PREDICT STOCK RETURN

2022-11-11

```r
library(ranger)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidymodels)
```

```
## ── Attaching packages ──────────────────────────────────── tidymodels 1.0.0 ──
```

```
## ✔ broom        1.0.0     ✔ rsample      1.1.0
## ✔ dials        1.0.0     ✔ tibble       3.1.8
## ✔ ggplot2      3.3.6     ✔ tidyr        1.2.1
## ✔ infer        1.0.3     ✔ tune         1.0.0
## ✔ modeldata    1.0.1     ✔ workflows    1.1.0
## ✔ parsnip      1.0.2     ✔ workflowsets 1.0.0
## ✔ purrr        0.3.4     ✔ yardstick    1.1.0
## ✔ recipes      1.0.1
```

```
## ── Conflicts ─────────────────────────────────────────── tidymodels_conflicts() ──
## ✖ purrr::discard() masks scales::discard()
## ✖ dplyr::filter()  masks stats::filter()
## ✖ dplyr::lag()     masks stats::lag()
## ✖ recipes::step()  masks stats::step()
## • Search for functions across packages at https://www.tidymodels.org/find/
```

```r
library(tidyverse)
```

```
## ── Attaching packages
## ───────────────────────────────────────
## tidyverse 1.3.2 ──
```

```
## ✔ readr   2.1.2     ✔ forcats 0.5.2
## ✔ stringr 1.4.1
## ── Conflicts ──────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ readr::col_factor() masks scales::col_factor()
## ✖ purrr::discard()    masks scales::discard()
## ✖ dplyr::filter()     masks stats::filter()
## ✖ stringr::fixed()    masks recipes::fixed()
## ✖ dplyr::lag()        masks stats::lag()
## ✖ readr::spec()       masks yardstick::spec()
```

```r
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following objects are masked from 'package:yardstick':
##
##     precision, recall, sensitivity, specificity
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(roll)
```

Loading Data file

```r
setwd("D:/UNI/COURSES/Big Data with application to Finance/FIE423/merged/RET")

load("merged.RData")
```

Wrangling Data

```r
data <- merged %>%
  filter(date < 20200101 ) %>%
  transmute(PERMNO,
            date,
            RET,
            RET1 = lag(RET,1),
            RET2 = lag(RET,2),
            market_cap = MARKETCAP,
            price1 = lag(PRC),
            asset = atq,
            liability = ltq,
            book_value = coalesce(seqq, ceqq + pstkq, atq - ltq) +
               coalesce(txditcq, txdbq + intaccq, 0) -
               coalesce(pstkrq, pstknq, pstkq, 0),
            cash = chq,
            revenue = revtq,
            earning = req,
            EPS = epsf12,
            book_to_market = book_value/MARKETCAP,
            PE = PRC / lag(epsf12),
            volume = lag(VOL)
  ) %>% drop_na()

head(data,10)
```

```
##    PERMNO     date       RET      RET1      RET2 market_cap  price1   asset
## 1   10001 20110729 -0.028139  0.008261  0.028050   91892.82 11.5500 129.777
## 2   10001 20110831 -0.013864 -0.028139  0.008261   91892.82 11.1800 129.777
## 3   10001 20110930  0.005009 -0.013864 -0.028139   91892.82 10.9800 129.777
## 4   10001 20111031  0.005005  0.005009 -0.013864   94144.05 10.9900 137.756
## 5   10001 20111130 -0.005000  0.005005  0.005009   94144.05 11.0000 137.756
## 6   10001 20111230  0.051835 -0.005000  0.005005   94144.05 10.9000 137.756
## 7   10001 20120131 -0.019702  0.051835 -0.005000   89590.48 11.4200 144.642
## 8   10001 20120229  0.005005 -0.019702  0.051835   89590.48 11.1500 144.642
## 9   10001 20120330  0.048760  0.005005 -0.019702   89590.48 11.1608 144.642
## 10  10001 20120430 -0.015000  0.048760  0.005005   93118.68 11.6600 156.411
##    liability book_value   cash revenue earning  EPS book_to_market       PE
## 1     52.866     77.103 13.104  40.151  33.762 0.83   0.0008390536 12.15217
## 2     52.866     77.103 13.104  40.151  33.762 0.83   0.0008390536 13.22892
## 3     52.866     77.103 13.104  40.151  33.762 0.83   0.0008390536 13.24096
## 4     61.566     77.294 13.133  18.673  33.022 0.80   0.0008210184 13.25301
## 5     61.566     77.294 13.133  18.673  33.022 0.80   0.0008210184 13.62500
## 6     61.566     77.294 13.133  18.673  33.022 0.80   0.0008210184 14.27500
## 7     69.375     76.648 10.490  12.321  32.082 0.83   0.0008555373 13.93750
## 8     69.375     76.648 10.490  12.321  32.082 0.83   0.0008555373 13.44675
## 9     69.375     76.648 10.490  12.321  32.082 0.83   0.0008555373 14.04819
## 10    81.639     77.856 10.505  28.072  31.570 0.66   0.0008360943 13.78325
##    volume
## 1    2472
## 2    2860
## 3    5307
## 4    3086
## 5    1644
## 6    1996
## 7    1804
## 8    2697
## 9    2209
## 10   2405
```

```r
summary(data)
```

```
##      PERMNO           date               RET                  RET1
##  Min.   :10001   Min.   :20110531   Min.   :-0.993600   Min.   :-0.993600
##  1st Qu.:16466   1st Qu.:20130830   1st Qu.:-0.059984   1st Qu.:-0.060512
##  Median :77857   Median :20151030   Median : 0.003593   Median : 0.003146
##  Mean   :59577   Mean   :20153045   Mean   : 0.007191   Mean   : 0.006658
##  3rd Qu.:88646   3rd Qu.:20171130   3rd Qu.: 0.064113   3rd Qu.: 0.063745
##  Max.   :93436   Max.   :20191231   Max.   :19.883589   Max.   :19.883589
##       RET2             market_cap            price1             asset
##  Min.   :-0.993600   Min.   :1.070e+02   Min.   :     0.0   Min.   :      0.0
##  1st Qu.:-0.060796   1st Qu.:1.584e+05   1st Qu.:     5.9   1st Qu.:    172.8
##  Median : 0.002876   Median :7.332e+05   Median :    17.5   Median :    877.5
##  Mean   : 0.006272   Mean   :5.923e+06   Mean   :    88.6   Mean   :  11600.9
##  3rd Qu.: 0.063355   3rd Qu.:3.053e+06   3rd Qu.:    41.5   3rd Qu.:   4000.2
##  Max.   :19.883589   Max.   :1.073e+09   Max.   :330495.1   Max.   :1988226.0
##    liability           book_value           cash              revenue
##  Min.   :      0.0   Min.   :-11968.0   Min.   :     0.00   Min.   : -3935.00
##  1st Qu.:     54.6   1st Qu.:    75.2   1st Qu.:    14.71   1st Qu.:    22.66
##  Median :    423.7   Median :   355.0   Median :    62.77   Median :   132.66
##  Mean   :   8513.6   Mean   :  3245.7   Mean   :   635.55   Mean   :  1225.18
##  3rd Qu.:   2396.8   3rd Qu.:  1494.8   3rd Qu.:   243.30   3rd Qu.:   593.13
##  Max.   :1790116.0   Max.   :404478.0   Max.   :131417.00   Max.   :207307.33
##     earning               EPS             book_to_market
##  Min.   :-130761.2   Min.   :-416219.7   Min.   :-0.119889
##  1st Qu.:   -152.3   1st Qu.:     -0.3   1st Qu.: 0.000258
##  Median :     12.0   Median :      0.5   Median : 0.000519
##  Mean   :   1714.5   Mean   :     -3.8   Mean   : 0.001701
##  3rd Qu.:    490.8   3rd Qu.:      1.9   3rd Qu.: 0.000944
##  Max.   : 402089.0   Max.   :  21683.0   Max.   : 5.060650
##        PE                volume
##  Min.   :-10186.500   Min.   :       0
##  1st Qu.:    -3.702   1st Qu.:   16631
##  Median :    12.628   Median :   65422
##  Mean   :       Inf   Mean   :  249250
##  3rd Qu.:    25.010   3rd Qu.:  213082
##  Max.   :       Inf   Max.   :36787516
```

```r
#calculate NA of each variable

sapply(data, function(x) sum(is.na(x)))
```

```
##         PERMNO           date            RET           RET1           RET2
##              0              0              0              0              0
##     market_cap         price1          asset      liability     book_value
##              0              0              0              0              0
##           cash        revenue        earning            EPS book_to_market
##              0              0              0              0              0
##             PE         volume
##              0              0
```

Creating New Variable

```r
data$vol6 <- roll_mean(data$volume, width = 6)
data$vol12 <- roll_mean(data$volume, width = 12)
data$price6 <- roll_mean(data$price1, width = 6)
data$price12 <- roll_mean(data$price1, width = 12)
```

Our target in this project is to predict stock return by using supervised machine learning. Even though our dataset is time-series and many features in the raw dataset have global trends with respect to time, taking percentage change between consecutive observations for all features will provide us better prediction.

```r
data <- data %>% transmute(PERMNO,
                          date,
                          RET,
                          delta_RET1 = (RET1 - lag(RET1))*100/lag(RET2),
                          delta_RET2 = (RET2 - lag(RET2))*100/lag(RET2),
                          delta_market_cap = (market_cap - lag(market_cap))*100/lag(market_cap),
                          delta_price1 = (price1 - lag(price1))*100/lag(price1),
                          delta_asset = (asset - lag(asset))*100/lag(asset),
                          delta_liability = (liability - lag(liability))*100/lag(liability),
                          delta_book_value = (book_value - lag(book_value))*100/lag(book_value),
                          delta_cash = (cash - lag(cash))*100/lag(cash),
                          delta_revenue = (revenue - lag(revenue))*100/lag(revenue),
                          delta_earning = (earning - lag(earning))*100/lag(earning),
                          delta_EPS = (EPS - lag(EPS))*100/lag(EPS),
                          delta_book_to_market = (book_to_market - lag(book_to_market))*100/lag(book_to_market),
                          delta_PE = (PE - lag(PE))*100/lag(PE),
                          delta_volume = (volume - lag(volume))*100/lag(volume),
                          delta_vol6 = (vol6 - lag(vol6))*100/lag(vol6),
                          delta_vol12 = (vol12 - lag(vol12))*100/lag(vol12),
                          delta_price6 = (price6 - lag(price6))*100/lag(price6),
                          delta_price12 = (price12 - lag(price12))*100/lag(price12)) %>% drop_na()
```

```r
#drop NA and infinite value
data <- data %>% filter(!is.infinite(delta_RET1)) %>%
                 filter(!is.infinite(delta_RET2)) %>%
                 filter(!is.infinite(delta_revenue)) %>%
                 filter(!is.infinite(delta_liability)) %>%
                 filter(!is.infinite(delta_earning)) %>%
                 filter(!is.infinite(delta_cash)) %>%
                 filter(!is.infinite(delta_EPS)) %>%
                 filter(!is.infinite(delta_book_to_market)) %>%
                 filter(!is.infinite(delta_PE)) %>%
                 filter(!is.infinite(delta_volume)) %>% as.data.frame()
```

In order to improve performance of model, we will standardize all independent features as follows

```r
data1 <- data %>% transmute(PERMNO,
                          date,
                          RET = (RET - mean(RET))/sd(RET),
                          delta_RET1 = (delta_RET1 - mean(delta_RET1))/sd(delta_RET1),
                          delta_RET2 = (delta_RET2 - mean(delta_RET2))/sd(delta_RET2),
                          delta_market_cap = (delta_market_cap - mean(delta_market_cap))/sd(delta_market_cap),
                          delta_price1 = (delta_price1 - mean(delta_price1))/sd(delta_price1),
                          delta_asset = (delta_asset - mean(delta_asset))/sd(delta_asset),
                          delta_liability = (delta_liability - mean(delta_liability))/sd(delta_liability),
                          delta_book_value = (delta_book_value - mean(delta_book_value))/sd(delta_book_value),
                          delta_cash = (delta_cash - mean(delta_cash))/sd(delta_cash),
                          delta_revenue = (delta_revenue - mean(delta_revenue))/sd(delta_revenue),
                          delta_earning = (delta_earning - mean(delta_earning))/sd(delta_earning),
                          delta_EPS = (delta_EPS - mean(delta_EPS))/sd(delta_EPS),
                          delta_book_to_market = (delta_book_to_market - mean(delta_book_to_market))/sd(delta_book
_to_market),
                          delta_PE = (delta_PE - mean(delta_PE))/sd(delta_PE),
                          delta_volume = (delta_volume - mean(delta_volume))/sd(delta_volume),
                          delta_vol6 = (delta_vol6 - mean(delta_vol6))/sd(delta_vol6),
                          delta_vol12 = (delta_vol12 - mean(delta_vol12))/sd(delta_vol12),
                          delta_price6 = (delta_price6 - mean(delta_price6))/sd(delta_price6),
                          delta_price12 = (delta_price12 - mean(delta_price12))/sd(delta_price12))
```

```r
summary(data1)
```

```
##      PERMNO           date               RET              delta_RET1
##  Min.   :10001   Min.   :20110531   Min.   :-6.64752   Min.   :-270.86733
##  1st Qu.:17005   1st Qu.:20130830   1st Qu.:-0.43236   1st Qu.:  -0.01443
##  Median :78066   Median :20150930   Median :-0.02037   Median :  -0.00054
##  Mean   :60336   Mean   :20152705   Mean   : 0.00000   Mean   :   0.00000
##  3rd Qu.:88612   3rd Qu.:20171031   3rd Qu.: 0.37364   3rd Qu.:   0.01326
##  Max.   :93436   Max.   :20191231   Max.   :55.30825   Max.   : 274.40823
##    delta_RET2         delta_market_cap    delta_price1       delta_asset
##  Min.   :-266.82561   Min.   : -0.0213   Min.   : -0.1970   Min.   : -0.0132
##  1st Qu.:  -0.01830   1st Qu.: -0.0093   1st Qu.: -0.0337   1st Qu.: -0.0062
##  Median :   0.00038   Median : -0.0093   Median : -0.0226   Median : -0.0062
##  Mean   :   0.00000   Mean   :  0.0000   Mean   :  0.0000   Mean   :  0.0000
##  3rd Qu.:   0.01886   3rd Qu.: -0.0093   3rd Qu.: -0.0119   3rd Qu.: -0.0062
##  Max.   : 200.32482   Max.   :474.6317   Max.   :389.5853   Max.   :523.0438
##  delta_liability    delta_book_value      delta_cash       delta_revenue
##  Min.   : -0.0043   Min.   :-114.1811   Min.   : -0.0034   Min.   : -0.4336
##  1st Qu.: -0.0032   1st Qu.:  -0.0047   1st Qu.: -0.0029   1st Qu.: -0.0064
##  Median : -0.0032   Median :  -0.0047   Median : -0.0029   Median : -0.0064
##  Mean   :  0.0000   Mean   :   0.0000   Mean   :  0.0000   Mean   :  0.0000
##  3rd Qu.: -0.0032   3rd Qu.:  -0.0047   3rd Qu.: -0.0029   3rd Qu.: -0.0064
##  Max.   :604.0974   Max.   : 508.4028   Max.   :609.4066   Max.   :336.3500
##  delta_earning         delta_EPS          delta_book_to_market
##  Min.   :-555.2106   Min.   :-103.8293   Min.   :-379.2291
##  1st Qu.:   0.0003   1st Qu.:  -0.0011   1st Qu.:  -0.0058
##  Median :   0.0003   Median :  -0.0011   Median :  -0.0058
##  Mean   :   0.0000   Mean   :   0.0000   Mean   :   0.0000
##  3rd Qu.:   0.0003   3rd Qu.:  -0.0011   3rd Qu.:  -0.0058
##  Max.   : 146.8899   Max.   : 599.7101   Max.   : 376.2343
##     delta_PE          delta_volume         delta_vol6        delta_vol12
##  Min.   :-458.5939   Min.   : -0.02623   Min.   : -0.1208   Min.   : -0.2435
##  1st Qu.:   0.0001   1st Qu.: -0.01712   1st Qu.: -0.0183   1st Qu.: -0.0188
##  Median :   0.0001   Median : -0.01430   Median : -0.0120   Median : -0.0109
##  Mean   :   0.0000   Mean   :  0.00000   Mean   :  0.0000   Mean   :  0.0000
##  3rd Qu.:   0.0001   3rd Qu.: -0.01027   3rd Qu.: -0.0053   3rd Qu.: -0.0025
##  Max.   : 415.4730   Max.   :275.13980   Max.   :422.8273   Max.   :461.5432
##   delta_price6       delta_price12
##  Min.   : -1.76621   Min.   : -4.7201
##  1st Qu.: -0.07539   1st Qu.: -0.1323
##  Median : -0.02063   Median : -0.0127
##  Mean   :  0.00000   Mean   :  0.0000
##  3rd Qu.:  0.02743   3rd Qu.:  0.0864
##  Max.   :263.52565   Max.   :363.8581
```

From the time series perspective, we split data into training set (31/01/2011 - 01/01/2027) and testing set (01/01/2017 - 31/12/2020)

```r
train <- data1 %>% filter(date < "2017-01-01")
test <- data1 %>% filter(date >= "2017-01-01")
```

LOGISTICS REGRESSION

```r
#Build model
formula <- RET ~ delta_RET1 + delta_RET2 + delta_market_cap + delta_price1 +
    delta_asset + delta_liability + delta_book_value +
    delta_cash + delta_revenue + delta_earning + delta_EPS +
    delta_book_to_market + delta_PE + delta_volume + delta_vol6 +
    delta_vol12 + delta_price6 + delta_price12

fitted_logistic <- glm(formula,
  data = train, family = "gaussian")
```

```r
#Use model to make prediction

prediction <- fitted_logistic  %>%
  predict(test) %>%
  as.data.frame() %>%
  mutate(truth = test$RET)
```

```r
#Measure the prediction performance

RMSE(prediction$truth, prediction$.)
```

```
## [1] 1.062924
```

```r
MAE(prediction$truth, prediction$.)
```

```
## [1] 0.6282375
```

```r
cor(prediction$truth, prediction$.)^2 #R-squared
```

```
## [1] 8.090615e-07
```

For logistics regression, RMSE: 1.061, MAE: 0.629, R2: 8.09e-07

RANDOM FOREST

```r
#tuning parameters

recipe <- train %>%
    recipe(RET ~ delta_RET1 + delta_RET2 + delta_market_cap + delta_price1 +
    delta_asset + delta_liability + delta_book_value +
    delta_cash + delta_revenue + delta_earning + delta_EPS +
    delta_book_to_market + delta_PE + delta_volume + delta_vol6 +
    delta_vol12 + delta_price6 + delta_price12)

data_folds <- vfold_cv(train, v = 10)

forest_mod <-
  rand_forest(
    trees = 250,
    mtry = tune(),
    min_n = tune()) %>%
  set_mode("regression") %>%
  set_engine("ranger")

forest_workflow <-
  workflow(recipe, forest_mod)

params <- parameters(min_n(range = c(0,20)),
                     mtry(range = c(0,20)))

forest_grid <- grid_max_entropy(params,
                         size = 10)

#tuning <- forest_workflow %>%
#  tune_grid(
#   resamples = data_folds,
#    grid = forest_grid,
#    metrics = metric_set(mape,mae),
#    control = control_grid(save_pred = TRUE)
#  )

#params_best <- select_best(tuning, "mae")
```

```r
#Random forest model with best params
forest_mod <-
  rand_forest(
    trees = 250,
    mtry = 7,
    min_n = 3) %>%
  set_mode("regression") %>%
  set_engine("ranger")

forest_workflow <-
  workflow(recipe, forest_mod)

final_model <- fit(forest_workflow, train)
```

```r
#Use optimal model to make prediction

prediction <- final_model %>% predict(new_data = test) %>%
  mutate(truth = test$RET,
         company = test$PERMNO)
```

```r
#Measure the prediction performance

RMSE(prediction$truth, prediction$.pred)
MAE(prediction$truth, prediction$.pred)
cor(prediction$truth, prediction$.pred)^2
```

For Random forest, RMSE: 0.767, MAE: 0.328, R2: 0.45

Based on calculated metrics above, it is easy to conclude that random forest is best model to predict return because its RMSE and MAE are lower than those of logistics regression while its R2 is higher.