Lab 5 – MATH 243

Theodore Dounias

October 17, 2017

1

```r
#a
d <- d %>%
  mutate(MAPE = Price/Earnings_10MA_back)

summary(d[7])
```

```
##       MAPE
##  Min.   : 4.785
##  1st Qu.:11.708
##  Median :15.947
##  Mean   :16.554
##  3rd Qu.:19.959
##  Max.   :44.196
##  NA's   :120
```

```r
summary(d[4])
```

```
##  Earnings_10MA_back
##  Min.   : 8.51
##  1st Qu.:13.89
##  Median :17.48
##  Mean   :25.70
##  3rd Qu.:37.19
##  Max.   :77.00
##  NA's   :120
```

```r
d <- na.omit(d)

attach(d)
```

```
## The following objects are masked from d (pos = 4):
##
##     Date, Earnings, Earnings_10MA_back, Price, Return_10_fwd,
##     Return_cumul
```

Here Earnings_10MA_back had 120 NA values which caused the same number of NA's in MAPE.

```r
#b
lm_MAPE <- lm(Return_10_fwd ~ MAPE, data = d)

coef(summary(lm_MAPE))[2, -3]
```

```
##       Estimate     Std. Error       Pr(>|t|)
##  -4.588536e-03   1.727170e-04  1.641337e-127
```

MAPE is significant in this model.

```r
#c
z<- (length(Price) +1)/5
x <- rep(1, length(Price))
```

```r
x[z:(2*z)] <- 2
x[(2*z):(3*z)] <- 3
x[(3*z):(4*z)] <- 4
x[(4*z):(5*z - 1)] <- 5

d <- d %>%
  mutate(fold = x)
mse <- rep(0, 5)

for(i in 1:5){
  d_train <- d %>%
    filter(fold != i)

  d_test <- d %>%
    filter(fold == i)

  lm_cv5 <- lm(Return_10_fwd ~ MAPE, data = d_train)

  mse[i] <- mean((d_test$Return_10_fwd - predict(lm_cv5, newdata = d_test))^2)
}

cv_mse <- .2 * sum(mse)
cv_mse
```

```
## [1] 0.002508343
```

2

```r
#a
d <- d %>%
  mutate(inv_MAPE = 1/MAPE)

lm_invMAPE <- lm(Return_10_fwd ~ inv_MAPE, data = d)
summary(lm_invMAPE)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ inv_MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106298 -0.030839  0.002955  0.028179  0.103866
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.007659   0.002878  -2.661  0.00788 **
## inv_MAPE     0.995904   0.036513  27.275  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04284 on 1482 degrees of freedom
## Multiple R-squared:  0.3342, Adjusted R-squared:  0.3338
## F-statistic: 743.9 on 1 and 1482 DF,  p-value: < 2.2e-16
```

```r
coef(summary(lm_invMAPE))[2, -3]
```

```
##      Estimate    Std. Error      Pr(>|t|)
##  9.959036e-01  3.651296e-02 4.408311e-133
```

Again, yes.

```r
#b
for(i in 1:5){
  d_train <- d %>%
    filter(fold != i)

  d_test <- d %>%
    filter(fold == i)

  lm_cv5 <- lm(Return_10_fwd ~ inv_MAPE, data = d_train)

  mse[i] <- mean((d_test$Return_10_fwd - predict(lm_cv5, newdata = d_test))^2)
}

cv_mse <- .2 * sum(mse)
cv_mse
```

```
## [1] 0.002260043
```

The MSE here is somewhat smaller than the MSE using the non-inverted model.

3

```r
#a
mse_3a <- mean((d$Return_10_fwd - d$inv_MAPE)^2)
mse_3a
```

```
## [1] 0.001896346
```

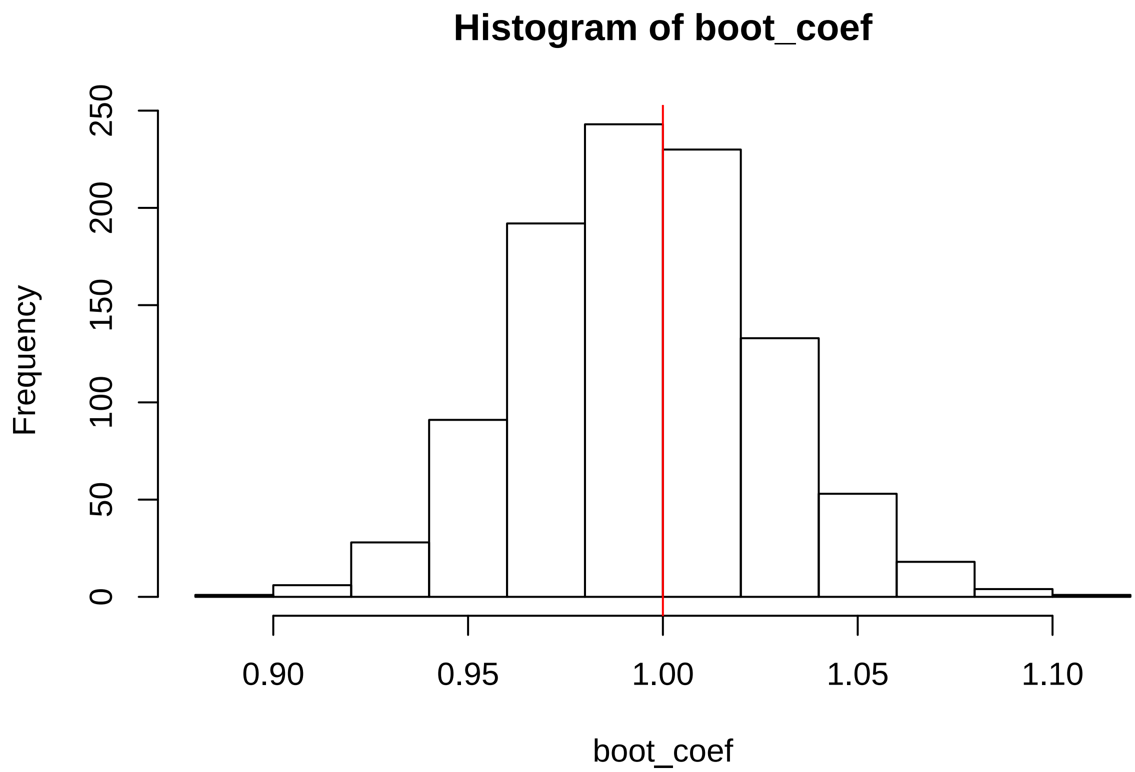
We are essentially not training the model, but just assuming its form. Therefore the training MSE is an estimate of the test MSE in the same way that we would have when cross validating.

4

```r
#a
boot.fn <- function(data, index){
  return(coef(lm(Return_10_fwd ~ inv_MAPE, data = data, subset = index)))
}

boot_coef <- rep(0, 1000)
for(i in 1:1000){
  boot_coef[i] <- boot.fn(d, sample(1484, 1484, replace = TRUE))[2]
}

hist(boot_coef)
abline(v=1,col="red")
```




```
confint(lm_invMAPE)[2, ]
```

```
##    2.5 %   97.5 %
## 0.924281 1.067526
```

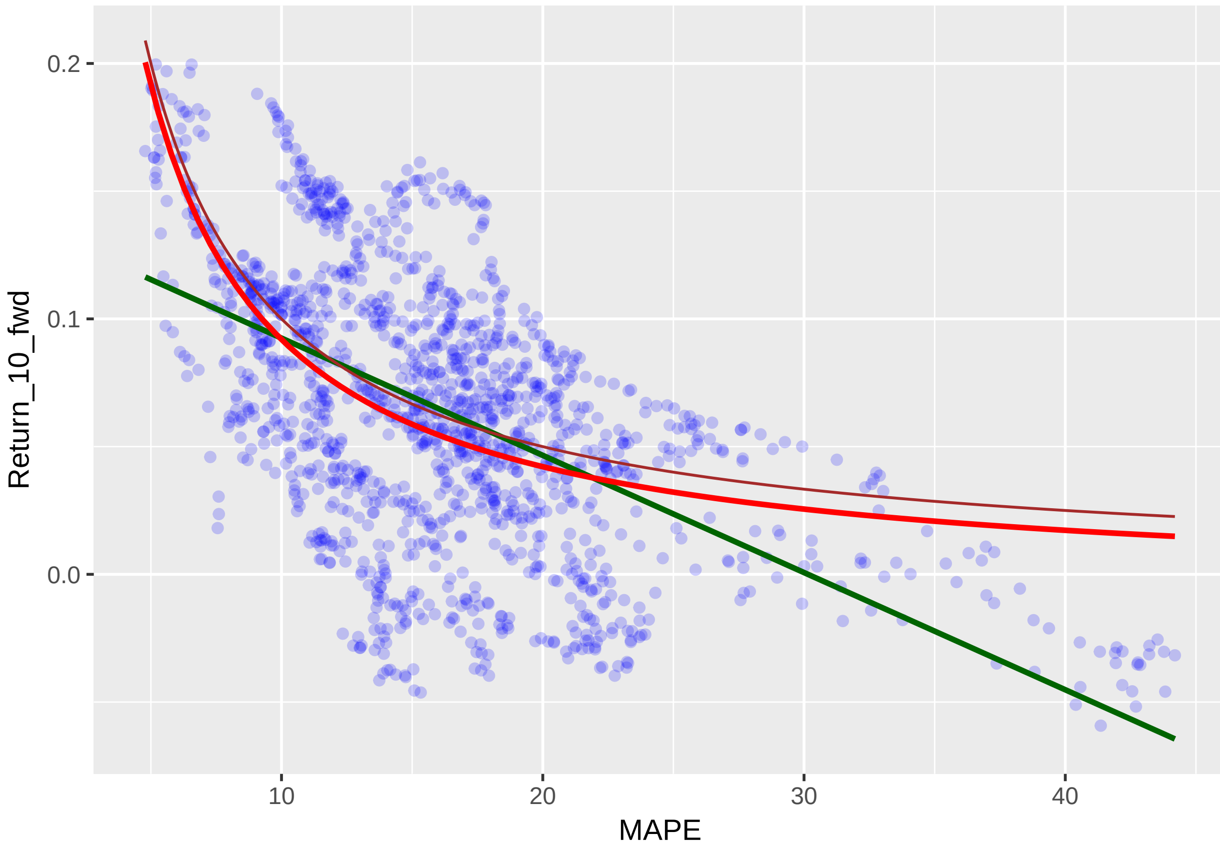
```
error <- qt(0.975,df=length(boot_coef) - 1)*sd(boot_coef)/sqrt(length(boot_coef))
confint_pred <- c(mean(boot_coef) - error, mean(boot_coef) + error)
confint_pred
```

```
## [1] 0.9930561 0.9969264
```

The confidence interval for the booststraped slope is significantly smaller because it reflects the confidence interval of the mean of the boostraped distribution, while confint is the interval for the beta_1 itself.

5

```
d %>% ggplot(aes(y = Return_10_fwd, x = MAPE)) +
  geom_point(aes(y = Return_10_fwd, x = MAPE), alpha = .2, color = "blue") +
  geom_smooth(method = lm, se = FALSE, color = "darkgreen", alpha = 3) +
  geom_smooth(method= lm ,formula= y ~ I(1/x), se = FALSE, color = "red") +
  geom_line(aes(y = inv_MAPE, x = MAPE), color = "brown")
```



6

a. Based on CV MSE we would choose the simplistic model. Looking at the plot, it looks like a competent model, or at least one that is no in any way worse than the other too. It does, however, seem to fail to clearly account for the high variance in the middle parts, and overshoots some observations near its tail. This prediction is strong in that it seems to be flexible and relativelly computationally non-intensive. However, it does seem like it might be susceptible to variance.

b. No. Given our data the simple-minded model's slope coefficient does not lie in our bootstrapped confidence interval.