Project 1 - Statistical learning

August Jonasson

2024-12-01

Task 1: Linear regression

Loading the stock data.

(a)

Fitting a linear regression model with the stock log-returns as predictors and the log-return of the capital index as response.

```
model <- lm(data = returns, rOMX ~ .)
summary(model)
```

```
##
## Call:
## lm(formula = rOMX ~ ., data = returns)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0082894 -0.0006629  0.0000542  0.0008093  0.0076761
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.400e-04  3.624e-05  -3.864 0.000117 ***
## rABB         3.372e-02  3.836e-03   8.789  < 2e-16 ***
## rNDA.SE.ST   4.936e-02  3.619e-03  13.642  < 2e-16 ***
## HM_B.ST      4.886e-02  1.874e-03  26.070  < 2e-16 ***
## ATCO_A.ST    8.173e-02  8.625e-03   9.475  < 2e-16 ***
## ERIC_B.ST    5.023e-02  2.064e-03  24.329  < 2e-16 ***
## ESSITY_B.ST  3.229e-02  2.899e-03  11.141  < 2e-16 ***
## SAND.ST      4.661e-02  3.533e-03  13.192  < 2e-16 ***
## BOL.ST       1.798e-02  2.055e-03   8.750  < 2e-16 ***
## GETI_B.ST    1.733e-02  1.778e-03   9.745  < 2e-16 ***
## ALFA.ST      2.981e-02  2.752e-03  10.834  < 2e-16 ***
## ATCO_B.ST    3.099e-02  8.724e-03   3.552 0.000395 ***
## VOLV_B.ST    6.517e-02  3.403e-03  19.151  < 2e-16 ***
## SHB_A.ST     3.686e-02  3.735e-03   9.868  < 2e-16 ***
## ELUX_B.ST    1.174e-02  2.131e-03   5.509 4.26e-08 ***
## SEB_A.ST     4.341e-02  4.213e-03  10.304  < 2e-16 ***
## ASSA_B.ST    6.293e-02  3.276e-03  19.208  < 2e-16 ***
## AZN.ST       3.545e-02  2.664e-03  13.305  < 2e-16 ***
## SWED_A.ST    4.685e-02  3.293e-03  14.228  < 2e-16 ***
## TELIA.ST     2.922e-02  3.462e-03   8.440  < 2e-16 ***
## TEL2_B.ST    1.761e-02  2.843e-03   6.195 7.56e-10 ***
## SBB_B.ST     3.117e-03  1.004e-03   3.104 0.001947 **
```

```
## INVE_B.ST    1.074e-01  4.983e-03  21.559  < 2e-16 ***
## SINCH.ST     9.177e-03  9.677e-04   9.484  < 2e-16 ***
## SCA_B.ST     1.678e-02  2.641e-03   6.356 2.76e-10 ***
## HEXA_B.ST    5.642e-02  2.860e-03  19.728  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001399 on 1475 degrees of freedom
## Multiple R-squared:  0.9866, Adjusted R-squared:  0.9864
## F-statistic:  4346 on 25 and 1475 DF,  p-value: < 2.2e-16
```

According to the p-values of the above summary, none of the features are insignificant in their ability to predict the log-returns on the capital market index. This is not surprising at all, since we chose our predictors as the most important stocks on the market. The capital market index is modeled after these stocks. Also, the effects (coefficient estimates) of all features are very similar, i.e. no particular one feature stands out as having more or less of an impact.

By significance, we mean that under the null-hypothesis: that said coefficient has no effect on the response while keeping the others constanst, the observed value would be less than 5 % likely to occur (95 % significance level). For all of our coefficients, this probability is well below 5 % and for most of them, this probability is more or less zero.

(b)

No, the results from part (a) cannot be used to answer the question of which of the stocks have to be included in the mdoel in order to mimic the behavior of the swedish capital market index. We have not adressed potential contaminators such as multicolinearity, overfitting and joint significance between the predictors (we have only checked marginal significance). We have also not validated our model in the sense that we have no idea how it will perform on actual test data.

(c)

Now using the forward selection in order to select the model. This is done by initially only using an intercept and the response variable, and then iteratively adding whichever feature would yield the most significance until no further improvement is seen.

```
forward_model <- step(model, direction = "forward",
                      scope = formula(~ .))
```

```
## Start:  AIC=-19703.28
## rOMX ~ rABB + rNDA.SE.ST + HM_B.ST + ATCO_A.ST + ERIC_B.ST +
##     ESSITY_B.ST + SAND.ST + BOL.ST + GETI_B.ST + ALFA.ST + ATCO_B.ST +
##     VOLV_B.ST + SHB_A.ST + ELUX_B.ST + SEB_A.ST + ASSA_B.ST +
##     AZN.ST + SWED_A.ST + TELIA.ST + TEL2_B.ST + SBB_B.ST + INVE_B.ST +
##     SINCH.ST + SCA_B.ST + HEXA_B.ST
```

```
summary(forward_model)
```

```
##
## Call:
## lm(formula = rOMX ~ rABB + rNDA.SE.ST + HM_B.ST + ATCO_A.ST +
##     ERIC_B.ST + ESSITY_B.ST + SAND.ST + BOL.ST + GETI_B.ST +
##     ALFA.ST + ATCO_B.ST + VOLV_B.ST + SHB_A.ST + ELUX_B.ST +
##     SEB_A.ST + ASSA_B.ST + AZN.ST + SWED_A.ST + TELIA.ST + TEL2_B.ST +
##     SBB_B.ST + INVE_B.ST + SINCH.ST + SCA_B.ST + HEXA_B.ST, data = returns)
##
## Residuals:
```

```
##        Min         1Q     Median         3Q        Max
## -0.0082894 -0.0006629  0.0000542  0.0008093  0.0076761
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.400e-04  3.624e-05  -3.864 0.000117 ***
## rABB         3.372e-02  3.836e-03   8.789  < 2e-16 ***
## rNDA.SE.ST   4.936e-02  3.619e-03  13.642  < 2e-16 ***
## HM_B.ST      4.886e-02  1.874e-03  26.070  < 2e-16 ***
## ATCO_A.ST    8.173e-02  8.625e-03   9.475  < 2e-16 ***
## ERIC_B.ST    5.023e-02  2.064e-03  24.329  < 2e-16 ***
## ESSITY_B.ST  3.229e-02  2.899e-03  11.141  < 2e-16 ***
## SAND.ST      4.661e-02  3.533e-03  13.192  < 2e-16 ***
## BOL.ST       1.798e-02  2.055e-03   8.750  < 2e-16 ***
## GETI_B.ST    1.733e-02  1.778e-03   9.745  < 2e-16 ***
## ALFA.ST      2.981e-02  2.752e-03  10.834  < 2e-16 ***
## ATCO_B.ST    3.099e-02  8.724e-03   3.552 0.000395 ***
## VOLV_B.ST    6.517e-02  3.403e-03  19.151  < 2e-16 ***
## SHB_A.ST     3.686e-02  3.735e-03   9.868  < 2e-16 ***
## ELUX_B.ST    1.174e-02  2.131e-03   5.509 4.26e-08 ***
## SEB_A.ST     4.341e-02  4.213e-03  10.304  < 2e-16 ***
## ASSA_B.ST    6.293e-02  3.276e-03  19.208  < 2e-16 ***
## AZN.ST       3.545e-02  2.664e-03  13.305  < 2e-16 ***
## SWED_A.ST    4.685e-02  3.293e-03  14.228  < 2e-16 ***
## TELIA.ST     2.922e-02  3.462e-03   8.440  < 2e-16 ***
## TEL2_B.ST    1.761e-02  2.843e-03   6.195 7.56e-10 ***
## SBB_B.ST     3.117e-03  1.004e-03   3.104 0.001947 **
## INVE_B.ST    1.074e-01  4.983e-03  21.559  < 2e-16 ***
## SINCH.ST     9.177e-03  9.677e-04   9.484  < 2e-16 ***
## SCA_B.ST     1.678e-02  2.641e-03   6.356 2.76e-10 ***
## HEXA_B.ST    5.642e-02  2.860e-03  19.728  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001399 on 1475 degrees of freedom
## Multiple R-squared:  0.9866, Adjusted R-squared:  0.9864
## F-statistic:  4346 on 25 and 1475 DF,  p-value: < 2.2e-16
```

(d)

```
backward_model <- step(model, direction = "backward")
```

```
## Start:  AIC=-19703.28
## rOMX ~ rABB + rNDA.SE.ST + HM_B.ST + ATCO_A.ST + ERIC_B.ST +
##     ESSITY_B.ST + SAND.ST + BOL.ST + GETI_B.ST + ALFA.ST + ATCO_B.ST +
##     VOLV_B.ST + SHB_A.ST + ELUX_B.ST + SEB_A.ST + ASSA_B.ST +
##     AZN.ST + SWED_A.ST + TELIA.ST + TEL2_B.ST + SBB_B.ST + INVE_B.ST +
##     SINCH.ST + SCA_B.ST + HEXA_B.ST
##
##               Df  Sum of Sq       RSS    AIC
## <none>                      0.0028870 -19703
## - SBB_B.ST     1 0.00001885 0.0029059 -19696
## - ATCO_B.ST    1 0.00002469 0.0029117 -19693
## - ELUX_B.ST    1 0.00005939 0.0029464 -19675
```

```
## - TEL2_B.ST    1 0.00007511 0.0029621 -19667
## - SCA_B.ST     1 0.00007906 0.0029661 -19665
## - TELIA.ST     1 0.00013941 0.0030264 -19635
## - BOL.ST       1 0.00014986 0.0030369 -19629
## - rABB         1 0.00015118 0.0030382 -19629
## - ATCO_A.ST    1 0.00017571 0.0030627 -19617
## - SINCH.ST     1 0.00017605 0.0030631 -19616
## - GETI_B.ST    1 0.00018586 0.0030729 -19612
## - SHB_A.ST     1 0.00019059 0.0030776 -19609
## - SEB_A.ST     1 0.00020782 0.0030948 -19601
## - ALFA.ST      1 0.00022976 0.0031168 -19590
## - ESSITY_B.ST  1 0.00024292 0.0031300 -19584
## - SAND.ST      1 0.00034063 0.0032277 -19538
## - AZN.ST       1 0.00034648 0.0032335 -19535
## - rNDA.SE.ST   1 0.00036425 0.0032513 -19527
## - SWED_A.ST    1 0.00039623 0.0032833 -19512
## - VOLV_B.ST    1 0.00071784 0.0036049 -19372
## - ASSA_B.ST    1 0.00072215 0.0036092 -19370
## - HEXA_B.ST    1 0.00076180 0.0036488 -19354
## - INVE_B.ST    1 0.00090970 0.0037967 -19294
## - ERIC_B.ST    1 0.00115858 0.0040456 -19199
## - HM_B.ST      1 0.00133025 0.0042173 -19137
```

```
summary(backward_model)
```

```
##
## Call:
## lm(formula = rOMX ~ rABB + rNDA.SE.ST + HM_B.ST + ATCO_A.ST +
##     ERIC_B.ST + ESSITY_B.ST + SAND.ST + BOL.ST + GETI_B.ST +
##     ALFA.ST + ATCO_B.ST + VOLV_B.ST + SHB_A.ST + ELUX_B.ST +
##     SEB_A.ST + ASSA_B.ST + AZN.ST + SWED_A.ST + TELIA.ST + TEL2_B.ST +
##     SBB_B.ST + INVE_B.ST + SINCH.ST + SCA_B.ST + HEXA_B.ST, data = returns)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0082894 -0.0006629  0.0000542  0.0008093  0.0076761
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.400e-04  3.624e-05  -3.864 0.000117 ***
## rABB         3.372e-02  3.836e-03   8.789  < 2e-16 ***
## rNDA.SE.ST   4.936e-02  3.619e-03  13.642  < 2e-16 ***
## HM_B.ST      4.886e-02  1.874e-03  26.070  < 2e-16 ***
## ATCO_A.ST    8.173e-02  8.625e-03   9.475  < 2e-16 ***
## ERIC_B.ST    5.023e-02  2.064e-03  24.329  < 2e-16 ***
## ESSITY_B.ST  3.229e-02  2.899e-03  11.141  < 2e-16 ***
## SAND.ST      4.661e-02  3.533e-03  13.192  < 2e-16 ***
## BOL.ST       1.798e-02  2.055e-03   8.750  < 2e-16 ***
## GETI_B.ST    1.733e-02  1.778e-03   9.745  < 2e-16 ***
## ALFA.ST      2.981e-02  2.752e-03  10.834  < 2e-16 ***
## ATCO_B.ST    3.099e-02  8.724e-03   3.552 0.000395 ***
## VOLV_B.ST    6.517e-02  3.403e-03  19.151  < 2e-16 ***
## SHB_A.ST     3.686e-02  3.735e-03   9.868  < 2e-16 ***
## ELUX_B.ST    1.174e-02  2.131e-03   5.509 4.26e-08 ***
## SEB_A.ST     4.341e-02  4.213e-03  10.304  < 2e-16 ***
```

```
## ASSA_B.ST    6.293e-02  3.276e-03  19.208  < 2e-16 ***
## AZN.ST       3.545e-02  2.664e-03  13.305  < 2e-16 ***
## SWED_A.ST    4.685e-02  3.293e-03  14.228  < 2e-16 ***
## TELIA.ST     2.922e-02  3.462e-03   8.440  < 2e-16 ***
## TEL2_B.ST    1.761e-02  2.843e-03   6.195 7.56e-10 ***
## SBB_B.ST     3.117e-03  1.004e-03   3.104 0.001947 **
## INVE_B.ST    1.074e-01  4.983e-03  21.559  < 2e-16 ***
## SINCH.ST     9.177e-03  9.677e-04   9.484  < 2e-16 ***
## SCA_B.ST     1.678e-02  2.641e-03   6.356 2.76e-10 ***
## HEXA_B.ST    5.642e-02  2.860e-03  19.728  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001399 on 1475 degrees of freedom
## Multiple R-squared:  0.9866, Adjusted R-squared:  0.9864
## F-statistic:  4346 on 25 and 1475 DF,  p-value: < 2.2e-16
```

The backward selection yields the same result as the forward selection.

(e)

First, we want to fit the ridge regression model.

```
x_var <- as.matrix(returns[,2:26])
y_var <- as.matrix(returns[,1])
ridge_fit <- glmnet(x_var, y_var, alpha = 0, lambda.min.ratio = 1e-6)
```

Next, in order to choose the best λ we will use leave-one-out cross-validation.

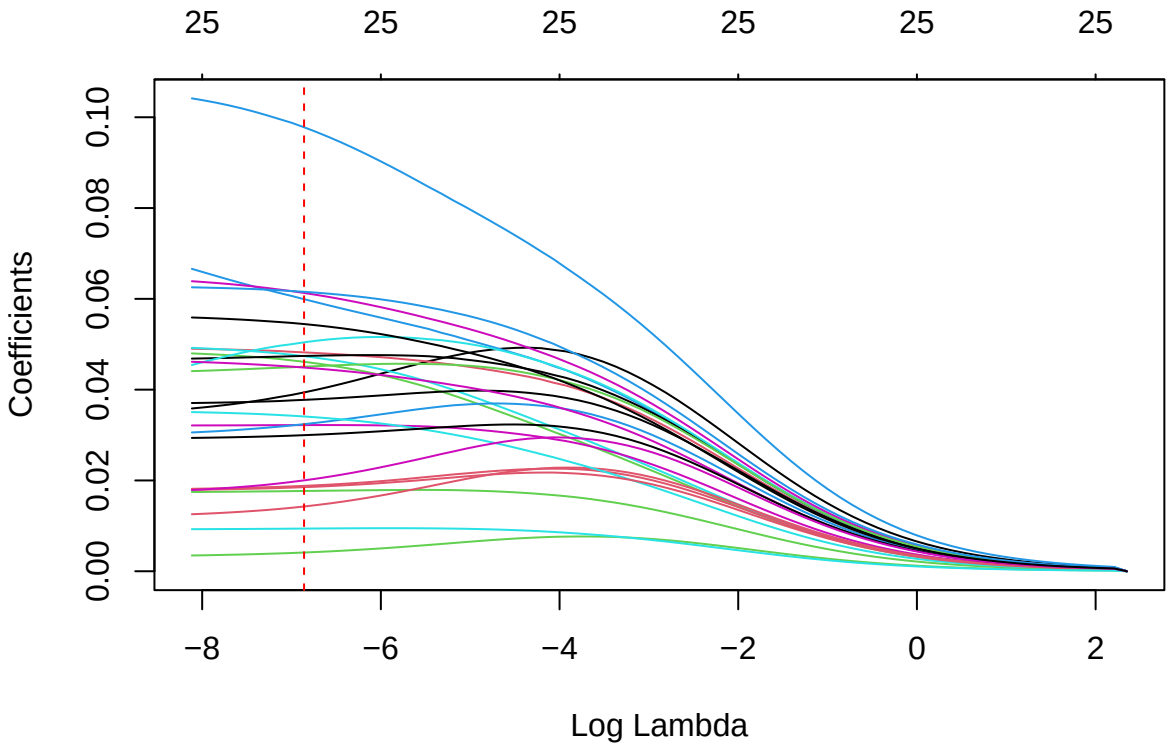
```
# cross-validation
ridge_cv <- cv.glmnet(x_var, y_var, alpha = 0)
```

```
# printing the best lambda for ridge
ridge_best_lambda <- ridge_cv$lambda.min
print(ridge_best_lambda)
```

```
## [1] 0.00104933
```

We can now print the coefficient estimates against the tested values on λ. We also include a red, dashed vertical line that indicates the optimal value on λ found by the cross-validation. As can be seen from the resulting plot, the λ that we found results in next to no shrinkage at all.

```
plot(ridge_fit, xvar = "lambda")
abline(v = log(ridge_best_lambda), lty = "dashed", col = "red")
```



Fitting a new ridge model using this best λ we can compare the coefficient estimates to the simple linear regression model from the first task and see that they are more or less identical.

```
best_ridge_fit <- glmnet(x_var, y_var, alpha = 0, lambda = ridge_best_lambda)
best_ridge_fit$beta
```

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## rABB         0.039066329
## rNDA.SE.ST   0.047695520
## HM_B.ST      0.045931917
## ATCO_A.ST    0.059198227
## ERIC_B.ST    0.047216647
## ESSITY_B.ST 0.032030016
## SAND.ST      0.047294609
## BOL.ST       0.018790452
## GETI_B.ST    0.017646012
## ALFA.ST      0.032533199
## ATCO_B.ST    0.050949063
## VOLV_B.ST    0.061424052
## SHB_A.ST     0.037806818
## ELUX_B.ST    0.014343249
## SEB_A.ST     0.045264003
## ASSA_B.ST    0.061729386
## AZN.ST       0.034144864
## SWED_A.ST    0.045026106
## TELIA.ST     0.030028556
```

```
## TEL2_B.ST   0.018556247
## SBB_B.ST    0.004146011
## INVE_B.ST   0.098201940
## SINCH.ST    0.009399519
## SCA_B.ST    0.020118086
## HEXA_B.ST   0.054480368
```

(f)

Using the Lasso regression and performing the same steps as in the ridge regression task.

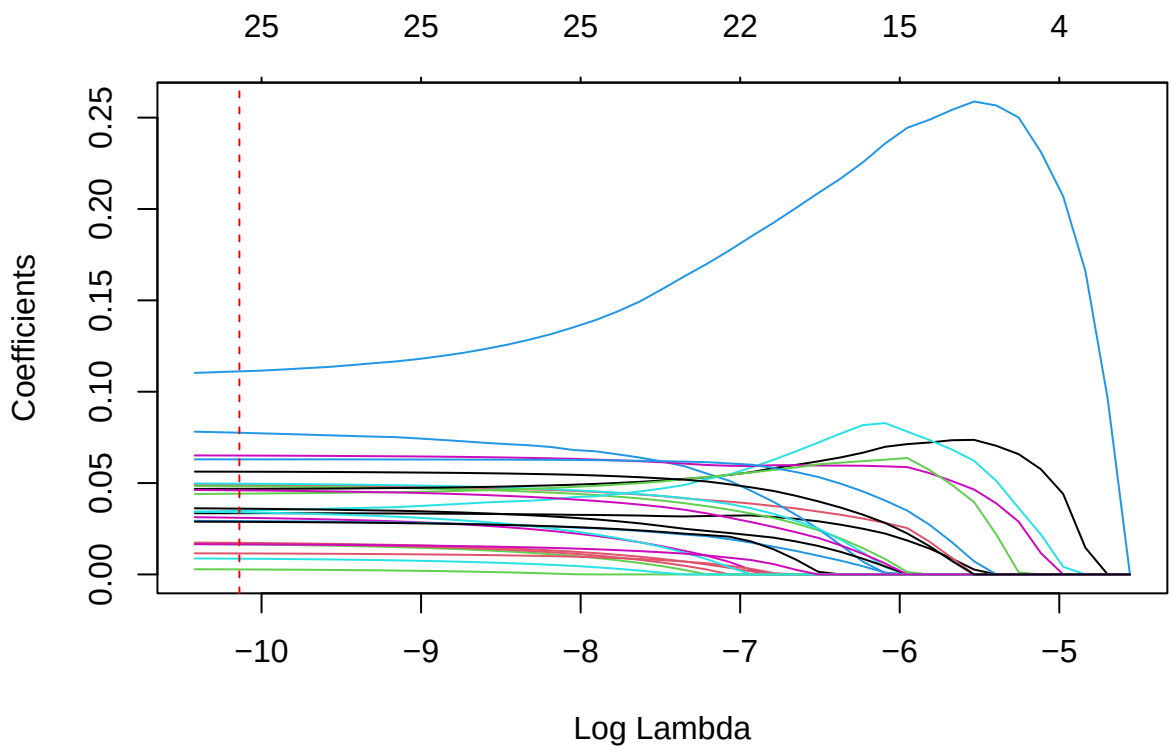
```
# fitting the model
lasso_fit <- glmnet(x_var, y_var, alpha = 1, lambda.min.ratio = 1e-6)

# cross-validation to get best lambda
lasso_cv <- cv.glmnet(x_var, y_var, alpha = 1)

# printing the best lambda for lasso
lasso_best_lambda <- lasso_cv$lambda.min
print(lasso_best_lambda)
```

```
## [1] 3.95066e-05
```

```
plot(lasso_fit, xvar = "lambda")
abline(v = log(lasso_best_lambda), lty = "dashed", col = "red")
```



Again, let us examine the coefficients of the model that uses this best lambda for the lasso regression. Again, we can see that next to no shrinkage at all has been applied.

```r
best_lasso_fit <- glmnet(x_var, y_var, alpha = 1, lambda = lasso_best_lambda)
best_lasso_fit$beta
```

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## rABB         0.033158413
## rNDA.SE.ST   0.048654740
## HM_B.ST      0.048171539
## ATCO_A.ST    0.080077125
## ERIC_B.ST    0.049677262
## ESSITY_B.ST 0.031030049
## SAND.ST      0.047054885
## BOL.ST       0.017319313
## GETI_B.ST    0.016449511
## ALFA.ST      0.029394134
## ATCO_B.ST    0.032348891
## VOLV_B.ST    0.065019136
## SHB_A.ST     0.036408792
## ELUX_B.ST    0.011569436
## SEB_A.ST     0.043738333
## ASSA_B.ST    0.062980998
## AZN.ST       0.034039360
## SWED_A.ST    0.046298243
## TELIA.ST     0.028847744
## TEL2_B.ST    0.016883769
## SBB_B.ST     0.002765315
## INVE_B.ST    0.111117194
## SINCH.ST     0.008621711
## SCA_B.ST     0.016465784
## HEXA_B.ST    0.056062769
```

(g)

According to the results from all of the previous tasks, we can (a bit naively) deduce that all stocks should be included in order to mimic the Swedish capital market index. This is naive in the sense that we have not examined potential co-linearity in any of the steps. Maybe even more importantly, many of the coefficient estimates are very small. Is the added complexity of the model really worth coefficients with such small effects? Probably not.

Task 2: Linear classification

(a)

We start by splitting the data into 80/20 (preserving chronology) and also convert our response variable to binary categorical such that if the return on capital market index on day t is positive we assign 1, and 0 otherwise.

```r
# converting capital market index variable to binary categorical
returns_cat <- returns %>%
  mutate(rOMX = ifelse(rOMX > 0, 1, 0))

# length of data in order make split
n_data <- nrow(returns_cat)

# creating integer representing roughly 80 % split of data.
```

```r
training_length <- round(n_data * 0.8)

# splitting the data into training and test
train_data <- returns_cat[1:training_length, ]
test_data <- returns_cat[(training_length+1):n_data, ]
```

Since the purpose of this task is to train two different models - one using all of the 25 stocks as predictors and the other only using the subset of stocks selected by the lasso regression in the previous task - but our lasso also selected all of the stocks as the best model, we decide to arbitrarily remove half of the coefficients for the second model. We remove the half which has the lowest coefficient estimates, as these should affect the model the least.

```r
# selecting the 12 stocks which coeff estimates were the highest in the lin mod
subset_stocks <- model$coefficients %>%
  data.frame() %>%
  set_colnames("coeff") %>%
  arrange(desc(coeff)) %>%
  t()

subset_stocks <- colnames(subset_stocks)[1:12]
```

```r
# the dimensions of the resulting data sets
dim(train_data)
```

```
## [1] 1201   26
```

```r
dim(test_data)
```

```
## [1] 300  26
```

```r
# all stocks model
logmodel_full <- glm(rOMX ~ ., family = binomial, data = train_data)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
# stock subset model
logmodel_subset <- glm(rOMX ~ ., family = binomial,
                       data = train_data[c("rOMX", subset_stocks)])
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(logmodel_full)
```

```
##
## Call:
## glm(formula = rOMX ~ ., family = binomial, data = train_data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.2520     0.1806  -1.395 0.162959
## rABB         75.7895    25.9476   2.921 0.003491 **
## rNDA.SE.ST   66.1452    16.3720   4.040 5.34e-05 ***
## HM_B.ST      62.5703    11.8673   5.273 1.35e-07 ***
## ATCO_A.ST    71.8045    44.1095   1.628 0.103553
## ERIC_B.ST    76.8049    13.9773   5.495 3.91e-08 ***
## ESSITY_B.ST  25.0559    16.3414   1.533 0.125209
## SAND.ST      41.1652    20.2616   2.032 0.042186 *
## BOL.ST       34.3593    12.2223   2.811 0.004936 **
```

```
## GETI_B.ST    32.8155    11.6140   2.826 0.004721 **
## ALFA.ST      55.0732    16.1454   3.411 0.000647 ***
## ATCO_B.ST    64.6149    41.8572   1.544 0.122662
## VOLV_B.ST    70.1197    19.1957   3.653 0.000259 ***
## SHB_A.ST     60.6139    20.7080   2.927 0.003422 **
## ELUX_B.ST    34.8465    15.3830   2.265 0.023496 *
## SEB_A.ST     78.8466    24.6822   3.194 0.001401 **
## ASSA_B.ST    66.8212    20.2082   3.307 0.000944 ***
## AZN.ST       68.4035    14.2453   4.802 1.57e-06 ***
## SWED_A.ST    57.0166    19.0162   2.998 0.002715 **
## TELIA.ST     43.1222    16.5309   2.609 0.009092 **
## TEL2_B.ST    18.7658    17.2242   1.090 0.275933
## SBB_B.ST     -8.4207     8.7810  -0.959 0.337572
## INVE_B.ST   154.9250    30.8342   5.024 5.05e-07 ***
## SINCH.ST     14.9996     5.3101   2.825 0.004732 **
## SCA_B.ST     32.8206    14.2964   2.296 0.021692 *
## HEXA_B.ST    68.2774    19.0559   3.583 0.000340 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1660.50  on 1200  degrees of freedom
## Residual deviance:  233.32  on 1175  degrees of freedom
## AIC: 285.32
##
## Number of Fisher Scoring iterations: 10
```

```
summary(logmodel_subset)
```

```
##
## Call:
## glm(formula = rOMX ~ ., family = binomial, data = train_data[c("rOMX",
##     subset_stocks)])
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.1079     0.1368  -0.789 0.430305
## INVE_B.ST   148.9714    23.4481   6.353 2.11e-10 ***
## ATCO_A.ST   127.7864    17.7138   7.214 5.44e-13 ***
## VOLV_B.ST    61.2186    13.5019   4.534 5.79e-06 ***
## ASSA_B.ST    70.5987    16.7696   4.210 2.55e-05 ***
## HEXA_B.ST    67.8493    13.9915   4.849 1.24e-06 ***
## ERIC_B.ST    55.8617     9.8106   5.694 1.24e-08 ***
## rNDA.SE.ST   50.9678    14.3683   3.547 0.000389 ***
## HM_B.ST      47.9624     9.0661   5.290 1.22e-07 ***
## SWED_A.ST    24.8381    11.4942   2.161 0.030701 *
## SAND.ST      54.1852    14.0494   3.857 0.000115 ***
## SEB_A.ST     76.4451    21.3346   3.583 0.000339 ***
## SHB_A.ST     38.8051    17.4656   2.222 0.026297 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 1660.5  on 1200  degrees of freedom
## Residual deviance:  362.5  on 1188  degrees of freedom
## AIC: 388.5
##
## Number of Fisher Scoring iterations: 9
```

(b)

```
# predictions of the full model
predictions_full <- predict(logmodel_full,
                        newdata = select(test_data, !rOMX),
                        type = "response")
pred_labels_full <- ifelse(predictions_full > 0.5, 1, 0)

# predictions of the subset model
predictions_subset <- predict(logmodel_subset,
                        newdata = test_data[subset_stocks],
                        type = "response")
pred_labels_subset <- ifelse(predictions_subset > 0.5, 1, 0)
```

And now checking the prediction accuracy.

```
n_test_data <- nrow(test_data)
true_labels <- test_data["rOMX"]

full_model_misclassifications <- abs(true_labels - pred_labels_full) %>% sum()
sub_model_misclassifications <- abs(true_labels - pred_labels_subset) %>% sum()

full_model_accuracy <- full_model_misclassifications / n_test_data
sub_model_accuracy <- sub_model_misclassifications / n_test_data

full_model_accuracy
```

```
## [1] 0.05666667
```

```
sub_model_accuracy
```

```
## [1] 0.07333333
```

(c)