

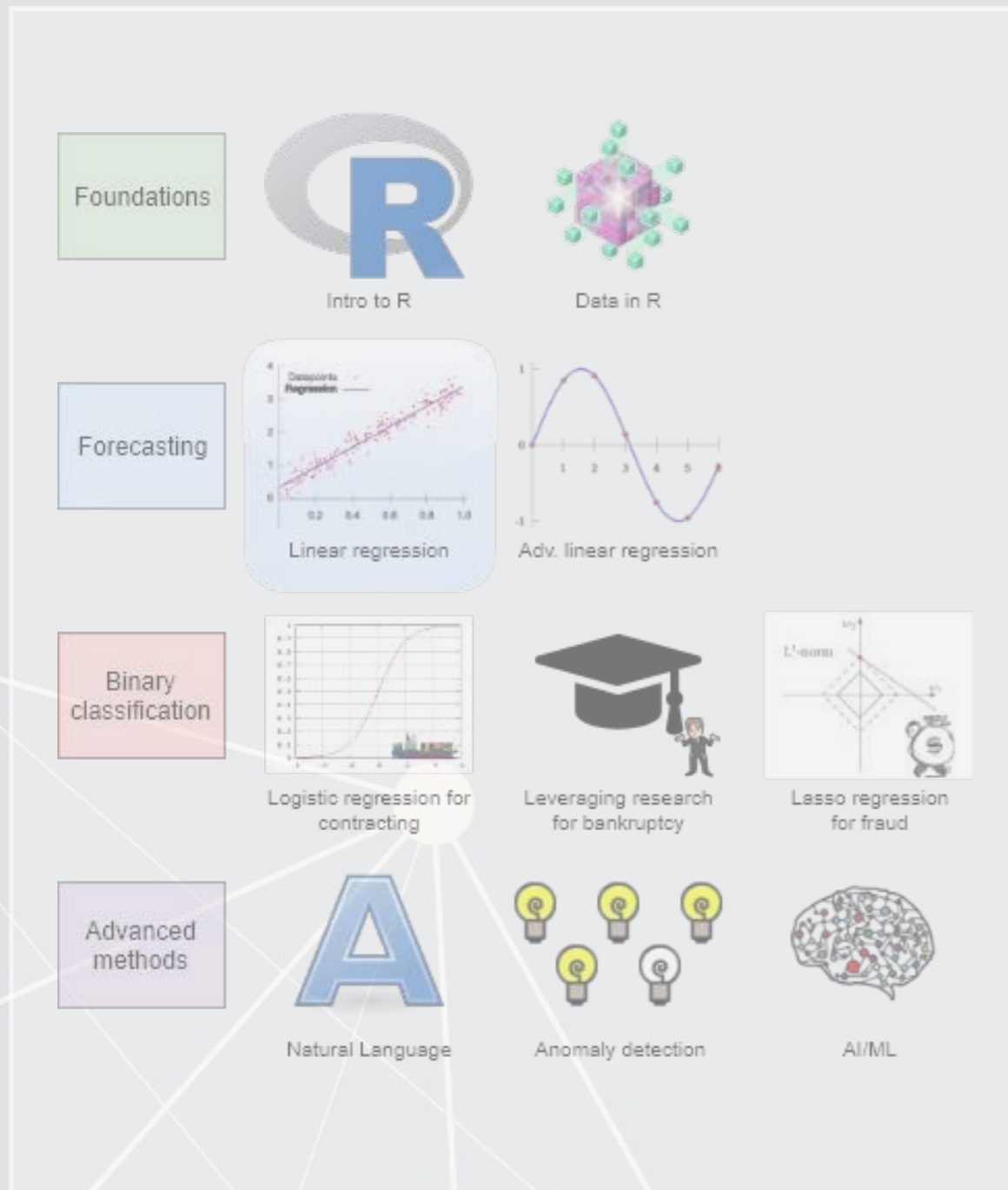
ACCT 420: Linear Regression

Session 3

Dr. Richard M. Crowley

Front matter

Learning objectives



■ Theory:

- Develop a logical approach to problem solving with data
- Hypothesis testing

■ Application:

- Predicting revenue for real estate firms

■ Methodology:

- Univariate stats
- Linear regression
- Visualization

Datacamp

- For next week:
 - Just 1 chapter on linear regression
- The full list of Datacamp materials for the course is up on eLearn

R Installation

- If you haven't already, make sure to install R and R Studio!
 - Instructions are in Session 1's slides
 - You will need it for this week's individual
- Please install a few packages using the following code
 - These packages are also needed for the first assignment
 - You are welcome to explore other packages as well, but those will not be necessary for now

```
# Run this in the R Console inside RStudio  
install.packages(c("tidyverse", "plotly", "tufte", "reshape2"))
```

- The individual assignment will be provided as an R Markdown file

The format will generally all be filled out – you will just add to it, answer questions, analyze data, and explain your work. Instructions and hints are in the same file


Assignments for this course

- Assignments will be posted online after the following lectures:
 1. Session 3, 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

For each assignment, you will have until the following Thursday at 11:59pm to finish it (9 days)

- Based on feedback received the following Tuesday, I may host extra office hours on Wednesday

R Markdown: A quick guide

- Headers and subheaders start with # and ##, respectively
- Code blocks starts with `{r}` and end with `}`
 - 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

How can we predict revenue for a company, leveraging data about that company, related companies, and macro factors

- Specific application: Real estate companies



More specifically...

- Can we use a company's own accounting data to predict its future revenue?
- Can we use other companies' accounting data to better predict all of their future revenue?
- Can we augment this data with macro economic data to further improve prediction?
 - Singapore business sentiment data

Linear models

What is a linear model?

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

- The 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
 - α and β are solved for
 - ε is the error in the measurement

I will refer to this as an *OLS* model – **Ordinary Least Square regression**

Example

Let's predict UOL's revenue for 2016



- Compustat has data for them since 1989
 - Complete since 1994
 - Missing CapEx before that

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

```
##      revt      at  
## Min.   : 94.78  Min.   : 1218  
## 1st Qu.: 193.41 1st Qu.: 3044  
## Median : 427.44 Median : 3478  
## Mean   : 666.38 Mean   : 5534  
## 3rd Qu.:1058.61 3rd Qu.: 7939  
## Max.   :2103.15 Max.   :19623
```

Linear models in R

- To run a linear model, use `lm()`
 - The first argument is a formula for your model, where `~` 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 `*`
 - I.e., `A*B` includes, A, B, and A times B in the model
 - Interactions using `:`
 - I.e., `A:B` just includes A times B in the model

Example:

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

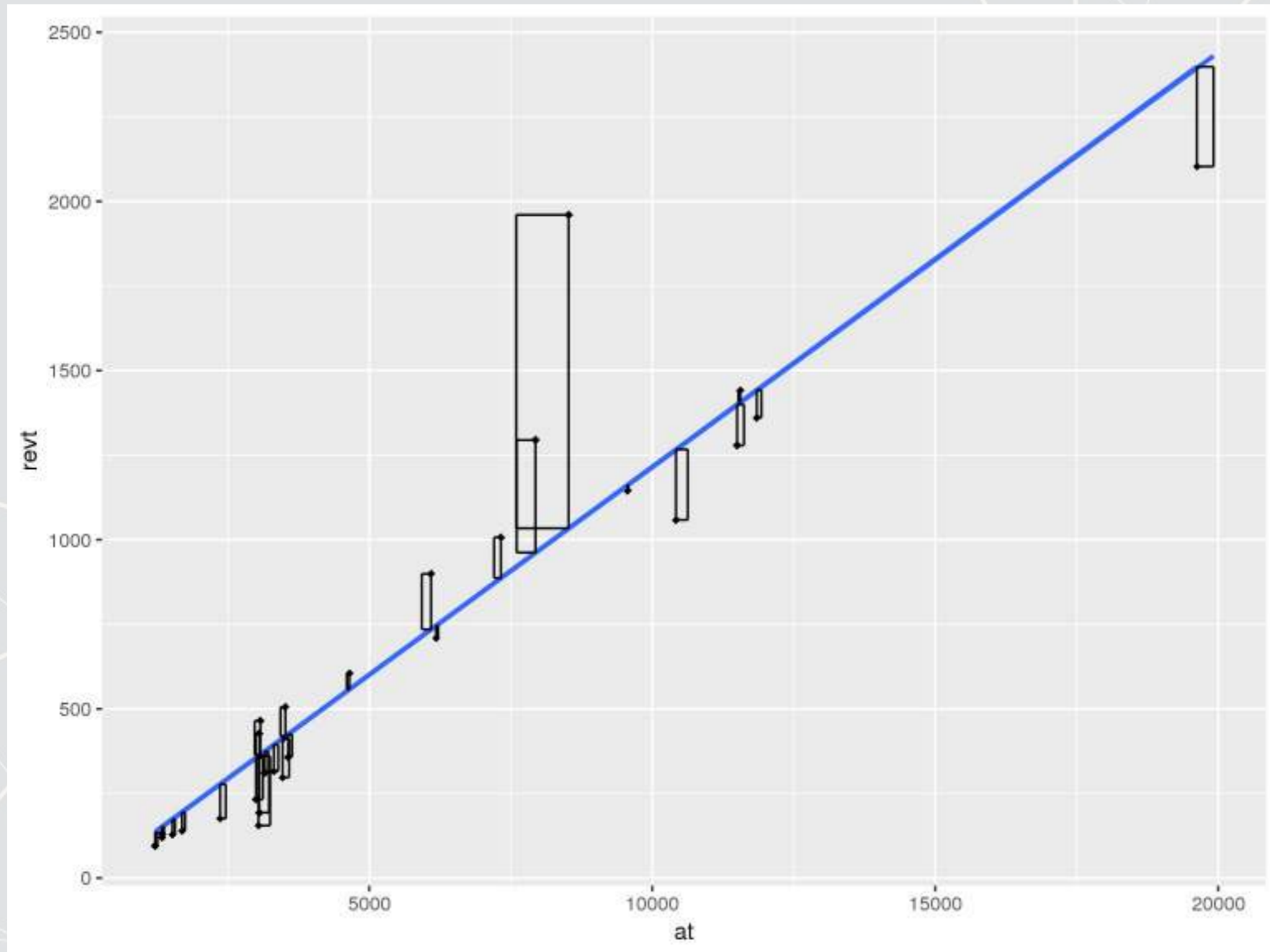
Example: UOL

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

```
##
## Call:
## lm(formula = revt ~ at, data = uol)
##
## Residuals:
##   Min     1Q   Median     3Q    Max
## -295.01 -101.29 -41.09  47.17  926.29
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.831399  67.491305  -0.205   0.839
## at           0.122914   0.009678  12.701 6.7e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 221.2 on 27 degrees of freedom
## Multiple R-squared:  0.8566, Adjusted R-squared:  0.8513
## F-statistic: 161.3 on 1 and 27 DF, p-value: 6.699e-13
```

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

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

```
# tidyverse  
uol <- uol %>%  
  mutate(revt_growth1 = revt / lag(revt) - 1)
```

```
# R way  
uol$revt_growth2 = uol$revt / c(NA, uol$revt[-length(uol$revt)]) - 1  
  
identical(uol$revt_growth1, uol$revt_growth2)
```

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

```
# faster with in place creation  
library(magrittr)  
uol %<>% mutate(revt_growth3 = revt / lag(revt) - 1)  
identical(uol$revt_growth1, uol$revt_growth3)
```

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

You can use whichever you are comfortable with

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
 - It's way more than needed for a simple ROA though.

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
## -0.57736 -0.10534 -0.00953  0.15132  0.42284
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.09024   0.05620   1.606  0.1204
## at_growth    0.53821   0.27717   1.942  0.0631 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2444 on 26 degrees of freedom
## (1 observation deleted due to missingness)
## Multiple R-squared:  0.1267, Adjusted R-squared:  0.09307
## F-statistic: 3.771 on 1 and 26 DF, p-value: 0.06307
```

Example: UOL with changes

- Δ Assets doesn't capture Δ 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 β_i is something we will solve for
- \hat{y} , α , and ε are the same as before

Scaling up our model

We have... 464 variables from Compustat Global alone!

- Let's just add them all?
- We only have 28 observations...
 - $28 \ll 464$...

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!

Scaling up our model

- Some potential sources to consider:
 - Direct accounting relations
 - Financing and expenditures
 - Business management
 - Some management characteristics may 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?

That's a lot!

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)) # From dplyr
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
## -0.46531 -0.15097  0.00205  0.17601  0.31997
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.07498    0.04915   1.526  0.14018
## lct_growth   0.23482    0.07319   3.209  0.00376 **
## che_growth  -0.11561    0.09227  -1.253  0.22230
## ebit_growth  0.03808    0.02208   1.724  0.09751 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2228 on 24 degrees of freedom
## (1 observation deleted due to missingness)
```

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?
2. Hypothesis
 - What do we think will happen? Build a model
3. Prediction
 - What exactly will we test? Formalize model into a statistical approach
4. Testing
 - Test the model
5. Analysis
 - Did it work?

Hypotheses

- Null hypothesis, a.k.a. H_0
 - The status quo
 - Typically: The model *doesn't* work
- Alternative hypothesis, a.k.a. H_1 or H_A
 - The model *does* work (and perhaps how it works)

We will use test statistics to test the hypotheses

Test statistics

- Testing a coefficient:
 - Use a t or z test
- Testing a model as a whole
 - F -test, check *adjusted* R squared as well
 - $\text{Adj } R^2$ tells us the amount of variation captured by the model (higher is better), after adjusting for the number of variables included
 - Otherwise, more variables (almost) always equals a higher amount of variation captured
- Testing across models
 - Chi squared (χ^2) test
 - Vuong test (comparing R^2)
 - **Akaike Information Criterion** (AIC) (Comparing MLEs, lower is better)

Revisiting the previous problem

Formalizing our last test

1. Question



2. Hypotheses

- H_0 :

- H_1 :

3. Prediction



4. Testing:



5. Statistical tests:

- Individual variables:

- Model:

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. Prediction

- Use an OLS model

4. Same results as before

5. Statistical tests:

- Individual variables
 - Growth in current liabilities loads (+) at $p < 0.01$ using a t-test
 - Growth in EBIT loads (+) at $p < 0.10$ using a t-test
- Model: F-test is significant at $p < 0.05$

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     26 1.5534
## 2     24 1.1918  2  0.36168  0.0262 *
## ---
## 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.

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 578 observations!  
print(c(nrow(df), nrow(df_clean)))
```

```
## [1] 5161 4583
```

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

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) # Let's us view bigger 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)  87.4    124.     0.707 0.486
## 2 lct          0.213    0.291    0.731 0.472
## 3 che          0.112    0.349    0.319 0.752
## 4 ebit         2.49     1.03     2.42 0.0236
```

```
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   0.655      0.612 357.    15.2 9.39e-6   4 -202.  414.  421.
## # ... with 2 more variables: deviance <dbl>, df.residual <int>
```

This model is ok, but 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) 15.6      97.0     0.161 0.874  
## 2 revt        1.49     0.414     3.59 0.00174  
## 3 act         0.324    0.165     1.96 0.0629  
## 4 che         0.0401   0.310     0.129 0.898  
## 5 lct        -0.198    0.179    -1.10 0.283  
## 6 dp         3.63     5.42     0.669 0.511  
## 7 ebit       -3.57     1.36    -2.62 0.0161
```

- Revenue to capture stickiness of revenue
- Current assest & Cash (and equivalents) to capture asset base
- Current liabilities to capture payments due
- Depreciation 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
## *   <dbl>      <dbl> <dbl>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1   0.903      0.875 203.    32.5 1.41e-9   7 -184. 385. 396.
## # ... with 2 more variables: deviance <dbl>, df.residual <int>
```

```
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     24 3059182
## 2     21  863005 3  2196177 1.477e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

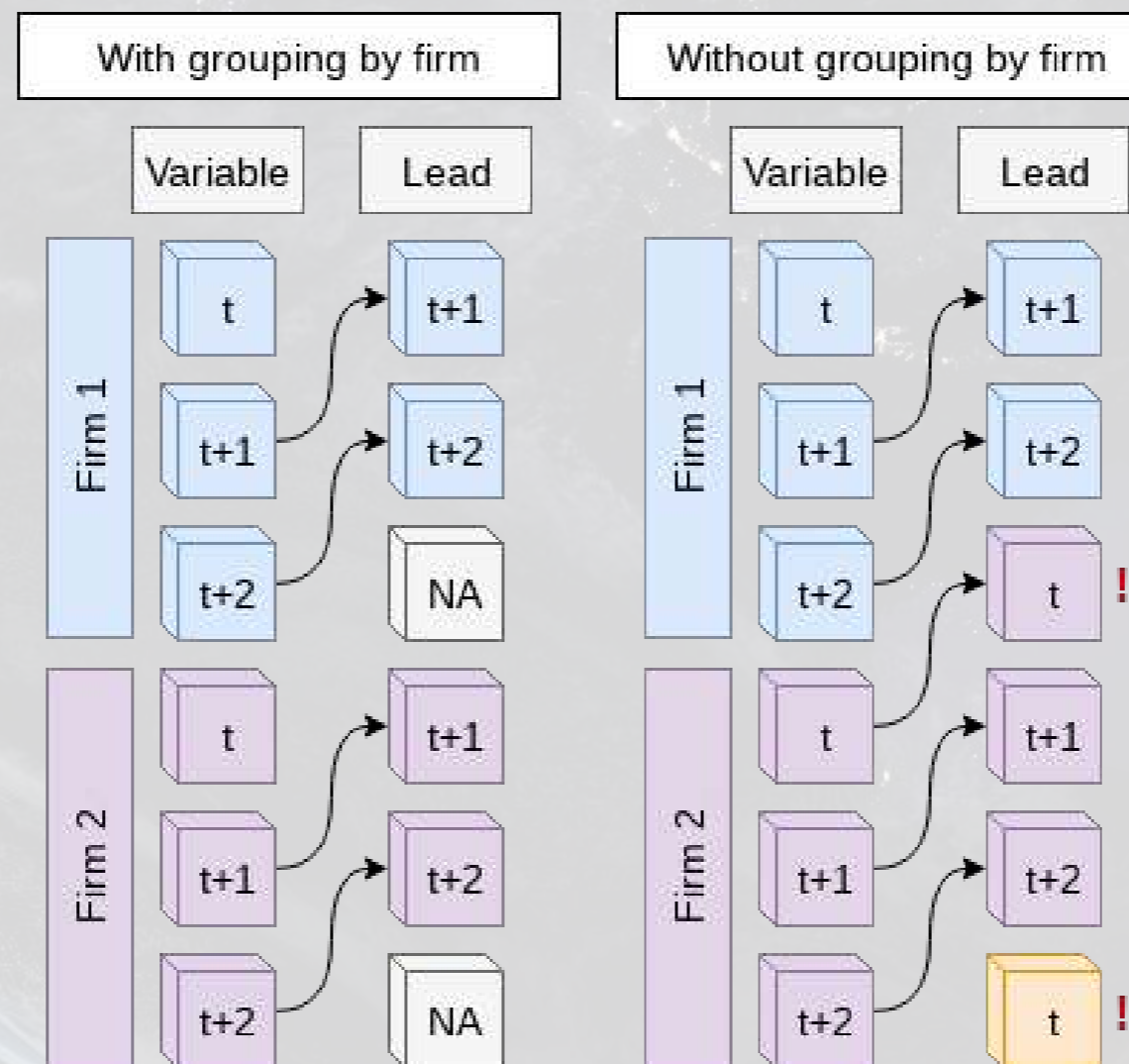
This is better (Adj. R^2 , χ^2 ,
AIC).

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
 - Wide: Indexed only by other dimensions

All Singapore real estate companies

```
# Note the 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()
```



All Singapore 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) 25.0     13.2     1.89 5.95e- 2  
## 2 revt        0.505    0.0762    6.63 1.43e-10  
## 3 act       -0.0999   0.0545   -1.83 6.78e- 2  
## 4 che        0.494    0.155     3.18 1.62e- 3  
## 5 lct        0.396    0.0860    4.60 5.95e- 6  
## 6 dp         4.46     1.55     2.88 4.21e- 3  
## 7 ebit      -0.951    0.271   -3.51 5.18e- 4
```

All Singapore real estate companies

```
glance(forecast3)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic p.value df logLik AIC
## *   <dbl>         <dbl> <dbl>   <dbl>   <dbl> <int> <dbl> <dbl>
## 1   0.844         0.841 210.    291. 2.63e-127   7 -2237. 4489.
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

Lower adjusted R^2 – This is worse?
Why?

- Note: χ^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) 222.    585.     0.379 7.04e- 1  
## 2 revt      0.997   0.00655 152.    0.  
## 3 act     -0.00221 0.00547  -0.403 6.87e- 1  
## 4 che     -0.150   0.0299  -5.02  5.36e- 7  
## 5 lct      0.0412  0.0113   3.64  2.75e- 4  
## 6 dp       1.52    0.184   8.26  1.89e-16  
## 7 ebit     0.308   0.0650   4.74  2.25e- 6
```

Worldwide real estate companies

```
glance(forecast4)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic p.value   df logLik   AIC
## *   <dbl>         <dbl> <dbl>   <dbl> <dbl> <int> <dbl> <dbl>
## 1   0.944         0.944 36459.  11299.    0     7 -47819. 95654.
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

Higher adjusted R^2 –
better!

- Note: χ^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

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
 - Control for this using a *Fixed Effect*
 - Note: ISIN uniquely identifies companies

```
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: 27 x 5  
##   term      estimate std.error statistic  p.value  
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>  
## 1 (Intercept)  1.58     39.4     0.0401  0.968  
## 2 revt         0.392    0.0977    4.01   0.0000754  
## 3 act        -0.0538   0.0602   -0.894  0.372  
## 4 che         0.304    0.177     1.72   0.0869  
## 5 lct         0.392    0.0921    4.26   0.0000276  
## 6 dp          4.71     1.73     2.72   0.00687  
## 7 ebit        -0.851    0.327    -2.60   0.00974  
## # ... with 20 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
## *   <dbl>      <dbl> <dbl>  <dbl>  <dbl> <int> <dbl> <dbl>
## 1   0.856      0.844 208.   69.4 1.15e-111  27 -2223. 4502.
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

```
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     324 14331633
## 2     304 13215145 20  1116488  0.1765
```

This isn't much different. Why? There is another source of noise within Singapore real estate companies

Another way to do fixed effects

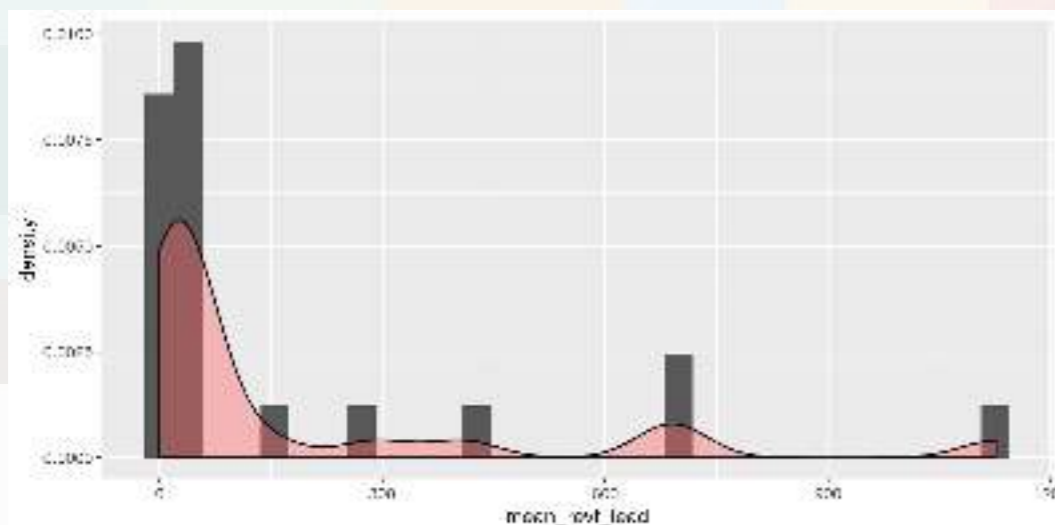
- The library `lfe` has `felm()`: fixed effects linear model
 - Better for complex models

```
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)
```

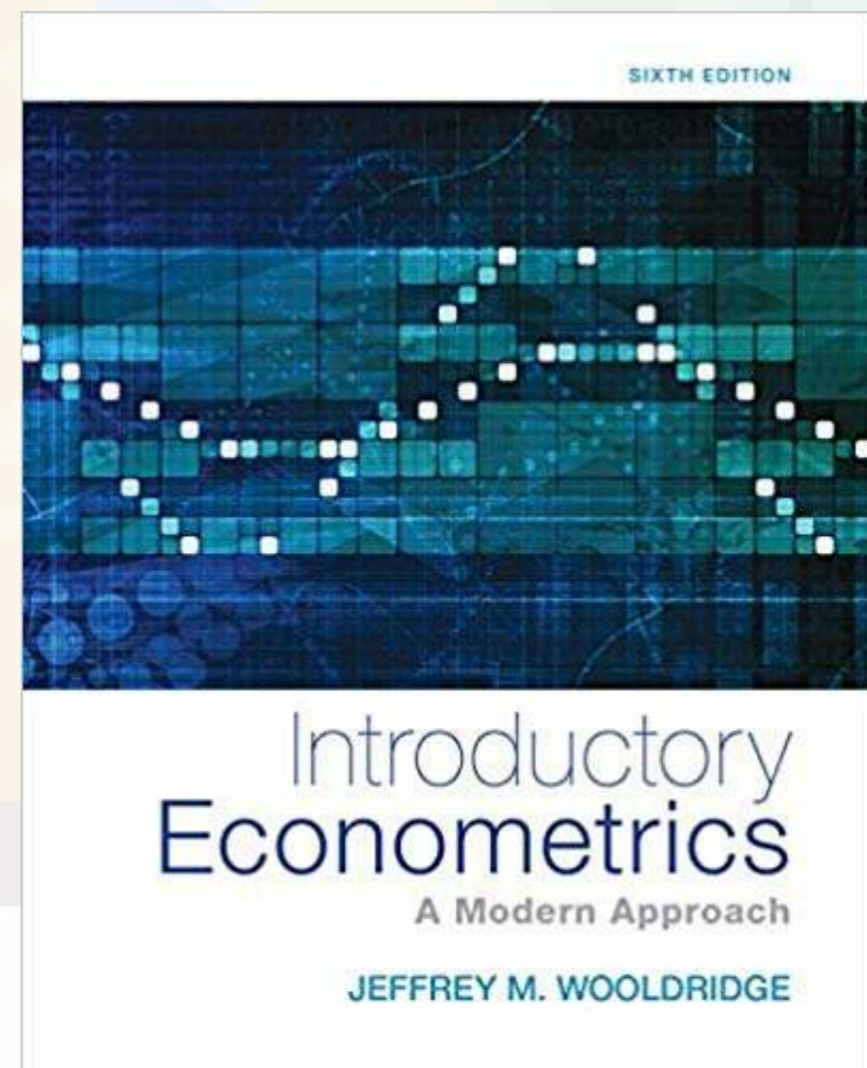
```
##
## Call:
## felm(formula = revt_lead ~ revt + act + che + lct + dp + ebit | factor(isin), data = df_clean[df_clean$
##
## Residuals:
##   Min     1Q  Median     3Q    Max
## -1181.88  -23.25  -1.87   18.03  1968.86
##
## Coefficients:
##   Estimate Std. Error t value Pr(>|t|)
## revt  0.39200   0.09767   4.013 7.54e-05 ***
## act  -0.05382   0.06017  -0.894  0.37181
## che   0.30370   0.17682   1.718  0.08690 .
## lct   0.39209   0.09210   4.257 2.76e-05 ***
## dp    4.71275   1.73168   2.721  0.00687 **
## ebit  -0.85080   0.32704  -2.602  0.00974 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 208.5 on 204 degrees of freedom
```

Why exactly would we use fixed effects?

- 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



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, 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)

```
# Import the csv file
expectations <- read.csv("general-business-expectations-by-detailed-services-industry-quarterly.csv",
                          stringsAsFactors = FALSE)
# split the year and quarter
expectations$year <- as.numeric(substr(expectations$quarter, 1, 4))
expectations$quarter <- as.numeric(substr(expectations$quarter, 7, 7))
# cast value to numeric
expectations$value <- as.numeric(expectations$value)
```

```
# extract out Q1, finance only
expectations_avg <- filter(expectations, quarter == 1 & level_2 == "Financial & Insurance")
```

```
# build a finance-specific measure
expectations_avg <- expectations_avg %>%
  group_by(year) %>%
  mutate(value=mean(value, na.rm=TRUE)) %>%
  slice(1)
```

```
# rename the value column to something more meaningful
colnames(expectations_avg)[colnames(expectations_avg) == "value"] <- "fin_sentiment"
```

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

R: Merging and sorting

dplyr makes things easy

- For merging, use `dplyr`'s `*_join()` commands
 - `left_join()` for merging a dataset into another
 - `inner_join()` for keeping only matched observations
 - `outer_join()` for making all possible combinations
- For sorting, `dplyr`'s `arrange()` command is easy to use
 - For sorting in reverse, combine `arrange()` with `desc()`

Merging example

Merge in the finance sentiment data to our accounting data

```
# subset out our Singaporean data, since our macro data is Singapore-specific  
df_SG <- df_clean[df_clean$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"
```

Sorting example

```
expectations %>%  
  filter(quarter == 1) %>% # using dplyr  
  arrange(level_2, level_3, desc(year)) %>% # using dplyr  
  select(year, quarter, level_2, level_3, value) %>% # using dplyr  
  datatable(options = list(pageLength = 5), rownames=FALSE) # using DT
```

Show entries

Search:

year	quarter	level_2	level_3	value
2018	1	Accommodation & Food Services	Accommodation	-7
2017	1	Accommodation & Food Services	Accommodation	-15
2016	1	Accommodation & Food Services	Accommodation	-25
2015	1	Accommodation & Food Services	Accommodation	4
2014	1	Accommodation & Food Services	Accommodation	3

Showing 1 to 5 of 216 entries

Previous

1

2

3

4

5

...

44

Next

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)
```

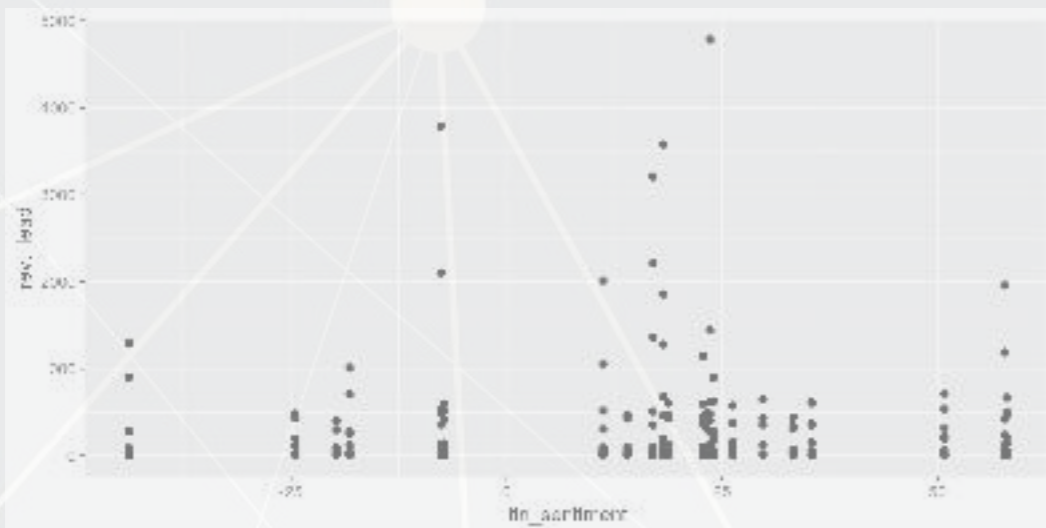
```
## # A tibble: 8 x 5  
##   term      estimate std.error statistic    p.value  
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>  
## 1 (Intercept) 24.0     15.9     1.50 0.134  
## 2 revt        0.497    0.0798    6.22 0.00000000162  
## 3 act       -0.102    0.0569   -1.79 0.0739  
## 4 che        0.495    0.167     2.96 0.00329  
## 5 lct        0.403    0.0903    4.46 0.0000114  
## 6 dp         4.54     1.63     2.79 0.00559  
## 7 ebit      -0.930    0.284    -3.28 0.00117  
## 8 fin_sentiment 0.122    0.472     0.259 0.796
```

It isn't significant. Why is this?

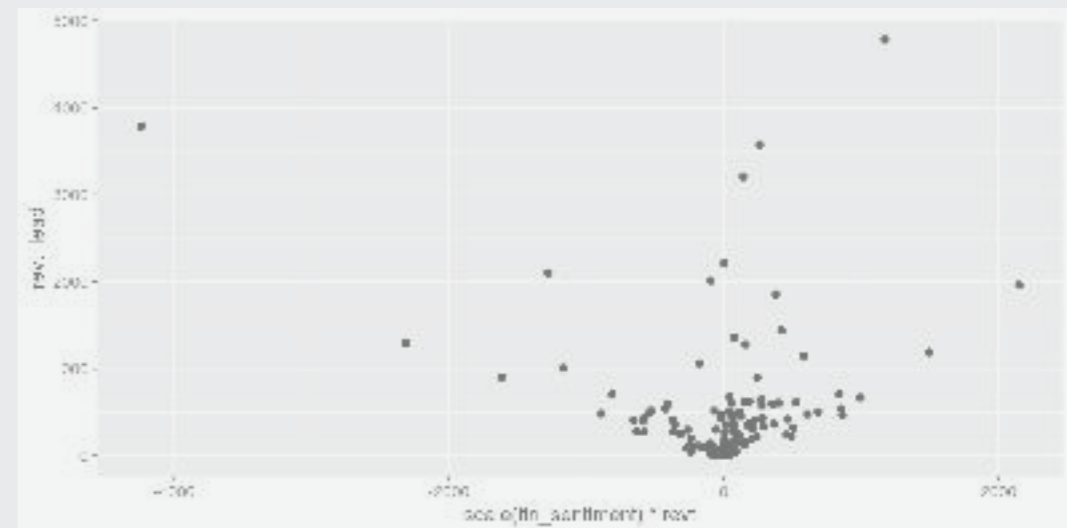
Scaling matters

- All of our firm data is on the same terms as revenue: dollars within a given firm
- But `fin_sentiment` is a constant scale...
 - Need to scale this to fit the problem
 - The current scale would work for revenue growth

```
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()
```



Scaled macro data

■ Normalize and scale by revenue

Scale creates z-scores, but returns a matrix by default. [,1] gives a vector

```
df_SG_macro$fin_sent_scaled <- scale(df_SG_macro$fin_sentiment)[,1]
```

```
macro3 <-
```

```
lm(revt_lead ~ revt + act + che + lct + dp + ebit + fin_sent_scaled:revt,  
   data=df_SG_macro)
```

```
tidy(macro3)
```

```
## # A tibble: 8 x 5
```

## term	estimate	std.error	statistic	p.value
## <chr>	<dbl>	<dbl>	<dbl>	<dbl>
## 1 (Intercept)	25.5	13.8	1.84	0.0663
## 2 revt	0.490	0.0789	6.21	0.00000000170
## 3 act	-0.0677	0.0576	-1.18	0.241
## 4 che	0.439	0.166	2.64	0.00875
## 5 lct	0.373	0.0898	4.15	0.0000428
## 6 dp	4.10	1.61	2.54	0.0116
## 7 ebit	-0.793	0.285	-2.78	0.00576
## 8 revt:fin_sent_scaled	0.0897	0.0332	2.70	0.00726

```
glance(macro3)
```

```
## # A tibble: 1 x 11
```

## r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC
## * <dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	<dbl>
## 1 0.847	0.844	215.	240.	1.48e-119	8	-2107.	4232.

... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>

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  
## *   <dbl>         <dbl> <dbl>   <dbl>   <dbl> <int> <dbl> <dbl>  
## 1   0.843         0.840 217.    273. 3.13e-119   7 -2111. 4237.  
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

```
glance(macro3)
```

```
## # A tibble: 1 x 11  
##   r.squared adj.r.squared sigma statistic p.value  df logLik  AIC  
## *   <dbl>         <dbl> <dbl>   <dbl>   <dbl> <int> <dbl> <dbl>  
## 1   0.847         0.844 215.    240. 1.48e-119   8 -2107. 4232.  
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

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    304 14285622
## 2    303 13949301 1    336321 0.006875 **
## ---
## 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` (25.7 points)
 - Revenue stickiness increases by ~9%
- Over the range of data (-43.8 to 58)...
 - Revenue stickiness ranges from -6.3% to 29.2%

Scaling up our model, again

Building a model requires careful thought!

- What macroeconomic data makes sense to add to our model?

This is where having accounting and business knowledge comes in!

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*

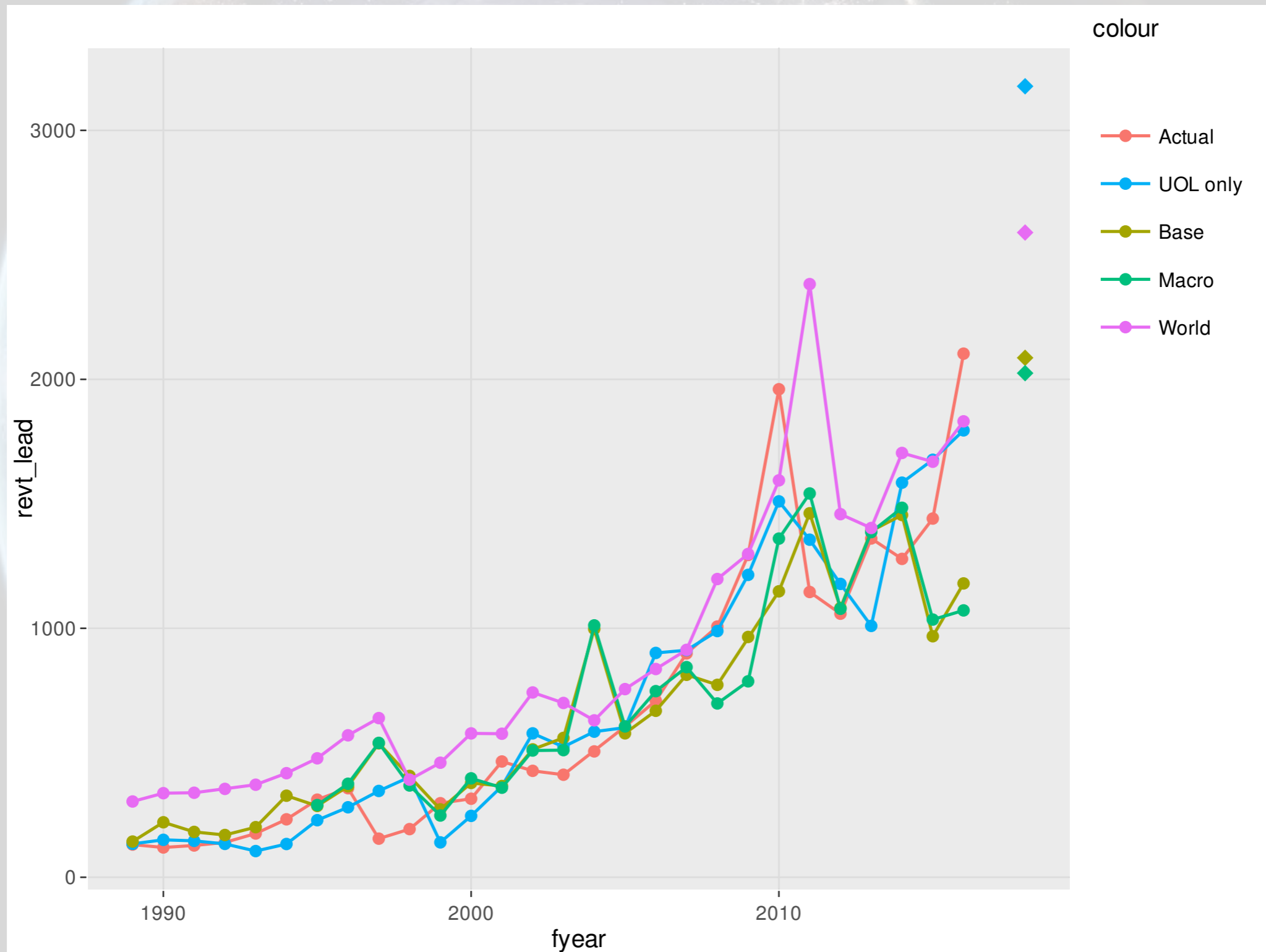
Estimation

- As we never constructed a hold out sample, let's end by estimating UOL's *2018* year revenue
- This should be announced in February 2019...

```
p_uol <- predict(forecast2, uol[uol$year==2017,])
p_base <- predict(baseline,
  df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$year==2017,])
p_macro <- predict(macro3,
  df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$year==2017,])
p_world <- predict(forecast4,
  df_clean[df_clean$isin=="SG1S83002349" & df_clean$year==2017,])
preds <- c(p_uol, p_base, p_macro, p_world)
names(preds) <- c("UOL 2018 UOL", "UOL 2018 Base", "UOL 2018 Macro",
  "UOL 2018 World")
preds
```

```
## UOL 2018 UOL UOL 2018 Base UOL 2018 Macro UOL 2018 World
## 3177.073 2086.437 2024.842 2589.636
```


Visualizing our prediction



In Sample Accuracy

```
# series vectors calculated here -- See appendix
```

```
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 2018 UOL", "UOL 2018 Base", "UOL 2018 Macro", "UOL 2018 World")  
rmse
```

```
## UOL 2018 UOL UOL 2018 Base UOL 2018 Macro UOL 2018 World  
## 175.5609 301.3161 344.9681 332.8101
```

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.

End matter



For next week

- For next week:
 - 1 chapter of 1 course on Datacamp
 - First individual assignment
 - Do this one individually!
 - Turn in on eLearn by the end of next Thursday

Packages used for these slides

- broom
- DT
- knitr
- lfe
- magrittr
- plotly
- revealjs
- tidyverse

Custom code

```
# Graph showing squared error (slide 4.6)
```

```
uolg <- uol[,c("at","revt")]  
uolg$resid <- mod1$residuals  
uolg$xleft <- ifelse(uolg$resid < 0,uolg$at,uolg$at - uolg$resid)  
uolg$xright <- ifelse(uolg$resid < 0,uolg$at - uolg$resid, uol$at)  
uolg$ytop <- ifelse(uolg$resid < 0,uolg$revt - uolg$resid,uol$revt)  
uolg$ybottom <- ifelse(uolg$resid < 0,uolg$revt, uolg$revt - uolg$resid)  
uolg$point <- TRUE
```

```
uolg2 <- uolg  
uolg2$point <- FALSE  
uolg2$at <- ifelse(uolg$resid < 0,uolg2$xright,uolg2$xleft)  
uolg2$revt <- ifelse(uolg$resid < 0,uolg2$ytop,uolg2$ybottom)
```

```
uolg <- rbind(uolg, uolg2)
```

```
uolg %>% ggplot(aes(y=revt, x=at, group=point)) +  
  geom_point(aes(shape=point)) +  
  scale_shape_manual(values=c(NA,18)) +  
  geom_smooth(method="lm", se=FALSE) +  
  geom_errorbarh(aes(xmax=xright, xmin = xleft)) +  
  geom_errorbar(aes(ymax=ytop, ymin = ybottom)) +  
  theme(legend.position="none")
```

```
# Chart of mean revt_lead for Singaporean firms (slide 7.19)
```

```
df_clean %>% # Our data frame  
  filter(fic=="SGP") %>% # Select only Singaporean firms  
  group_by(isin) %>% # Group by firm  
  mutate(mean_revt_lead=mean(revt_lead, na.rm=T)) %>% # Determine each firm's mean revenue (lead)  
  slice(1) %>% # Take only the first observation for each group  
  ungroup() %>% # Ungroup (we don't need groups any more)  
  ggplot(aes(x=mean_revt_lead)) + # Initialize plot and select data  
  geom_histogram(aes(y = ..density..)) + # Plots the histogram as a density so that geom_density is visible  
  geom_density(alpha=.4, fill="#FF6666") # Plots smoothed density
```

Custom code

Chart of predictions (slide 11.4)

```
library(plotly)
df_SG_macro$pred_base <- predict(baseline, df_SG_macro)
df_SG_macro$pred_macro <- predict(macro3, df_SG_macro)
df_clean$pred_world <- predict(forecast4, df_clean)
uol$pred_uol <- predict(forecast2, uol)
df_preds <- data.frame(preds=preds, fyear=c(2018,2018,2018,2018), model=c("UOL only", "Base", "Macro", "World"))
plot <- ggplot() +
  geom_point(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=revt_lead,x=fyear, color="Actual")) +
  geom_line(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=revt_lead,x=fyear, color="Actual")) +
  geom_point(data=uol[uol$fyear < 2017,], aes(y=pred_uol,x=fyear, color="UOL only")) +
  geom_line(data=uol[uol$fyear < 2017,], aes(y=pred_uol,x=fyear, color="UOL only")) +
  geom_point(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=pred_base,x=fyear, color="Base")) +
  geom_line(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=pred_base,x=fyear, color="Base")) +
  geom_point(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=pred_macro,x=fyear, color="Macro")) +
  geom_line(data=df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,], aes(y=pred_macro,x=fyear, color="Macro")) +
  geom_point(data=df_clean[df_clean$isin=="SG1S83002349" & df_clean$fyear < 2017,], aes(y=pred_world,x=fyear, color="World")) +
  geom_line(data=df_clean[df_clean$isin=="SG1S83002349" & df_clean$fyear < 2017,], aes(y=pred_world,x=fyear, color="World")) +
  geom_point(data=df_preds, aes(y=preds, x=fyear, color=model), size=1.5, shape=18)
ggplotly(plot)
```

Calculating Root Mean Squared Error (Slide 11.5)

```
actual_series <- df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,]$revt_lead
uol_series <- uol[uol$fyear < 2017,]$pred_uol
base_series <- df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,]$pred_base
macro_series <- df_SG_macro[df_SG_macro$isin=="SG1S83002349" & df_SG_macro$fyear < 2017,]$pred_macro
world_series <- df_clean[df_clean$isin=="SG1S83002349" & df_clean$fyear < 2017,]$pred_world

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 2018", "UOL only", "UOL 2018 Baseline", "UOL 2018 w/ macro", "UOL 2018 w/ world")
rmse
```