HW3 - Past Predictions, Future Results

Emulie Chhor

30/12/2022

Question 1 -

Q1a)

```
stocks$MAPE <- stocks$Price / stocks$Earnings_10MA_back
summary(stocks$MAPE)
```

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

```
nrow(stocks[is.na(stocks$Earnings_10MA_back) == TRUE, ])
```

```
## [1] 120
```

There are 120 NA's because there are 120 rows with NA value for Earning_10MA_back

Q1b)

```
model1 <- lm(Return_10_fwd~MAPE, data = stocks)
summary(model1)$coefficients
```

```
##                 Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)  0.138347527 0.002988943  46.28643 4.285102e-290
## MAPE        -0.004588536 0.000172717 -26.56679 1.641337e-127
```

The coefficient estimate is -0.004588536 with standard error of 0.000172717.

Q1c)

```
cv.lm <- function(data, formulae, nfolds = 5) {
  data <- na.omit(data)
  formulae <- sapply(formulae, as.formula)
  n <- nrow(data)
  fold.labels <- sample(rep(1:nfolds, length.out = n))
  mses <- matrix(NA, nrow = nfolds, ncol = length(formulae))
  colnames <- as.character(formulae)
  for (fold in 1:nfolds) {
    test.rows <- which(fold.labels == fold)
    train <- data[-test.rows, ]
    test <- data[test.rows, ]
    for (form in 1:length(formulae)) {
      current.model <- lm(formula = formulae[[form]], data = train)
      predictions <- predict(current.model, newdata = test)
      test.responses <- eval(formulae[[form]][[2]], envir = test)
      test.errors <- test.responses - predictions
      mses[fold, form] <- mean(test.errors^2)
```

```
    }
  }
  return(colMeans(mses))
}

cv.lm(stocks, c("Return_10_fwd ~ MAPE"), nfolds = 5)
```

```
## [1] 0.001871038
```

The MSE for this model under 5-fold is 0.001866959

Question 2 -

Q2a)

$Y = X + \epsilon$ $X = 1/MAPE$

Q2b)

The in-sample data correspond to the data we use to fit our model $MSE = \frac{1}{n}\sum(pred - true)^2$

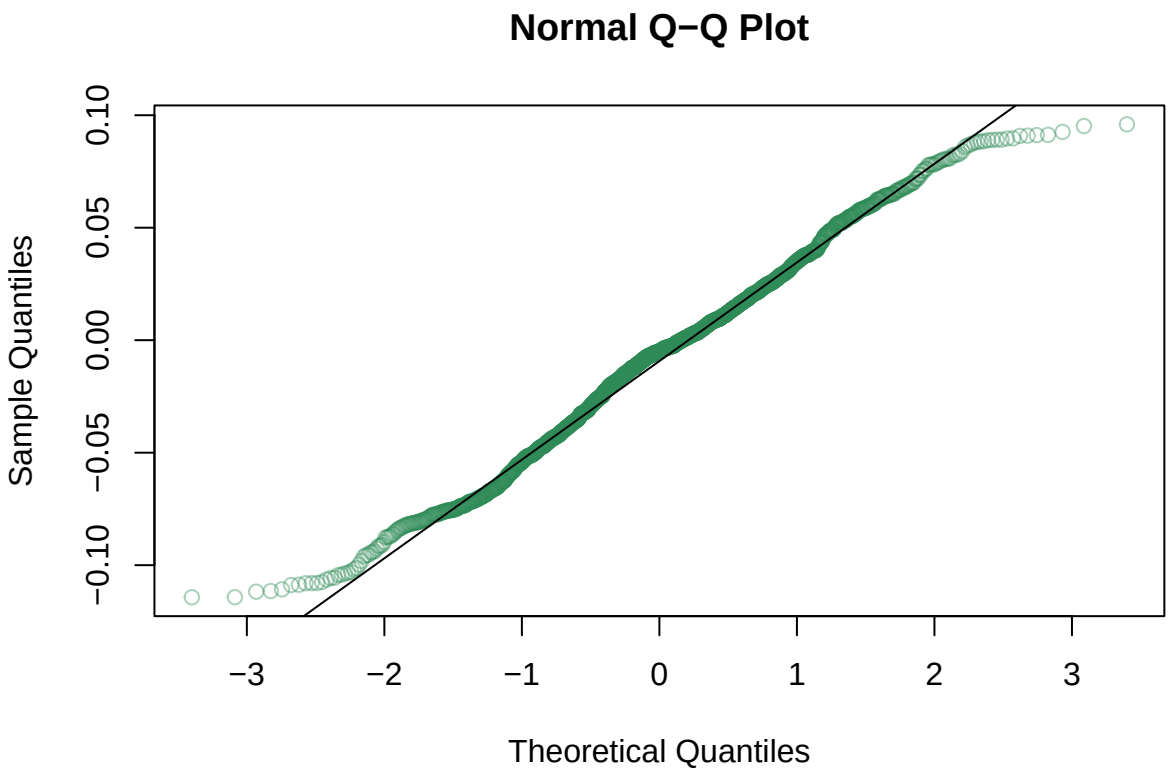
```
model1.insample.mse <- with(stocks, mean((Return_10_fwd - 1/MAPE)^2, na.rm = TRUE))
```

Q2c) CHECK

In this model, since the model is one of the variable of the model, the slope and the intercept are fixed. This means that the model stay the same, regardless if the data are in or out of sample.

Q2d)

```
resids <- with(na.omit(stocks), Return_10_fwd - 1 / MAPE)
qqnorm(resids, col=alpha('seagreen', 0.4))
qqline(resids)
```



Q2e)

By drawing the Q-Q plot, we see that the residuals seems Gaussian. However, at the end of the tails, it seems that they are not.

Correction: The residuals looks Gaussian, but it seems that they have thinner tails than what would have been expected if the distribution were Gaussian

Question 3 -

Q3a)

```
model2 <- lm(Return_10_fwd~ I(1/MAPE), data = stocks)
summary(model2)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ I(1/MAPE), data = stocks)
##
## 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 **
## I(1/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
##   (240 observations deleted due to missingness)
## Multiple R-squared:  0.3342, Adjusted R-squared:  0.3338
## F-statistic: 743.9 on 1 and 1482 DF,  p-value: < 2.2e-16
```

The slope for the coefficient is $\beta_1 = 0.9959$ with standard error of 0.036513.

Q3b)

```
# Compute the 5 fold CV MSE
cv.lm <- function(data, formulae, nfolds = 5) {
  data <- na.omit(data)
  formulae <- sapply(formulae, as.formula)
  n <- nrow(data)
  fold.labels <- sample(rep(1:nfolds, length.out = n))
  mses <- matrix(NA, nrow = nfolds, ncol = length(formulae))
  colnames <- as.character(formulae)
  for (fold in 1:nfolds) {
    test.rows <- which(fold.labels == fold)
    train <- data[-test.rows, ]
    test <- data[test.rows, ]
    for (form in 1:length(formulae)) {
      current.model <- lm(formula = formulae[[form]], data = train)
      predictions <- predict(current.model, newdata = test)
      test.responses <- eval(formulae[[form]][[2]], envir = test)
      test.errors <- test.responses - predictions
      mses[fold, form] <- mean(test.errors^2)
    }
  }
  return(colMeans(mses))
}

model2.mse <- cv.lm(stocks, "Return_10_fwd ~ I(1/MAPE)", nfolds = 5)
kable(data.frame(model=c('model 1', 'model 2'), mse=c(model1.insample.mse, model2.mse)))
```

model	mse
model 1	0.0018963
model 2	0.0018421

The mse for the generalized model is slightly better than the one for the first model and basic model

Question 4 -

Q4a)

```
summary(model1)$coefficients[1,4]
```

```
## [1] 4.285102e-290
```

Since the p-value for the slope of the first model is under $\alpha = 0.05$, the coefficient MAPE is statistically significant

Q4b)

```
summary(model2)$coefficients[1,4]
```

```
## [1] 0.007875108
```

Since the p-value for the slope of the first model is under $\alpha = 0.05$, the coefficient 1/MAPE is statistically significant

Q4c)

```
model3 <- lm(Return_10_fwd ~ MAPE + I(1/MAPE), data = stocks)
summary(model3)$coefficients
```

```
##                Estimate   Std. Error   t value     Pr(>|t|)
## (Intercept)  0.05803291 0.0094470491  6.142967 1.038755e-09
## MAPE        -0.00226037 0.0003101464 -7.288074 5.094613e-13
## I(1/MAPE)    0.59104079 0.0661352073  8.936855 1.162279e-18
```

The coefficient for MAPE is -0.00226 where as the coefficient for 1/MAPE is 0.591 . Both of these variables are statistically significant since their p-value in under $\alpha = .05$

Q4d)

```
model4 <- lm(Return_10_fwd ~ MAPE + I(1/MAPE) + I(MAPE^2), data=stocks)
summary(model4)$coefficients
```

```
##                  Estimate   Std. Error    t value     Pr(>|t|)
## (Intercept)  2.550277e-02 2.612463e-02  0.9761964 3.291267e-01
## MAPE        -2.193897e-04 1.559368e-03 -0.1406914 8.881329e-01
## I(1/MAPE)    7.355515e-01 1.268071e-01  5.8005558 8.068757e-09
## I(MAPE^2)   -3.577618e-05 2.678824e-05 -1.3355183 1.819121e-01
```

The coefficient for MAPE and the square of MAPE are not statistically significant because their p-value are over $\alpha = 0.05$. The coefficient 1/MAPE is statistically significant though.

Q4e)

Whenever we introduce more variable in the model, there are correlation between these variables and thus, the coefficient are less statistically significant. Therefore, it is difficult to decide which variables matter using a hypothesis test on the coefficient.

Question 5 -

Q5a)

We can perform a hypothesis test, where $H_0 : \beta_0 = 0; H_A : \beta_0 \neq 0$: $H_0 : \beta_1 = 1; H_A : \beta_1 \neq 1$:

$P(|t| > t_{df-2}(0.05))$

```
summary(model2)$coefficients
```

```
##                Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept) -0.00765869 0.002878128 -2.660996  7.875108e-03
## I(1/MAPE)    0.99590361 0.036512964 27.275343 4.408311e-133
```

```
beta0_val <- summary(model2)$coefficients[1,1]
beta0_std <- summary(model2)$coefficients[1,2]
beta1_val <- summary(model2)$coefficients[2,1]
beta1_std <- summary(model2)$coefficients[2,2]
n <- nrow(stocks)
```

```r
t0 <- abs(beta0_val - 0) / beta0_std
pt(t0, n-2, lower.tail = FALSE)
```

```
## [1] 0.00393168
```

```r
t1 <- abs(beta1_val -1) / beta1_std
pt(t1, n-2, lower.tail = FALSE)
```

```
## [1] 0.4553429
```

Testing that the original null hypothesis holds is a form of testing whether the basic model is right. If 1/MAPE is significant, then MAPE should be significant too.

Q5b)

In 2d) and 2e), we draw a qqplot of the residuals and found that they were not Gaussian because they were too thin around the tails. R calculate p-value by assuming that the residuals follows a Gaussian, so the computation is not accurate.

Q5c)

```r
# estimate a t distribution from residuals
resids.t.dist <- fitdistr(resids, "t")
```

```
## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced
```

```
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
## Warning in log(s): NaNs produced
```

```
## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced

## Warning in log(s): NaNs produced
```

```
resids.t.dist$estimate
```

```
##             m             s            df
##  -0.007947201   0.042609029 149.169488525
```

```
resids.t.dist$sd
```

```
##            m            s           df
## 1.113452e-03 8.014757e-04 1.001233e+02
```

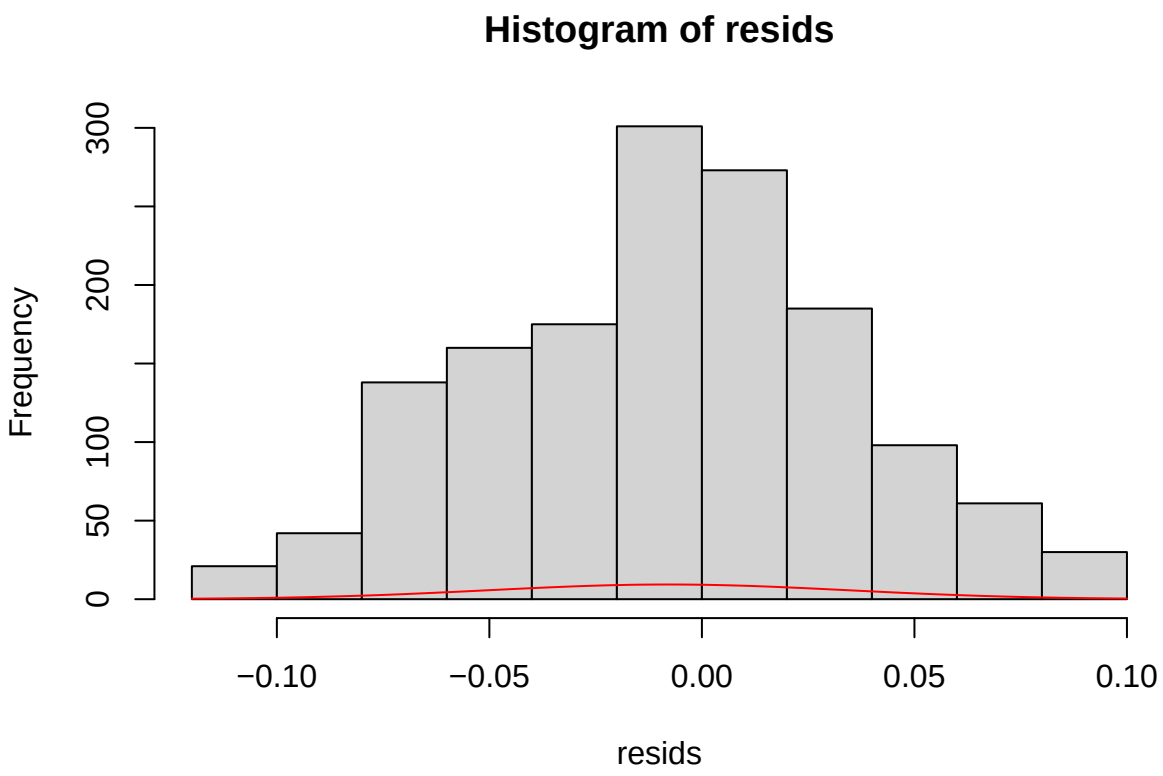
```
results <- data.frame(estimate= resids.t.dist$estimate, sd = resids.t.dist$sd )

# plot a histogram of the residuals and add the estimated t density
hist(resids)

dt.fitted <- function(x,fitted.t) { # estimate density curve from
  m <- fitted.t$estimate["m"]
  s <- fitted.t$estimate["s"]
  df <- fitted.t$estimate["df"]
```

```
  return((1/s)*dt((x-m)/s,df=df))
}

curve(dt.fitted(x, resids.t.dist), add = TRUE, col = 'red')
```



Q5d)

By drawing the qqplot of the residuals in 2d), we saw that the errors can't be normally distributed because they were thinner on the tails. However, we tried to fit the residuals on a t-distribution, we saw that it fitted well.

We choose TODO

Question 6 -

Q6a)

To simulate the model, we need to generate noise following a t-distribution

```
sim.basic.model <- function(MAPE, t.params) {
  n <- length(MAPE)
  m <- t.params['m']
  s <- t.params['s']
  df <- t.params['df']

  noises <- rt(n, df) * s + m
```

```
  data <- data.frame(
    MAPE = MAPE,
    Return_10_fwd = 1/MAPE + noises
  )

  return(data)
}

data2 <- sim.basic.model(stocks$MAPE, resids.t.dist$estimate)
```

Q6b)

```
sim.slope <- function(data) {
  model <- lm(Return_10_fwd ~ I(1/MAPE), data = data)
  return(coef(model)[2])
}
sim.slope(data2)
```

```
## I(1/MAPE)
## 0.9812836
```

```
sim.slope(stocks)
```

```
## I(1/MAPE)
## 0.9959036
```

Q6c)

```
num_simulations <- 2000
sim.results <- replicate(num_simulations, sim.slope(sim.basic.model(stocks$MAPE, resids.t.dist$estimate)
sim.proba <- sum(abs(sim.results - 1) >= abs(coef(model2)[2] - 1)) / num_simulations
```

Q6d)

$H_0 : \tilde{\beta}_1 = 1; H_A : \tilde{\beta}_1 \neq 1$

$t_{score} = \frac{|\beta_1 - 1|}{SE_{\beta_1}} \sim t_{n-2}$

```
signif(sim.proba, 3)
```

```
## [1] 0.909
```

There is insufficient evidence to reject the null hypothesis, so we conclude that the slope is exactly 1.0

Question 7 -

Q7a)

```
# https://en.wikipedia.org/wiki/Kernel_regression
model5 <- npreg(Return_10_fwd~MAPE, data = stocks)
```

```
## Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 /Multistart 1 of 1 |Multis
```

```
kable(model5$bw, col.names = "Bandwidth")
```

Bandwidth
0.5805076

```
kable(model5$bws$fval, col.names = "CV MSE")
```

CV MSE
0.0016927

We have tested 2 others models: MAPE and 1/MAPE with t-distributed errors. The kernel regression has a lower CV MSE than the others models.

Question 8 -

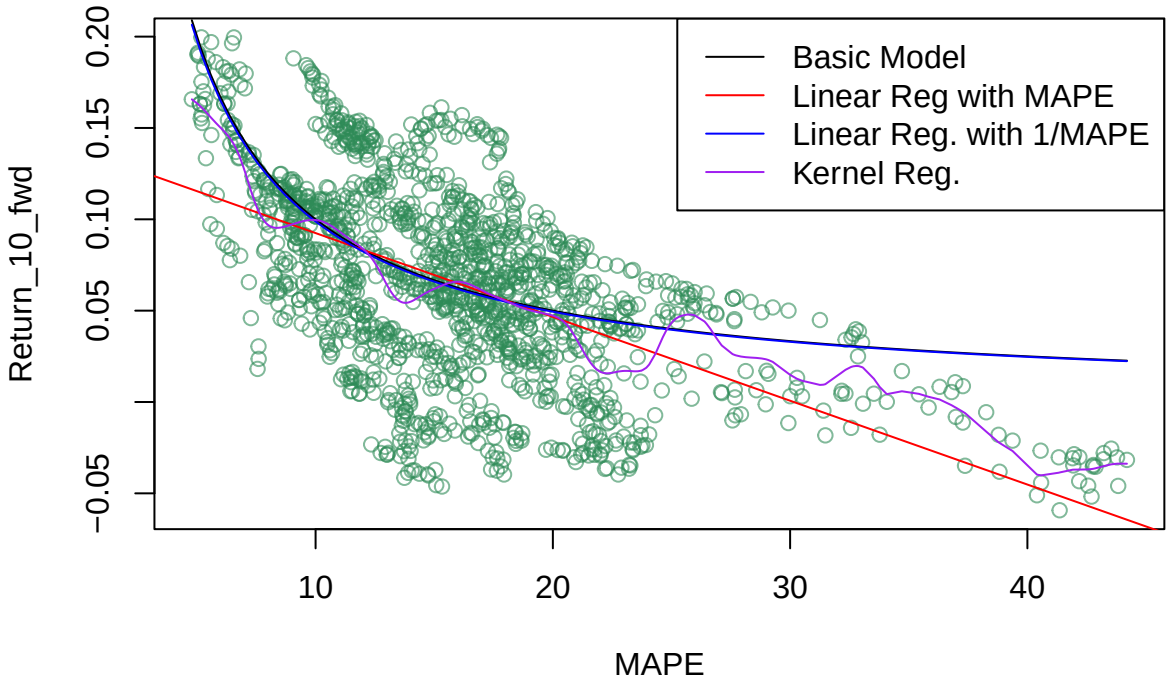
```
# **Q8a)**
with(stocks, plot(MAPE, Return_10_fwd, col=alpha('seagreen', 0.6)))

# **Q8b)**
curve(1/x, add = TRUE, col='black')

# **Q8c)**
abline(model1, col='red')
curve((coef(model2)[1] + coef(model2)[2])/x, add=TRUE, col='blue')

# **Q8d)**
cleaned.stocks <- na.omit(stocks)
MAPE.sorted.idx <- order(cleaned.stocks$MAPE)
lines(cleaned.stocks$MAPE[MAPE.sorted.idx], fitted(model5)[MAPE.sorted.idx],
      col='purple')

# add legend
legend("topright", c("Basic Model", "Linear Reg with MAPE", "Linear Reg. with 1/MAPE",
         "Kernel Reg."), col=c("black", "red", "blue", "purple"), lty = 1)
```



Problem: linear reg. with 1/MAPE should be lower than basic model

Question 9 -

Q9a)

```
kernel.fitted.values <- function(data) {
  npreg.fit <- npreg(Return_10_fwd~MAPE, data = data)
  return(fitted(npreg.fit))
}

# kernel.fitted.values(stocks)
```

Q9b)

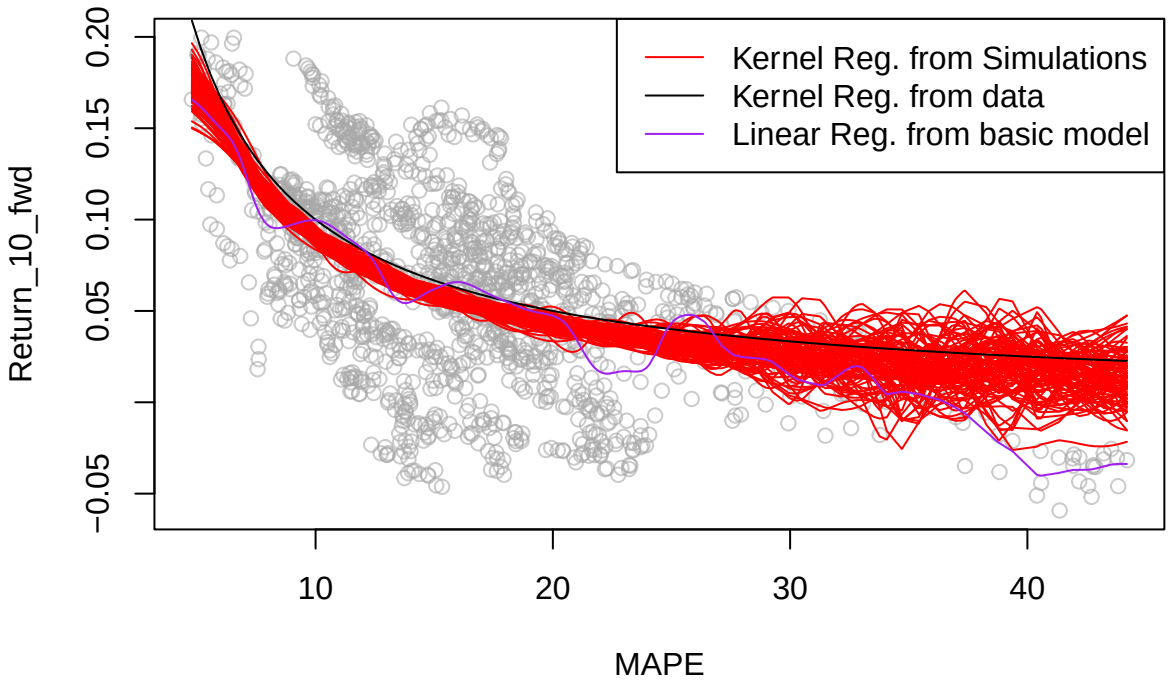
We will simulate the model with $R_t = \frac{1}{MAPE} + \epsilon_t$.

```
kernel.fts <- replicate(100, kernel.fitted.values(
  sim.basic.model(stocks$MAPE, resids.t.dist$estimate)))
```

```
## Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 /Multistart 1 of 1 |Multis
```

```
with(stocks, plot(MAPE, Return_10_fwd, col=alpha('darkgrey', 0.6)))
matplot(cleaned.stocks$MAPE[MAPE.sorted.idx], kernel.fts[MAPE.sorted.idx, ],
        add = TRUE, lty = 1, col = 'red', type = 'l')
curve(1/x, add = TRUE, col='black')
lines(cleaned.stocks$MAPE[MAPE.sorted.idx], fitted(model5)[MAPE.sorted.idx],
      col = 'purple')
```

```
legend("topright", c("Kernel Reg. from Simulations", "Kernel Reg. from data",
                     "Linear Reg. from basic model"),
       col = c('red', 'black', 'purple'), lty = 1)
```



Q9c)

The true-data curve differ from the simulation curves, which indicates that the model might not be accurate as we the model doesn't accurately generate the data.

Question 10 -

Q10a)

Since the CV MSE for the Linear Reg. model 1/MAPE is lower than the Linear Reg. Model for MAPE, it looks like the model $R_t = \beta_0 + \beta_1 \frac{1}{MAPE} + \epsilon_t$ model the data. However, by simulating the data and plotting them using the kernel regression, we see that this model might not be entirely accurate because it doesn't generate the data properly.