HW2

Erik Andersen

2024-04-18

```r
here::i_am("HW2/code/HW2.Rmd")
```

```
## here() starts at /Users/erikandersen/Documents/Classes/FIN-592
```

```r
# Load packages
pacman::p_load(tidyverse, magrittr, sandwich, kableExtra)
```

```r
# Load data
equity_df = read_csv(here::here("HW2", "data", "equity_returns.csv"))
```

```
## Rows: 99 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (2): vwretd, vwretx
## date (1): caldt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
tbill_df = read_csv(here::here("HW2", "data", "tbill_returns.csv"))
```

```
## Rows: 99 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): t90ret
## date (1): caldt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Merge into one data set
returns_df = left_join(equity_df, tbill_df) |>
  mutate(date = year(caldt)) |> # Extract the year
  select(date, everything(), -caldt) |>
  filter(!is.na(vwretd)) # Remove 1925 because it has no data
```

```
## Joining with `by = join_by(caldt)`
```

```r

# Construct dividend price ratio from equity returns data
# To get there, we will do the following. First, subtracting returns without dividends from returns wit

# Dividend growth we can construct by multiplying current dividend price by returns without dividends d

returns_df =
```

```r
  returns_df |>
    mutate(div_price = (vwretd + 1) /(vwretx + 1) - 1,
           div_growth = div_price / lag(div_price) + vwretx)
```

b)

```r
# Plot dividend price, dividend returns, stock returns, and risk free returns on the same graph

returns_plot = returns_df |>
  drop_na() |>
  mutate(across(-c(date, div_growth), function(x) x*100)) |> # Convert to %s
  select(-c(vwretx)) |> # Its redundant to plot returns with and without dividends
  pivot_longer(cols = -date) |>  # Rearrange the data so it makes a nice legend
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  cowplot::theme_cowplot() +
  scale_color_brewer(name = "", palette = "Dark2", labels = c("Dividend Growth", "Dividend Price Ratio"
  labs(y = "Return (%)", x = "", title = "Returns Since 1926" )

ggsave(here::here("HW2", "plots", "returns.pdf"))
```

c)

```
## Saving 6.5 x 4.5 in image
```

Question 2

```r
# Calculate cumulative returns for every horizon from 1 to 10 years.
# I couldn't come up with a good way to do this so I'm doing it manually
horizons_return_df = returns_df |>
  mutate(demean = vwretd + 1,
         r1 = demean * lead(demean)-1,
         r2 = (r1 + 1)*lead(demean,2)-1,
         r3 = (r2 + 1)*lead(demean,3)-1,
         r4 = (r3 + 1)*lead(demean,4)-1,
         r5 = (r4 + 1)*lead(demean,5)-1,
         r6 = (r5 + 1)*lead(demean,6)-1,
         r7 = (r6 + 1)*lead(demean,7)-1,
         r8 = (r7 + 1)*lead(demean,8)-1,
         r9 = (r8 + 1)*lead(demean,9)-1,
         r10= (r9 + 1)*lead(demean,10)-1)

# Calculate the regressions for various return horizons on dividend price ratio
return_horizons = sapply(paste0("r", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(return_horizons) = 1:10
```

```r
rownames(return_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")

# Add excess returns
horizons_return_df %<>%
  mutate(rf = t90ret + 1,
         rf1 = r1 - (rf * lead(rf)-1),
         rf2 = r2 - (rf1 + 1) * lead(rf,2)-1,
         rf3 = r3 - (rf2 + 1) * lead(rf,3)-1,
         rf4 = r4 - (rf3 + 1) * lead(rf,4)-1,
         rf5 = r5 - (rf4 + 1) * lead(rf,5)-1,
         rf6 = r6 - (rf5 + 1) * lead(rf,6)-1,
         rf7 = r7 - (rf6 + 1) * lead(rf,7)-1,
         rf8 = r8 - (rf7 + 1) * lead(rf,8)-1,
         rf9 = r9 - (rf8 + 1) * lead(rf,9)-1,
         rf10=r10 - (rf9 + 1) * lead(rf,10)-1)

# Calculate regressions
risk_free_horizons = sapply(paste0("rf", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(risk_free_horizons) = 1:10
rownames(risk_free_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")

# Add Dividend growth
horizons_return_df %<>%
  mutate(d1 = div_growth * lead(div_growth),
         d2 = d1 * lead(div_growth,2),
         d3 = d2 * lead(div_growth,3),
         d4 = d3 * lead(div_growth,4),
         d5 = d4 * lead(div_growth,5),
         d6 = d5 * lead(div_growth,6),
         d7 = d6 * lead(div_growth,7),
         d8 = d7 * lead(div_growth,8),
         d9 = d8 * lead(div_growth,9),
         d10= d9 * lead(div_growth,10))

# Calculate regrssions
div_growth_horizons = sapply(paste0("d", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(div_growth_horizons) = 1:10
```

```r
rownames(div_growth_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")
```

```r
# Calculate the Hansen Hoddrick standard errors
hansen_hodrick = sapply(paste0("r", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |>
    kernHAC(kernel = 'Truncated', prewhite = F, adjust = T, sandwich = T)

  se = tibble(se = sqrt(temp[2,2]))

  return(se)
}) |> unlist()

# Calcualte t-stats
hh_t = as.numeric(return_horizons[2,])/hansen_hodrick
```

```r
# Calculate non-overlapping standard errors
# What we're doing in this loop is subsetting the data into chunks that are the length of the return we
no_overlap = sapply(1:10, function(i){
  temp = horizons_return_df %>%
    # This next bit is complicated and dense because it uses non dplyr fuctions so they don;t work in a
    # Starting from the inside, first we find the row number in the dataset. Why its only the date colu
    # Take the modulo of that for each legth of time. Then select only those rows for which that equals
    slice(which((row_number(horizons_return_df$date) -1) %% i == 0)) %>%
    lm(paste0("r", i, "~", "div_price"),.) |>
    summary() |>
    coef()

  se = temp[2,2]

  return(tibble(se))

}) |> unlist()

# Calculate t-stat
no_overlap_t = as.numeric(return_horizons[2,])/no_overlap
```

c)

Question 3

```r
# Predict returns for one year horizon using the returns_horizon object and plot
one_year_returns = horizons_return_df |>
  mutate(predicted_returns = as.numeric(return_horizons[2,1]) * div_price) |>
  select(date, vwretd, predicted_returns) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Predicted Returns", "Actual Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("One Year Horizon Predicted Versus Actual Returns") +
  cowplot::theme_cowplot()
```

```r
ggsave(here::here("HW2", "plots", "one_year_returns.png"))
```

```
## Saving 6.5 x 4.5 in image
```

```r
# Same thing but for dividend growth
# Predict returns for one year horizon using the div_growth_horizons object and plot
one_year_div = horizons_return_df |>
  mutate(predicted_div_growth = as.numeric(div_growth_horizons[1,1]) +  as.numeric(div_growth_horizons[2
  select(date, d1, predicted_div_growth) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Actual Returns", "Predicted Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("One Year Horizon Predicted Versus Actual Dividend Growth") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "one_year_div.png"))
```

```
## Saving 6.5 x 4.5 in image

## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```r
# Now graph the same things but over 7 year horizon
seven_year_returns = horizons_return_df |>
  mutate(predicted_returns = as.numeric(return_horizons[1, 7]) + as.numeric(return_horizons[2, 7]) * di
  select(date, r7, predicted_returns) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(date, value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Predicted Returns", "Actual Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("Seven Year Horizon Predicted Versus Actual Returns") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "seven_year_returns.png"))
```

```
## Saving 6.5 x 4.5 in image

## Warning: Removed 7 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```r
# Finally seven year horizon for dividend growth
seven_year_div = horizons_return_df |>
  mutate(predicted_div_growth = as.numeric(div_growth_horizons[1,7]) +  as.numeric(div_growth_horizons[2
  select(date, d7, predicted_div_growth) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Actual Returns", "Predicted Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("Seven Year Horizon Predicted Versus Actual Dividend Growth") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "sevel_year_div.png"))
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 8 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

Question 4

```
# Construct variance decomposition for the price dividend ratio. First we'll do the part for dividends
#
var_dividends = sapply(1:nrow(horizons_return_df), function(i){ # Loop over each row
  # Select each row because we want to calculate the discounted sum of dividend growth for each time pe
  temp = horizons_return_df[i,]
  sapply(1:10, function(j){ # Question asks for 10 year horizon
    # We already calculated the dividend growth up to 10 years indexed by d1:d10, so we just discount t
    as.numeric(0.96^(j-1) * temp[paste0("d", j)])
  }) |> sum()
})

# Calculate the relevant statistic
# We have to take the negative log of dividend price because that's that the table uses
div_decomp = 100 *cov(var_dividends,
                  -log(horizons_return_df$div_price),
                  # Complete.obs gets rid of all the NAs
                  "complete.obs") / var(-log(horizons_return_df$div_price), na.rm = T)
```

```
# Do the same thing for returns
var_returns = sapply(1:nrow(horizons_return_df), function(i){
  temp = horizons_return_df[i,]
  sapply(1:10, function(j){
    # Only difference is we iterate over returns indexed by r1:r10
    as.numeric(0.96^(j-1) * temp[paste0("r", j)])
  }) |> sum()
})

# Calculate stat
return_decomp = cov(var_returns,
                    horizons_return_df$div_price,
                    "complete.obs") / var(horizons_return_df$div_price, na.rm = T)
```