Portfolio Practice

Team 8

4/12/2020

Creating A Portfolio From Machine Learning Algorithms

Preparing The Data

```
set.seed(123456)
library(tidyverse)
library(scorer)
library(glmnet)
library(gbm)
library(rpart)
library(randomForest)
library(janitor)
library(scales)


data <- read_csv("portfolio_data.csv")
data <- data[-c(1)]
data <- data %>% select(X1_1, everything())
# Deleting any prices from around the Corona Virus
data <- data[-c(200:505)]
data <- data[-c(2:130)]


## Split data for portfolio and for selecting model dropping ticker symbol and the tag
smp_size <- floor(0.5 * nrow(data))
train_ind <- sample(seq_len(nrow(data)), size = smp_size)
model_data <- data[train_ind, ]
model_data <- model_data[-c(1)]
port_data <- data[-train_ind, ]


## Splt into Train Test
smp_size <- floor(0.75 * nrow(model_data))
train_ind <- sample(seq_len(nrow(model_data)), size = smp_size)
train <- model_data[train_ind, ]
test <- model_data[-train_ind, ]
test_real <- as.vector(test[[c(1)]])
```

Finding The Best Algorithm

```
results <- tibble(x = 0, model = 0, mse = 0)
```

```r
for (x in 2:ncol(train)) {
  y <- train[c(1:x)]
  y_test <- test[c(2:x)]

  ## Linear
  fit <- lm(y$`10/2/2019` ~ ., data = y)
  fit_predict <- predict(fit, y_test)
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "linear", mse = mse)

  ## Decision Tree
  tree <- rpart(y$`10/2/2019` ~ ., y)
  fit_predict <- predict(tree, y_test)
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "Decision Tree", mse = mse)

  ## Random Forest
  y <- clean_names(y)
  y_test <- clean_names(y_test)
  rf <- randomForest(x10_2_2019 ~ ., y, ntree = 250, do.trace = F)
  fit_predict <- predict(rf, y_test)
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "RForest", mse = mse)

  # Boosting
  boost <- gbm(x10_2_2019 ~ . ,
               data = y,
               n.trees = 250,
               shrinkage = .001,
               distribution = "gaussian")
  fit_predict <- predict.gbm(boost, y_test, 250)
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "Boost", mse = mse)
}

for (x in 3:ncol(train)){
  y_matrix <- as.matrix(train[c(2:x)])
  y_test <- as.matrix(test[c(2:x)])
  y_train_real <- as.matrix(train[c(1)])

  lasso <- cv.glmnet(y_matrix, y_train_real, alpha = 1, nfolds = 10)
  fit_predict <- predict(lasso, y_test)
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "Lasso", mse = mse)
}

for (x in 3:ncol(train)){
  y_matrix <- as.matrix(train[c(2:x)])
  y_test <- as.matrix(test[c(2:x)])
  y_train_real <- as.matrix(train[c(1)])

  ridge <- cv.glmnet(y_matrix, y_train_real, alpha = 0, nfolds = 10)
  fit_predict <- predict(ridge, y_test)
```

```
  mse <- mean_squared_error(test_real, fit_predict)
  results <- results %>% add_row(x = x, model = "Ridge", mse = mse)
}

results <- results[order(results$mse, decreasing = F),]
results[2,]
```

```
## # A tibble: 1 x 3
##       x model    mse
##   <dbl> <chr>  <dbl>
## 1    54 linear  1.88
```

The results above show x, the best window size, model, the type of model used, and the MSE that was the best.

Final Model & Creating Predictions

```
## Creating the Final Model
#### The Best performing model from above was simple linear regression with a window of 54
final_model_data <- model_data[1:55]
colnames(final_model_data) <- c(1:55)
final_model <- lm(final_model_data$`1` ~ ., data = final_model_data)

## Preparing The Portfolio Data
port_data <- port_data[1:56]
tickers <- port_data[c(1)]
port_data <- port_data[-c(1)]
colnames(port_data) <- c(1:55)
port_data <- cbind(tickers, port_data)

## Predicting For Portfolio Data
port_predictions <- predict(final_model, port_data)
port_data <- cbind(port_predictions, port_data)
port_data <- port_data %>%  mutate(spread = (port_predictions - port_data$`2`))
port_data <- port_data[order(port_data$spread, decreasing = T),]
port_data <- port_data %>% mutate(actual_spread = (port_data$`1` - port_data$`2`))
```

The Winning Basket

For this exercise we are just picking the top predictions that would gain us the most money. Meaning The difference between the predicition and the day before that would give use the most money in the aggregate. We are not looking at the price of the stock, but rather an entity that would gain the most money in the aggregate.

```
### Winning Basket
winning_basket <- port_data[1:5,]
x <- winning_basket[,1:3]
colnames(x) <- c("Predictions", "Tickers", "Actual Price")
x <- x %>% select(Tickers, Predictions, `Actual Price`)
x
```

```
##   Tickers Predictions Actual Price
## 1    ULTA    268.6909       262.79
## 2    NFLX    272.0996       268.03
## 3     MLM    270.0856       261.02
## 4    NVDA    175.1000       173.04
## 5     MHK    121.6421       118.98
```

Creating Random Baskets

These baskets are randomly created to compete against our predicted winning Basket.

```r
### Random Baskets For Comparision
rb1 <- port_data %>% sample_n(5)
set.seed(2)
rb2 <- port_data %>% sample_n(5)
set.seed(3)
rb3 <- port_data %>% sample_n(5)
set.seed(4)
rb4 <- port_data %>% sample_n(5)
set.seed(5)
rb5 <- port_data %>% sample_n(5)
set.seed(6)
rb6 <- port_data %>% sample_n(5)
set.seed(7)
rb7 <- port_data %>% sample_n(5)
set.seed(8)
rb8 <- port_data %>% sample_n(5)
set.seed(9)
rb9 <- port_data %>% sample_n(5)
set.seed(10)
rb10 <- port_data %>% sample_n(5)
```

Creating A Dataframe & Data

```r
### Gains/Losses
Gains_Losses <- c(sum(winning_basket$actual_spread),
                  sum(rb1$actual_spread),
                  sum(rb2$actual_spread),
                  sum(rb3$actual_spread),
                  sum(rb4$actual_spread),
                  sum(rb5$actual_spread),
                  sum(rb6$actual_spread),
                  sum(rb7$actual_spread),
                  sum(rb8$actual_spread),
                  sum(rb9$actual_spread),
                  sum(rb10$actual_spread))

### Average Gains/Losses
Average <- c(sum(winning_basket$actual_spread)/5,
                 sum(rb1$actual_spread)/5,
                 sum(rb2$actual_spread)/5,
```

```r
                   sum(rb3$actual_spread)/5,
                   sum(rb4$actual_spread)/5,
                   sum(rb5$actual_spread)/5,
                   sum(rb6$actual_spread)/5,
                   sum(rb7$actual_spread)/5,
                   sum(rb8$actual_spread)/5,
                   sum(rb9$actual_spread)/5,
                   sum(rb10$actual_spread)/5)

### Portfolio Names
Name <- c('Winner',
                'Basket1',
                'Basket2',
                'Basket3',
                'Basket4',
                'Basket5',
                'Basket6',
                'Basket7',
                'Basket8',
                'Basket9',
                'Basket10')

Portfolio <- data.frame(Name, Gains_Losses, Average)
Portfolio <- Portfolio %>%  mutate(Beat_Winner = if_else(Gains_Losses > -15, "Yes", "No"))
Portfolio <- Portfolio %>%
  mutate(Beat_Aggregate = if_else(Average > mean(port_data$actual_spread), "Yes", "No"))

Portfolio
```

```
##        Name Gains_Losses   Average Beat_Winner Beat_Aggregate
## 1    Winner   -15.019989 -3.003998          No             No
## 2   Basket1   -14.140007 -2.828001         Yes             No
## 3   Basket2   -20.120018 -4.024004          No             No
## 4   Basket3   -22.579998 -4.516000          No             No
## 5   Basket4   -10.320019 -2.064004         Yes            Yes
## 6   Basket5   -15.399990 -3.079998          No             No
## 7   Basket6    -9.020012 -1.804002         Yes            Yes
## 8   Basket7   -11.589998 -2.318000         Yes             No
## 9   Basket8    -9.460001 -1.892000         Yes            Yes
## 10  Basket9    -8.920006 -1.784001         Yes            Yes
## 11 Basket10   -14.119999 -2.824000         Yes             No
```

An Analysis

Winning Basket

```r
Portfolio[1,]
```

```
##     Name Gains_Losses   Average Beat_Winner Beat_Aggregate
## 1 Winner    -15.01999 -3.003998          No             No
```

The above output shows the winning basket's proporties. The real Gains_Losses shows the loss incurred for the portfolio. The average shows the average loss of the portfolio for all the stocks. The Beat Winner variables is useless as it is the winner. The Beat Aggregate shows that it did not beat the Aggregate, the aggregate is just the average of all of the prices difference between the last day and the current day. It was then compared against the average of the portfolios returns.

Random Baskets

Removing the Winning Basket and converting data back into numerical data for some statistics.

```
### Doing Some Analysis
Analysis <- Portfolio[2:11,]
Analysis <- Analysis %>%  mutate(Beat_Winner = if_else(Beat_Winner == "Yes", 1, 0))
Analysis <- Analysis %>%  mutate(Beat_Aggregate = if_else(Beat_Aggregate == "Yes", 1, 0))
```

Amount That Beat The Winning Portfolio

70 percent of the randomly put together portfolios are beating our winning portfolio.

```
percent(mean(Analysis$Beat_Winner))
```

```
## [1] "70.0%"
```

Amount That Beat The Aggregate

40 percent of the randomly put together portfolios are beating the aggregate.

```
percent(mean(Analysis$Beat_Aggregate))
```

```
## [1] "40.0%"
```