wide).
- The `package::dplyr` has a `gather()` function to do so.

Gather() `gather(year, growth, q1_2017:q4_2018)`

Messy

country	q1_2017	q2_2017	q3_2017	q4_2017
A	0.03	0.05	0.04	0.03
B	0.05	0.07	0.05	0.02
C	0.01	0.02	0.01	0.04

tidier

country	time	growth
A	q1_2017	0.03
B	q1_2017	0.05
C	q1_2017	0.01
A	q2_2017	0.05
B	q2_2017	0.07
C	q2_2017	0.02
A	q3_2017	0.04
B	q3_2017	0.05
C	q3_2017	0.01
A	q4_2017	0.03

# Wide versus long data

```
university_wide # randomly generated numbers
```

```
##   university rand.2016 rand.2017 rand.2018
## 1          SMU        25         82         83
## 2          NTU        25         82         83
## 3          NUS        25         82         83
```

```
# convert wide to long dataset
```

```
library("tidyr", "dplyr")
```

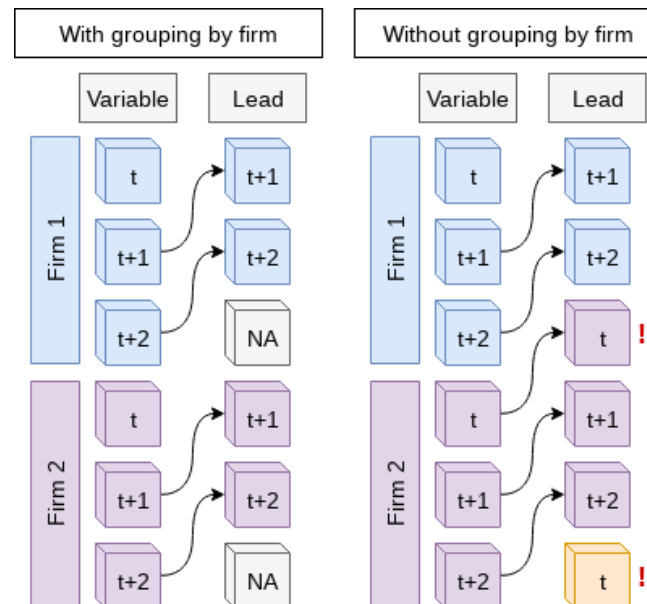
```
university_long <- university_wide %>%
  gather(year, rand, rand.2016:rand.2018) %>%
  mutate(year = as.numeric(gsub("rand.", "", year))) %>%
  arrange(desc(year))
university_long
```

```
##   university year rand
## 1          SMU 2018   83
## 2          NTU 2018   83
## 3          NUS 2018   83
## 4          SMU 2017   82
## 5          NTU 2017   82
## 6          NUS 2017   82
## 7          SMU 2016   25
## 8          NTU 2016   25
```

# All SG real estate companies

```
# group_by - without it, lead() will pull from the subsequent firm!  
# ungroup() tells R that we finished grouping  
df_clean <- df_clean %>%  
  group_by(isin) %>%  
  mutate(revt_lead = lead(revt)) %>%  
  ungroup()
```

- Do Exercises 2 and 3 of the **R Practice**



# All SG real estate companies

```
forecast3 <- lm(revt_lead ~ revt + act + che + lct + dp + ebit,  
               data = df_clean[df_clean$fic == "SGP", ])  
tidy(forecast3)
```

```
## # A tibble: 7 x 5  
##   term          estimate std.error statistic  p.value  
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>  
## 1 (Intercept)  21.5      11.6      1.86 6.39e- 2  
## 2 revt         0.537     0.0579    9.26 1.07e-18  
## 3 act          0.00999    0.0405    0.247 8.05e- 1  
## 4 che          0.480     0.118     4.07 5.59e- 5  
## 5 lct          0.218     0.0612    3.56 4.20e- 4  
## 6 dp           4.38      0.960     4.56 6.67e- 6  
## 7 ebit        -1.13      0.238    -4.72 3.17e- 6
```

# All SG real estate companies

```
glance(forecast3)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic    p.value    df logLik   AIC    BIC de
##   <dbl>      <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>
## 1    0.836      0.833  206.     352. 1.30e-159     7 -2850. 5717. 5749. 176
```

Lower adjusted  $R^2$  -- This is worse? Why?

- Note:  $\chi^2$  can only be used for models on the same data
  - Same for AIC

# Worldwide real estate companies

```
forecast4 <-  
  lm(revt_lead ~ revt + act + che + lct + dp + ebit , data = df_clean)  
tidy(forecast4)
```

```
## # A tibble: 7 x 5  
##   term          estimate std.error statistic  p.value  
##   <chr>          <dbl>      <dbl>      <dbl>    <dbl>  
## 1 (Intercept) 220.        579.         0.379 7.04e- 1  
## 2 revt         1.05        0.00634    165.     0.  
## 3 act        -0.0234       0.00539    -4.33 1.50e- 5  
## 4 che         0.0203       0.0269     0.756 4.49e- 1  
## 5 lct         0.0553       0.00866     6.39 1.82e-10  
## 6 dp          0.172        0.186     0.927 3.54e- 1  
## 7 ebit         0.126        0.0652     1.94 5.29e- 2
```

# Worldwide real estate companies

```
glance(forecast4)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC
##   <dbl>      <dbl>    <dbl>     <dbl>   <dbl> <int>  <dbl>   <dbl> <dbl>
## 1    0.947        0.947 40818.    15138.     0     7 -61343. 122702. 122754.
```

Higher adjusted  $R^2$  -- better!

- Note:  $\chi^2$  can only be used for models on the same data
  - Same for AIC

# Model accuracy

| Why is 1 model better while the other model is worse?

■ Ranking:

1. Worldwide real estate model
2. UOL model
3. Singapore real estate model

| Different sources of noise, amounts of data



# Dealing with noise

# Noise

- | Statistical noise is random error in the data
- Many sources of noise:
  - Other factors not included in
  - Error in measurement
    - Accounting measurement!
  - Unexpected events / shocks
- | Noise is OK, but the more we remove, the better!

# Removing noise: Singapore model

- Different companies may behave slightly differently (but time-invariant)
  - Control for this using a *Fixed Effect* of companies
  - Note: ISIN uniquely identifies companies
  - `factor(isin)`: (n-1) dummy variables
  - FE equivalent to unique intercept for each company

```
forecast3.1 <-  
  lm(revt_lead ~ revt + act + che + lct + dp + ebit + factor(isin),  
     data = df_clean[df_clean$fic == "SGP", ])  
# n=7 to prevent outputting every fixed effect  
print(tidy(forecast3.1), n=7)
```

```
## # A tibble: 30 x 5  
##   term          estimate std.error statistic    p.value  
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>  
## 1 (Intercept) -0.00946    36.8    -0.000257 1.00  
## 2 revt         0.403      0.0712    5.66    0.0000000293  
## 3 act          0.0486     0.0453    1.07     0.284  
## 4 che          0.276     0.139     1.99     0.0472  
## 5 lct          0.239     0.0656    3.65     0.000300  
## 6 dp           4.86      1.05     4.63     0.00000487  
## 7 ebit         -1.07      0.269    -3.98     0.0000825  
## # ... with 23 more rows
```

# Removing noise: Singapore model

```
glance(forecast3.1)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic    p.value    df logLik   AIC    BIC    de
##   <dbl>      <dbl> <dbl>      <dbl>      <dbl> <int>  <dbl> <dbl> <dbl>
## 1     0.852        0.841   201.        77.9 8.39e-144    30 -2828.  5719.  5844.  159
```

```
anova(forecast3, forecast3.1, test="Chisq")
```

```
## Analysis of Variance Table
##
## Model 1: revt_lead ~ revt + act + che + lct + dp + ebit
## Model 2: revt_lead ~ revt + act + che + lct + dp + ebit + factor(isin)
##   Res.Df      RSS Df Sum of Sq Pr(>Chi)
## 1     416 17663454
## 2     393 15915304 23   1748150 0.006616 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

■ The model 3.1 is better

# Another way to do fixed effects

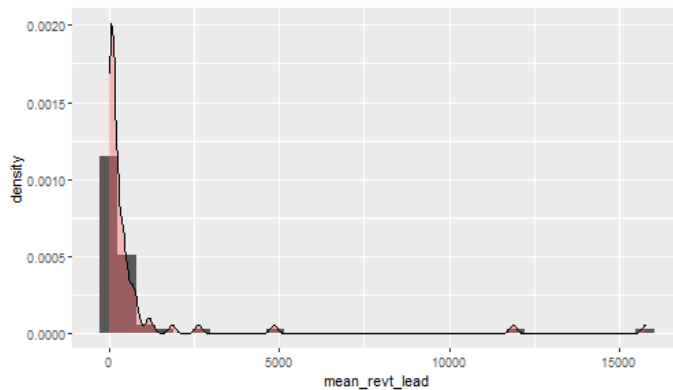
- The library `package:lfe` has `felm()`: fixed effects linear model
  - Better for complex models

```
library(lfe)
forecast3.2 <-
  felm(revt_lead ~ revt + act + che + lct + dp + ebit | factor(isin),
        data = df_clean[df_clean$fic == "SGP", ])
summary(forecast3.2)
```

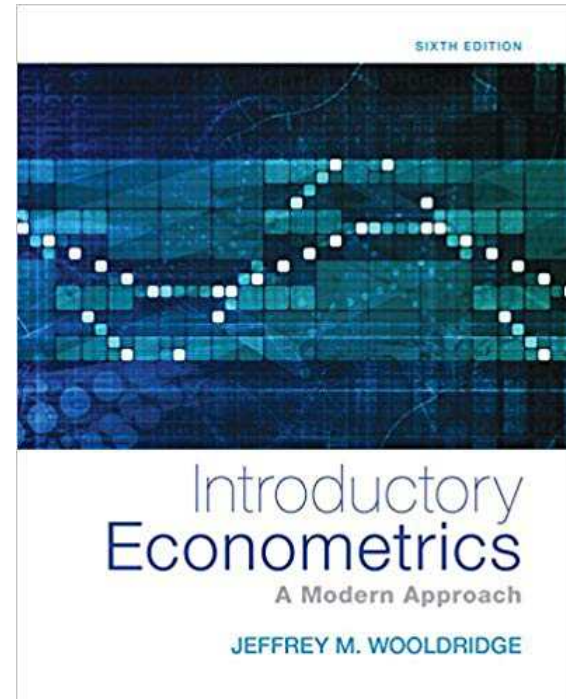
```
##
## Call:
##   felm(formula = revt_lead ~ revt + act + che + lct + dp + ebit |      factor
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1190.94   -38.87     0.24    18.63   1967.21
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## revt   0.40306   0.07122   5.659 2.93e-08 ***
## act    0.04857   0.04527   1.073  0.2840
## che    0.27601   0.13863   1.991  0.0472 *
## lct    0.23942   0.06563   3.648  0.0003 ***
## dp     4.85697   1.04793   4.635 4.87e-06 ***
## ebit  -1.07069   0.26910  -3.979 8.25e-05 ***
```

# Why exactly would we use FE?

- Fixed effects are used when the average of  $\hat{y}$  varies by some group in our data
  - In our problem, the average revenue of each firm is different
- Fixed effects absorb this difference



- Further reading:
  - Introductory Econometrics by Jeffrey M. Wooldridge



# What else can we do?

- | What else could we do to improve our prediction model?

**Macro data**



# Macro data sources

- For Singapore: [Data.gov.sg](https://data.gov.sg)
  - Covers: Economy, education, environment, finance, health, infrastructure, society, technology, transport
- For real estate in Singapore: URA's REALIS system
  - Access through the library
- WRDS has some as well
- For US: [data.gov](https://data.gov), as well as many agency websites
  - Like [BLS](https://www.bls.gov) or the [Federal Reserve](https://www.federalreserve.gov)



# Loading macro data

- Singapore business expectations data (from [data.gov.sg](https://data.gov.sg))

```
expectations %>%  
  arrange(level_2, level_3, desc(year)) %>% # sort the data  
  select(year, quarter, level_2, level_3, value) %>%  
  datatable(options = list(pageLength = 3), rownames=FALSE)
```

Show  entries

Search:

year	quarter	level_2	level_3	value
2019	1	Accommodation & Food Services	Accommodation	-2
2019	2	Accommodation & Food Services	Accommodation	25
2019	3	Accommodation & Food Services	Accommodation	22

Showing 1 to 3 of 891 entries

Previous

1

2

3

4

5

...

297

Next

# Transforming macro data

```
# extract out F&I only, calculate annual average value
expectations_avg <- expectations %>%
  filter(level_2 == "Financial & Insurance") %>%      # Keep F&I sector
  group_by(year) %>%                                   # Group data by year
  mutate(fin_sentiment=mean(value, na.rm=TRUE)) %>%    # Calculate yearly a
  slice(1)                                              # Take only 1 row pe
head(expectations_avg)
```

```
## # A tibble: 6 x 7
## # Groups:   year [6]
##   quarter level_1          level_2          level_3
##   <dbl> <chr>          <chr>          <chr>
## 1      1 Total Services Sector Financial & Insurance Banks & Finance Companies
## 2      1 Total Services Sector Financial & Insurance Banks & Finance Companies
## 3      1 Total Services Sector Financial & Insurance Banks & Finance Companies
## 4      1 Total Services Sector Financial & Insurance Banks & Finance Companies
## 5      1 Total Services Sector Financial & Insurance Banks & Finance Companies
## 6      1 Total Services Sector Financial & Insurance Banks & Finance Companies
```

- At this point, we can merge with our accounting data

# dplyr makes merging easy

- For merging, use **package:dplyr**'s `*_join()` commands
  - `left_join()` and `right_join()` for merging a dataset into another
  - `inner_join()` for keeping only matched observations
  - `full_join()` for making all possible combinations

**Left\_join()**

Table 1		Table 2	
ID	y	ID	z
A	5	A	30
B	5	B	21
C	8	C	22
D	0	D	25
F	9	E	29




Table 3		
ID	y	z
A	5	30
B	5	21
C	8	22
D	0	25
F	9	NA

**right\_join()**

Table 1		Table 2	
ID	y	ID	z
A	5	A	30
B	5	B	21
C	8	C	22
D	0	D	25
F	9	E	29

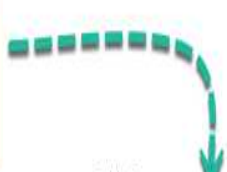


Table 3		
ID	y	z
A	5	30
B	5	21
C	8	22
D	0	25
E	NA	29

# dplyr makes merging easy

- inner\_join() vs. full\_join()

inner\_join()

Table 1

ID	y
A	5
B	5
C	8
D	0
F	9

Table 2

ID	z
A	30
B	21
C	22
D	25
E	29



Table 3

ID	y	z
A	5	30
B	5	21
C	8	22
D	0	25

full\_join()

Table 1

ID	y
A	5
B	5
C	8
D	0
F	9

Table 2

ID	z
A	30
B	21
C	22
D	25
E	29

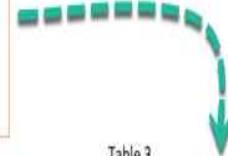


Table 3

ID	y	z
A	5	30
B	5	21
C	8	22
D	0	25
E	NA	29
F	9	NA

# Merging example

■ Merge in the finance sentiment data to our accounting data

```
# subset out our data, since our macro data is Singapore-specific  
df_SG <- df_clean %>% filter(fic == "SGP")  
  
# Create year in df_SG (date is given by datadate as YYYYMMDD)  
df_SG$year = round(df_SG$datadate / 10000, digits=0)  
  
# Combine datasets  
# Notice how it automatically figures out to join by "year"  
df_SG_macro <- left_join(df_SG,  
                          expectations_avg[,c("year", "fin_sentiment")])  
  
## Joining, by = "year"
```

**Predicting with macro data**

# Building in macro data

- First try: Just add it in

```
macro1 <- lm(revt_lead ~ revt + act + che + lct + dp + ebit + fin_sentiment  
             data = df_SG_macro)  
tidy(macro1)
```

```
## # A tibble: 8 x 5  
##   term          estimate std.error statistic  p.value  
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>  
## 1 (Intercept)    19.1      13.8       1.39 1.66e- 1  
## 2 revt           0.532     0.0599     8.88 2.47e-17  
## 3 act            0.0119    0.0421     0.283 7.78e- 1  
## 4 che            0.483     0.124     3.89 1.16e- 4  
## 5 lct            0.216     0.0635     3.41 7.19e- 4  
## 6 dp             4.42      0.992     4.46 1.08e- 5  
## 7 ebit           -1.12     0.247    -4.55 7.10e- 6  
## 8 fin_sentiment  0.302     0.561     0.538 5.91e- 1
```

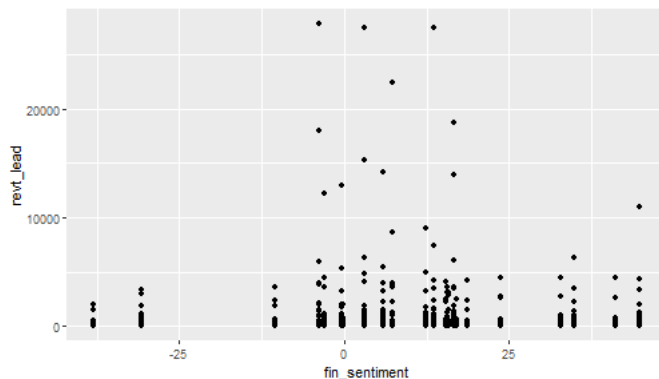
■ It isn't significant. Why is this?



# Scale matters

- All of our firm data is on the same scale as revenue: dollars within a given firm
- But `fin_sentiment` has much smaller range of constant scale (-38 to 44.65)
  - one sentiment value is corresponding to many revenue points, as depicted in the left chart below
  - Need to scale (standardize or normalize) this to fit the problem
- Do Exercise 4 of the **R Practice** on visualization using `ggplot2`

```
df_SG_macro %>%  
  ggplot(aes(y=revt_lead,  
             x=fin_sentiment)) +  
  geom_point()
```



```
df_SG_macro %>%  
  ggplot(aes(y=revt_lead,  
             x=scale(fin_sentiment)*revt)) +  
  geom_point()
```

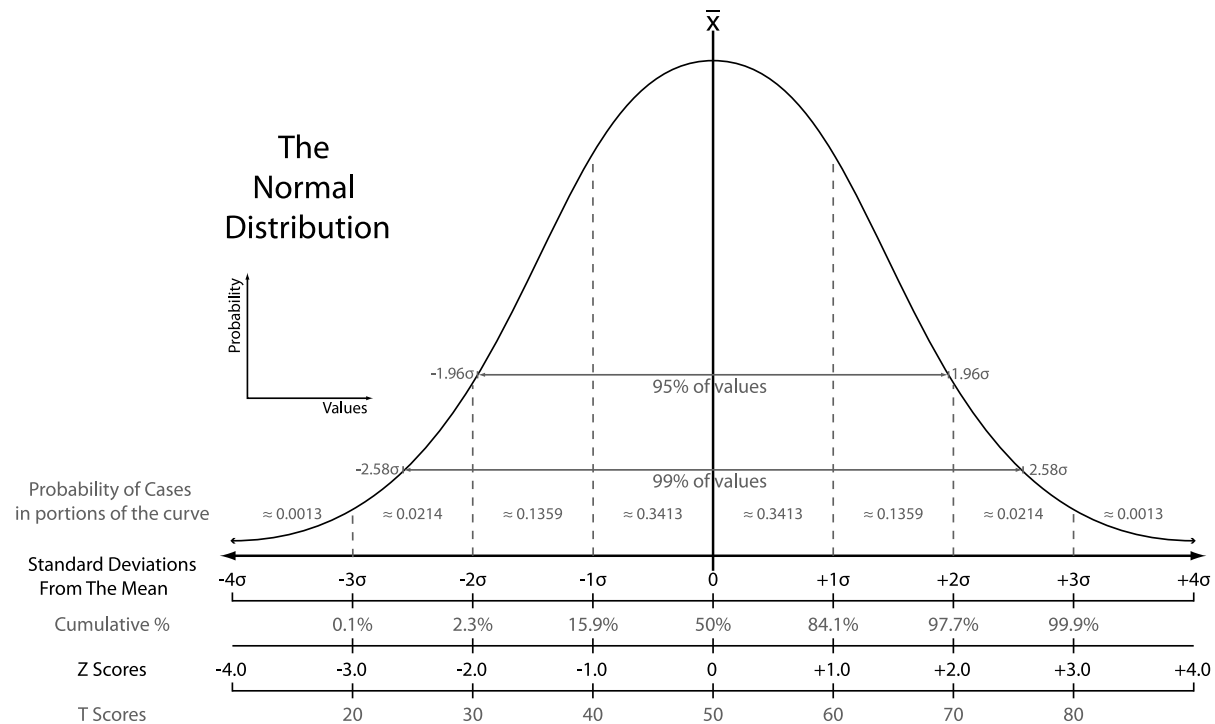
# Feature scaling

- There are various ways to **scale variables/features**. In general, one way is to scale to a standard normal distribution ("standardization") and the other is to scale to range [0, 1] ("normalization")
- Standardization (or **Z-score normalization**): features will be rescaled so that they'll have the properties of a standard normal distribution with zero mean ( $\mu = 0$ ) and one standard deviation ( $\sigma = 1$ ).
- Standard scores (also called **z scores**) are calculated as follows:

$$z = \frac{x - \mu}{\sigma}$$

- z score measures how many S.D. below or above the variable mean, thus the unit of z score is S.D. of the variable

# The normal distribution

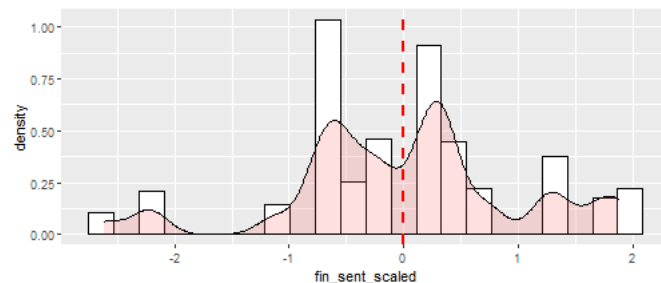
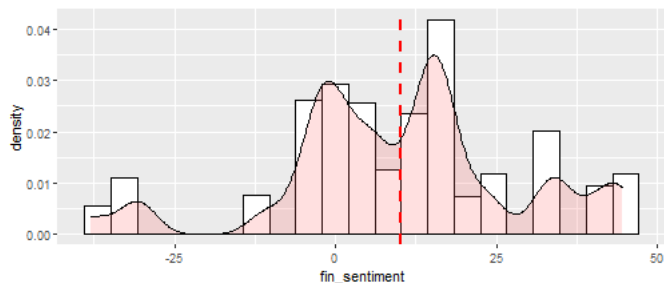


# The scale() function in Base R

- `scale()` function centers/scales the columns of a numeric matrix.
- `package::standardize` is another option.

```
# Scale creates z-scores with 0 mean and 1 sd  
df_SG_macro$fin_sent_scaled <- scale(df_SG_macro$fin_sentiment)  
summary(df_SG_macro[,c("fin_sentiment", "fin_sent_scaled")])
```

```
## fin_sentiment      fin_sent_scaled.V1  
## Min.      :-38.0000  Min.      :-2.61441  
## 1st Qu.: -0.4667    1st Qu.: -0.57333  
## Median : 12.4500    Median : 0.12909  
## Mean    : 10.0762    Mean     : 0.00000  
## 3rd Qu.: 17.0000    3rd Qu.: 0.37652  
## Max.     : 44.6500    Max.     : 1.88014  
## NA's    :68         NA's     :68
```



# Scaled macro data

- z-score normalization and scale by revenue

```
# Scale creates z-scores
df_SG_macro$fin_sent_scaled <- scale(df_SG_macro$fin_sentiment)
macro3 <-
  lm(revt_lead ~ revt + act + che + lct + dp + ebit + fin_sent_scaled:revt
      data=df_SG_macro) # fin_sent_scaled:revt = fin_sent_scaled * revt
tidy(macro3)
```

```
## # A tibble: 8 x 5
##   term                estimate std.error statistic  p.value
##   <chr>              <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)        21.8      12.0      1.81 7.09e- 2
## 2 revt                0.533     0.0593    8.99 1.04e-17
## 3 act                 0.0220    0.0417    0.527 5.98e- 1
## 4 che                 0.419     0.125     3.35 8.74e- 4
## 5 lct                 0.227     0.0628    3.62 3.39e- 4
## 6 dp                  3.89     0.999     3.90 1.14e- 4
## 7 ebit               -0.949     0.252    -3.77 1.86e- 4
## 8 revt:fin_sent_scaled 0.0907    0.0315    2.88 4.25e- 3
```

# Model comparisons

```
baseline <-  
  lm(revt_lead ~ revt + act + che + lct + dp + ebit,  
      data=df_SG_macro[!is.na(df_SG_macro$fin_sentiment),])  
glance(baseline)
```

```
## # A tibble: 1 x 11  
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC    BIC de  
##   <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>  
## 1      0.835           0.833  211.      333. 6.69e-151     7 -2712. 5441. 5473. 176
```

```
glance(macro3)
```

```
## # A tibble: 1 x 11  
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC    BIC de  
##   <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>  
## 1      0.839           0.836  210.      292. 2.15e-151     8 -2708. 5434. 5470. 172
```

Adjusted  $R^2$  and AIC are slightly better with macro data

# Model comparisons

```
anova(baseline, macro3, test="Chisq")
```

```
## Analysis of Variance Table
```

```
##
```

```
## Model 1: revt_lead ~ revt + act + che + lct + dp + ebit
```

```
## Model 2: revt_lead ~ revt + act + che + lct + dp + ebit + fin_sent_scaled:revt
```

```
##   Res.Df      RSS Df Sum of Sq Pr(>Chi)
```

```
## 1      394 17617000
```

```
## 2      393 17253888  1    363112 0.004029 **
```

```
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Macro model definitely fits better than the baseline model!

# Takeaway

1. Adding macro data can help explain some exogenous variation in a model
  - Exogenous meaning outside of the firms, in this case
2. Scaling is very important
  - Not scaling properly can suppress some effects from being visible

## ■ Interpretating the macro variable

- For every 1 S.D. increase in `fin_sentiment` (18.4 points)
  - Revenue stickiness increases by ~9%
- Over the range of data (-38 to 44.65)...
  - Revenue stickiness change ranges from -23.7% to 17%



**Validation: Is it better?**

# Validation

- Ideal:
  - Withhold the last year (or a few) of data when building the model
  - Check performance on *hold out sample*
- Sometimes acceptable:
  - Withhold a random sample of data when building the model
  - Check performance on *hold out sample*

This is the basic idea of machine learning, we will cover more formally in future topics

# Estimation

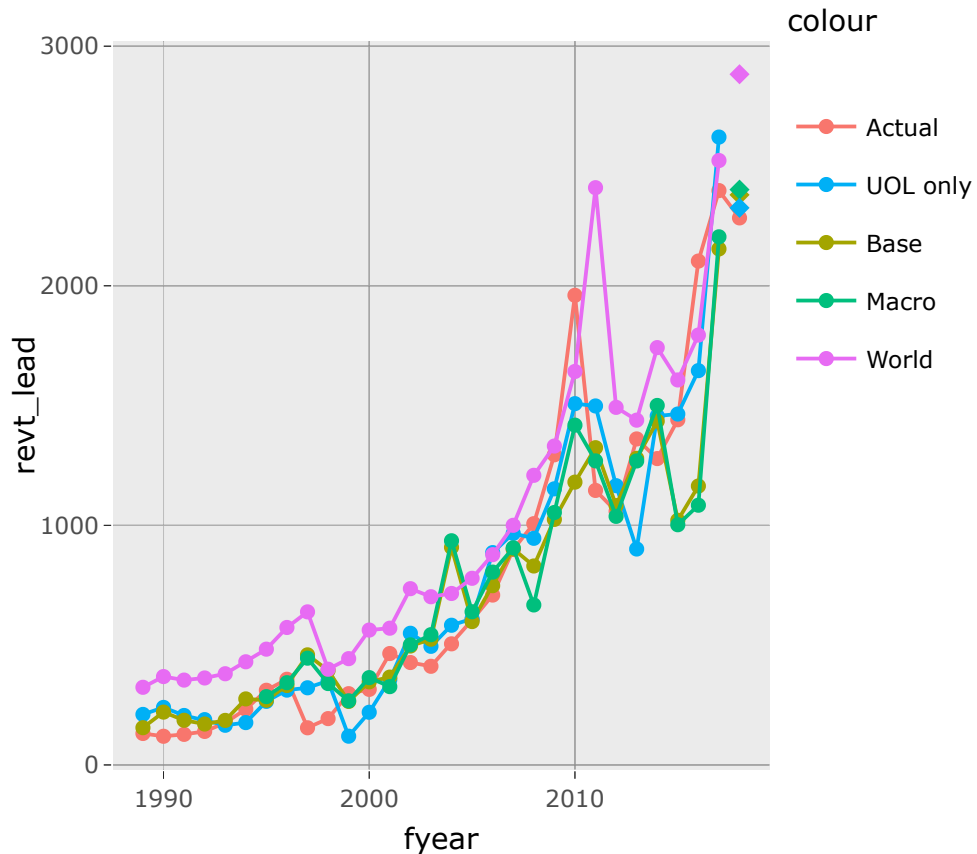
- As we never constructed a hold out sample, let's end by estimating UOL's 2019 year revenue
- It is lead prediction, so fyear = 2018 for the 2019 forecast

```
p_uol <- predict(forecast2, uol[uol$fyear==2018,])
p_base <- predict(baseline,
  df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear==2018,
p_macro <- predict(macro3,
  df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear==2018,
p_world <- predict(forecast4,
  df_clean[df_clean$isin=="SG1S83002349" & df_clean$fyear==2018,])
preds <- c(p_uol, p_base, p_macro, p_world)
names(preds) <- c("UOL 2019 UOL", "UOL 2019 Base", "UOL 2019 Macro",
  "UOL 2019 World")
preds
```

```
##      UOL 2019 UOL      UOL 2019 Base UOL 2019 Macro UOL 2019 World
##      2325.326      2379.430      2401.298      2882.955
```

# Visualizing our prediction

- I plot 2019 forecast separately from other years' forecast
- I also plot the actual revenue for comparison
- Click the legend to mute it



# In Sample Accuracy

```
# series data is calculated, see the R code file
# Root Mean Square Error (RMSE)
# the st. deviation of the residuals (prediction errors).
rmse <- function(v1, v2) {
  sqrt(mean((v1 - v2)^2, na.rm=T))
}

rmse <- c(rmse(actual_series, uol_series),
          rmse(actual_series, base_series),
          rmse(actual_series, macro_series),
          rmse(actual_series, world_series))
names(rmse) <- c("UOL 2019 UOL", "UOL 2019 Base",
                 "UOL 2019 Macro", "UOL 2019 World")

rmse
```

```
##      UOL 2019 UOL      UOL 2019 Base UOL 2019 Macro UOL 2019 World
##      189.2207      273.6917      300.6274      349.6541
```

Why is UOL the best for in sample?

UOL is trained to minimize variation only in that context. It is potentially overfitted, meaning it won't predict well *out of sample*. Out of sample prediction is much more useful than in sample, however.

# Summary of Session 2

# For next week

- Try to replicate the code
- Continue your Datacamp career track
- First individual assignment
  - Do this one individually!
  - Assignment on eLearn at midnight today
  - Turn in on eLearn by midnight of next Monday