TaigaHasegawaHW2

Taiga Hasegawa(taigah2)

2019/2/2

1

```
#read the file
es_trades=read.csv("ES_Trades.csv")
```

```
#where the symbol is ESU13
esu13=es_trades[es_trades$Symbol=="ESU13",]
```

```
#show the first 6 rows
head(esu13)
```

```
##   Symbol       Date         Time   Price Volume Market.Flag
## 1  ESU13 09/01/2013 17:00:00.083 1640.25      8           E
## 2  ESU13 09/01/2013 17:00:00.083 1640.25      1           E
## 3  ESU13 09/01/2013 17:00:00.083 1640.25      2           E
## 4  ESU13 09/01/2013 17:00:00.083 1640.25      1           E
## 5  ESU13 09/01/2013 17:00:00.083 1640.25      1           E
## 6  ESU13 09/01/2013 17:00:00.083 1640.25     12           E
##   Sales.Condition Exclude.Record.Flag Unfiltered.Price
## 1               0                  NA          1640.25
## 2               0                  NA          1640.25
## 3               0                  NA          1640.25
## 4               0                  NA          1640.25
## 5               0                  NA          1640.25
## 6               0                  NA          1640.25
```

```
#define dollar bars
dollar_bars <- function(dat, nvol)
{
  n=cumsum(esu13$Unfiltered.Price)
  winIdx <- as.factor(floor(n/nvol))
  H <- aggregate(dat$Unfiltered.Price, by = list(winIdx), max)$x
  L <- aggregate(dat$Unfiltered.Price, by = list(winIdx), min)$x
  O <- aggregate(dat$Unfiltered.Price, by = list(winIdx), function(x){x[1]})$x
  C <- aggregate(dat$Unfiltered.Price, by = list(winIdx), function(x){x[length(x)]})$x
  list(H=H,L=L,O=O,C=C)
}
```

```
#implementing the dollar bar
dollar_bar=dollar_bars(esu13,1000000)
length(dollar_bar$H)
```

```
## [1] 5572
```

When the threhold is 1,000,000, we have 5572 dollar bars.

2

```r
#difine cusum filter
istar_CUSUM <- function(yvec, h)
{
  S_pos <- S_neg <- 0
  istar <- NULL
  yminusEy <- diff(yvec)
  n <- length(yminusEy)
  for(i in 1:n)
  {
    S_pos <- max(0, S_pos + yminusEy[i])
    S_neg <- min(0, S_neg + yminusEy[i])
    if(max(S_pos, -S_neg) >= h) # note that Snippet 2.4 in AFML does not follow the definition of S_t
    {
      istar <- c(istar, i)
      S_pos <- S_neg <- 0
    }
  }
  return(istar)
}
```

```r
i_CUSUM <- istar_CUSUM(dollar_bar$C, h=3)
```

```r
i_CUSUM
```

```
##   [1]   13   27   48   67   78   92  111  141  155  161  167  184  195  214
##  [15]  244  317  337  363  381  431  450  479  496  521  565  638  667  687
##  [29]  700  714  727  755  771  791  810  825  843  909  921  960  991 1005
##  [43] 1053 1063 1088 1106 1124 1151 1187 1260 1316 1344 1448 1513 1521 1590
##  [57] 1606 1633 1639 1647 1661 1682 1704 1731 1754 1784 1865 1964 2081 2106
##  [71] 2156 2168 2177 2193 2222 2236 2262 2282 2313 2333 2376 2391 2416 2425
##  [85] 2440 2456 2485 2529 2552 2567 2580 2594 2617 2647 2707 2738 2756 2786
##  [99] 2810 2928 2951 2986 3044 3071 3132 3154 3177 3205 3219 3250 3268 3295
## [113] 3317 3354 3397 3514 3551 3624 3650 3778 3784 3800 3818 3832 3860 3941
## [127] 3984 4098 4172 4194 4255 4392 4493 4496 4511 4535 4561 4616 4665 4725
## [141] 4795 4834 4951 5076 5123 5166 5212 5271 5332 5368 5402 5436 5484 5510
## [155] 5557
```

When h is 3, we have 155 feature bars and it is reasonable.

3

```r
#define the triple barrier method
#return the dataframe
label_meta=function(x,events,ptSl){
  t0 <- events$t0
  t1 <- events$t1
  trgt <- events$trgt
  side <- events$side
  u <- ptSl[1]
  l <- ptSl[2]
  rstlist=data.frame()
  for (i in 1:dim(events)[1]){
```

```r
    i_trgt=trgt[i]
    i_x=x[t0[i]:t1[i]]
    i_side=side[i]
    if(i_side==0){
      up <- i_trgt*u
      lo <- i_trgt*l
      isup <- (i_x/i_x[1]-1) >= up
      islo <- -(i_x/i_x[1]-1) >= lo
      T_up <- ifelse(sum(isup)>0, min(which(isup)), Inf)
      T_lo <- ifelse(sum(islo)>0, min(which(islo)), Inf)
      ret <- i_x[min(T_up, T_lo, length(i_x))] / i_x[1] - 1
      rst <- c(T_up, T_lo, length(i_x), ret)
    }else if(i_side==1){
      up <- i_trgt*u
      isup <- (i_x/i_x[1]-1) >= up
      T_up <- ifelse(sum(isup)>0, min(which(isup)), Inf)
      T_lo <- Inf
      ret <- i_x[min(T_up, T_lo, length(i_x))] / i_x[1] - 1
      rst <- c(T_up, T_lo, length(i_x), ret)
    }else{
      lo <- i_trgt*l
      islo <- -(i_x/i_x[1]-1) >= lo
      T_up <- Inf
      T_lo <- ifelse(sum(islo)>0, min(which(islo)), Inf)
      ret <- i_x[min(T_up, T_lo, length(i_x))] / i_x[1] - 1
      rst <- c(T_up, T_lo, length(i_x), ret)
    }
    rstlist=rbind(rstlist,rst)
  }
  colnames(rstlist)=c("T_up","T_lo","length","ret")
  return(rstlist)
}
```

```r
#where ptSl=[1,1] and t1=70
n_event=length(i_CUSUM)
events <- data.frame(t0=i_CUSUM+1, t1 = i_CUSUM+70, trgt = rep(0.002, n_event), side=rep(0,n_event))
x=dollar_bar$C
ptSl=c(1,1)
triplebarrier=label_meta(x,events,ptSl)
triplebarrier
```

```
##     T_up T_lo length           ret
## 1     36  Inf     70  0.0021302495
## 2     40  Inf     70  0.0021289538
## 3    Inf  Inf     70  0.0001518372
## 4    Inf  Inf     70 -0.0010615711
## 5    Inf  Inf     70 -0.0015192950
## 6    Inf   51     70 -0.0022751403
## 7    Inf   32     70 -0.0024264483
## 8    Inf  Inf     70 -0.0006077180
## 9      7  Inf     70  0.0024356828
## 10   Inf    7     70 -0.0021260440
## 11    18  Inf     70  0.0021305737
## 12   Inf   12     70 -0.0022779043
```

```
## 13    20  Inf      70  0.0021308980
## 14    46  Inf      70  0.0021263670
## 15   Inf  Inf      70 -0.0009097801
## 16   Inf  Inf      70 -0.0007593014
## 17    34  Inf      70  0.0021279830
## 18   Inf   28      70 -0.0021241086
## 19   Inf  Inf      70 -0.0004557885
## 20   Inf   48      70 -0.0021247534
## 21   Inf   46      70 -0.0021279830
## 22   Inf   22      70 -0.0021296015
## 23   Inf   70      70 -0.0024379095
## 24   Inf  Inf      70 -0.0001526019
## 25   Inf  Inf      70  0.0004582251
## 26   Inf   38      70 -0.0021396913
## 27   Inf  Inf      70  0.0000000000
## 28   Inf   56      70 -0.0021403455
## 29   Inf  Inf      70  0.0006126512
## 30   Inf   29      70 -0.0021403455
## 31   Inf  Inf      70  0.0003063256
## 32   Inf   27      70 -0.0021410002
## 33    70  Inf      70  0.0021446078
## 34    64   29      70 -0.0021416552
## 35    31  Inf      70  0.0022981462
## 36    27  Inf      70  0.0022953328
## 37   Inf  Inf      70  0.0013748854
## 38   Inf   15      70 -0.0021347972
## 39   Inf  Inf      70  0.0019862490
## 40    35  Inf      70  0.0021367521
## 41   Inf  Inf      70 -0.0015246227
## 42   Inf   64      70 -0.0021354484
## 43   Inf   11      70 -0.0022865854
## 44    66  Inf      70  0.0021390374
## 45    65  Inf      70  0.0021361001
## 46    23  Inf      70  0.0021390374
## 47    30  Inf      70  0.0022883295
## 48    43  Inf      70  0.0021318715
## 49   Inf  Inf      70  0.0012159903
## 50   Inf  Inf      70  0.0018206645
## 51    55  Inf      70  0.0021212121
## 52   Inf  Inf      70  0.0004539952
## 53   Inf  Inf      70  0.0013623978
## 54   Inf   13      70 -0.0021160822
## 55   Inf  Inf      70 -0.0009085403
## 56    23  Inf      70  0.0021218551
## 57    44  Inf      70  0.0022692890
## 58   Inf   31      70 -0.0022651767
## 59    11  Inf      70  0.0024209411
## 60   Inf   17      70 -0.0021144842
## 61   Inf  Inf      70  0.0003025261
## 62   Inf  Inf      70  0.0003023432
## 63   Inf   36      70 -0.0022655188
## 64    35  Inf      70  0.0021183235
## 65    33  Inf      70  0.0021154427
## 66   Inf  Inf      70 -0.0004524887
```

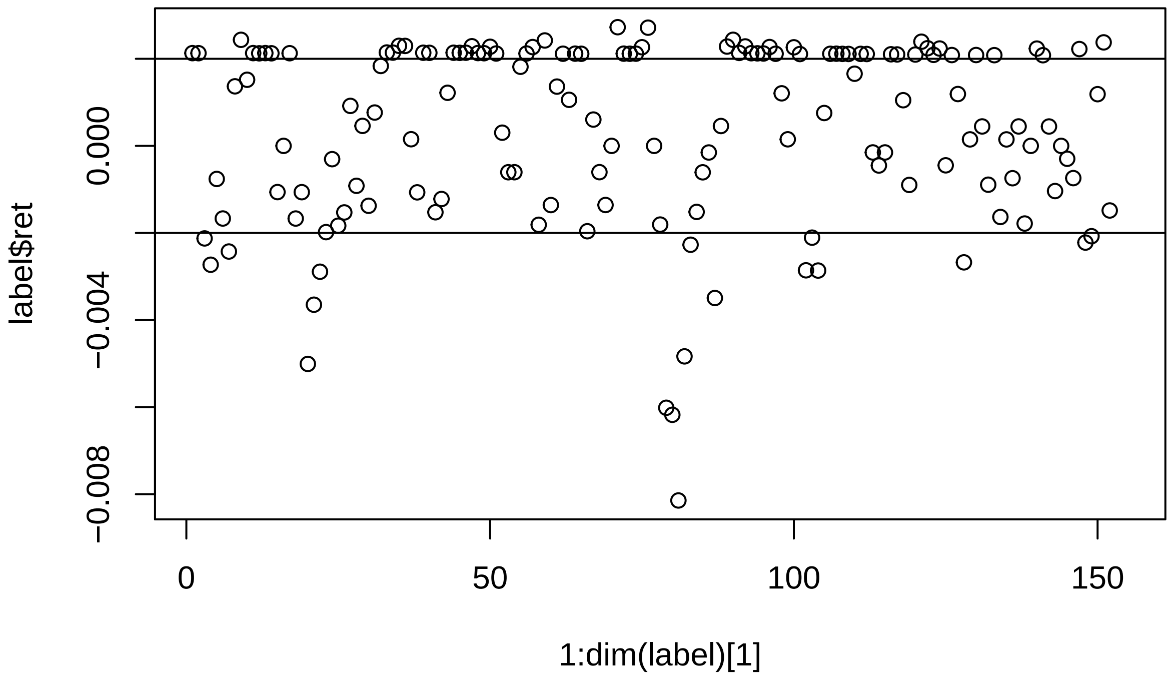
```
## 67   Inf  Inf      70 -0.0001510574
## 68   Inf  Inf      70 -0.0007548309
## 69   Inf   54      70 -0.0022638092
## 70   Inf   53      70 -0.0024191110
## 71    26  Inf      70  0.0027256208
## 72    14  Inf      70  0.0021186441
## 73    60  Inf      70  0.0021160822
## 74    36  Inf      70  0.0021180030
## 75    20  Inf      70  0.0022655188
## 76    30  Inf      70  0.0027149321
## 77   Inf  Inf      70  0.0000000000
## 78   Inf  Inf      70 -0.0006025004
## 79   Inf   30      70 -0.0021052632
## 80   Inf   52      70 -0.0021090690
## 81   Inf   18      70 -0.0021112954
## 82   Inf   27      70 -0.0022662034
## 83   Inf   22      70 -0.0022706630
## 84   Inf   32      70 -0.0022747953
## 85   Inf   22      70 -0.0022768670
## 86   Inf  Inf      70  0.0010639915
## 87   Inf  Inf      70 -0.0012152514
## 88   Inf   41      70 -0.0021276596
## 89   Inf   26      70 -0.0024334601
## 90    60  Inf      70  0.0024364245
## 91    18  Inf      70  0.0021351228
## 92    33  Inf      70  0.0022838002
## 93    48  Inf      70  0.0021279830
## 94   Inf  Inf      70  0.0016699560
## 95    40  Inf      70  0.0021215336
## 96    46  Inf      70  0.0022699758
## 97    41  Inf      70  0.0021154427
## 98   Inf  Inf      70  0.0010561255
## 99   Inf  Inf      70 -0.0001506478
## 100   34  Inf      70  0.0022634676
## 101   41  Inf      70  0.0021093868
## 102  Inf  Inf      70 -0.0007521059
## 103  Inf   42      70 -0.0021074816
## 104  Inf  Inf      70 -0.0004521477
## 105  Inf  Inf      70 -0.0009059339
## 106   68  Inf      70  0.0021148036
## 107   29  Inf      70  0.0021173624
## 108  Inf  Inf      70  0.0009055237
## 109  Inf  Inf      70  0.0001507841
## 110  Inf   20      70 -0.0021090690
## 111   33  Inf      70  0.0021128886
## 112   56  Inf      70  0.0021097046
## 113  Inf  Inf      70  0.0007521059
## 114  Inf  Inf      70 -0.0007515407
## 115  Inf  Inf      70 -0.0003008424
## 116  Inf  Inf      70  0.0010518407
## 117  Inf  Inf      70  0.0003000300
## 118  Inf  Inf      70  0.0011988611
## 119  Inf  Inf      70 -0.0010469638
## 120    7  Inf      70  0.0020976925
```

```
## 121   37  Inf     70  0.0023923445
## 122   21  Inf     70  0.0022424877
## 123   33  Inf     70  0.0020904883
## 124   43  Inf     70  0.0022358027
## 125  Inf  Inf     70  0.0001488982
## 126  Inf  Inf     70  0.0011918951
## 127  Inf  Inf     70  0.0002976190
## 128  Inf  Inf     70 -0.0008916630
## 129  Inf  Inf     70 -0.0001488317
## 130   65  Inf     70  0.0020867491
## 131  Inf  Inf     70  0.0007439369
## 132  Inf  Inf     70  0.0002972210
## 133    4  Inf     70  0.0020833333
## 134  Inf   23     70 -0.0020790021
## 135  Inf  Inf     70 -0.0001487652
## 136  Inf   49     70 -0.0020793109
## 137  Inf  Inf     70  0.0014880952
## 138  Inf   52     70 -0.0020805469
## 139  Inf  Inf     70 -0.0005954153
## 140  Inf  Inf     70  0.0014896470
## 141   47  Inf     70  0.0020820940
## 142  Inf  Inf     70  0.0008906041
## 143  Inf  Inf     70 -0.0005928561
## 144  Inf  Inf     70 -0.0008888889
## 145  Inf  Inf     70  0.0007410701
## 146  Inf   55     70 -0.0020719254
## 147  Inf  Inf     70  0.0008895478
## 148  Inf  Inf     70 -0.0013327410
## 149  Inf   68     70 -0.0020746888
## 150  Inf  Inf     70  0.0001484340
## 151   43  Inf     70  0.0023770614
## 152  Inf  Inf     70 -0.0004451699
## 153  Inf  Inf     70 -0.0002968680
## 154   NA   NA     70            NA
## 155   NA   NA     70            NA
```

4

```r
#where ptSl=[1,0] and t1=100
events <- data.frame(t0=i_CUSUM+1, t1 = i_CUSUM+100, trgt = rep(0.002, n_event), side=rep(1,n_event))
ptSl=c(1,0)
label=label_meta(x,events,ptSl)
```

```r
#plot the rst and threshold
plot(1:dim(label)[1],label$ret)
abline(h=events$trgt[1])
abline(h=-events$trgt[1])
```




```r
#calculatet the features from feature bars
iTmp <- c(0, i_CUSUM)
fMat0 <- t(sapply(1:(length(i_CUSUM)),
                  function(i){
                      winTmp <- x[(iTmp[i]+1):(iTmp[i+1])]
                      C <- winTmp[length(winTmp)]
                      SD <- sd(winTmp)
                      return(c(C,SD))
                    }
  ))
```

```r
#change into the dataframe
fMat0 <- data.frame(fMat0)
names(fMat0) <- c("Close", "SD")
X_train=fMat0
```

```r
#labeling
Y_train <- rep(0, n_event)
Y_train[label$ret>=events$trgt*ptSl[1]] <- 1
```

```r
#linear regression
fit1 <- glm(Y_train ~ X_train$Close + X_train$SD, family = "binomial")
summary(fit1)
```

```
##
## Call:
## glm(formula = Y_train ~ X_train$Close + X_train$SD, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3142  -1.0677  -0.8107   1.2260   1.6758
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)   35.92470   18.29996   1.963   0.0496 *
## X_train$Close -0.02223    0.01093  -2.034   0.0420 *
## X_train$SD     0.74561    1.48766   0.501   0.6162
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 210.83  on 154  degrees of freedom
## Residual deviance: 205.75  on 152  degrees of freedom
## AIC: 211.75
##
## Number of Fisher Scoring iterations: 4
```

I used the close price and standard deviation as predictors. The result showed that close price and intercept is significant with p value less than 5%.

Next I used only close price as predictors.

```r
fit2 <- glm(Y_train ~  X_train$Close , family = "binomial")
summary(fit2)
```

```
##
## Call:
## glm(formula = Y_train ~ X_train$Close, family = "binomial")
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.267  -1.061  -0.809   1.228   1.637
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   37.97554   17.85390   2.127   0.0334 *
## X_train$Close -0.02313    0.01079  -2.145   0.0320 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 210.83  on 154  degrees of freedom
## Residual deviance: 206.00  on 153  degrees of freedom
## AIC: 210
##
## Number of Fisher Scoring iterations: 4
```

Close price and intercept was still significant.

I predicted the outcome, using close price as predictors and made the confusion matrix.

```r
pred <- predict(fit2, type="response")
#confusion matrix
table(Y_train, pred > 0.5)
```

```
##
## Y_train FALSE TRUE
##       0    75   15
##       1    53   12
```

I couldn't categorize well when the true label is 1.

ROC shows that the closer the ROC curve is to upper left corner, the higher the overall accuracy of the test is. The below ROC was almost straight line and it was difficult to tell which point was the highest accuracy but 0.7 true positive rate and 0.5 false positive seemed to be good.

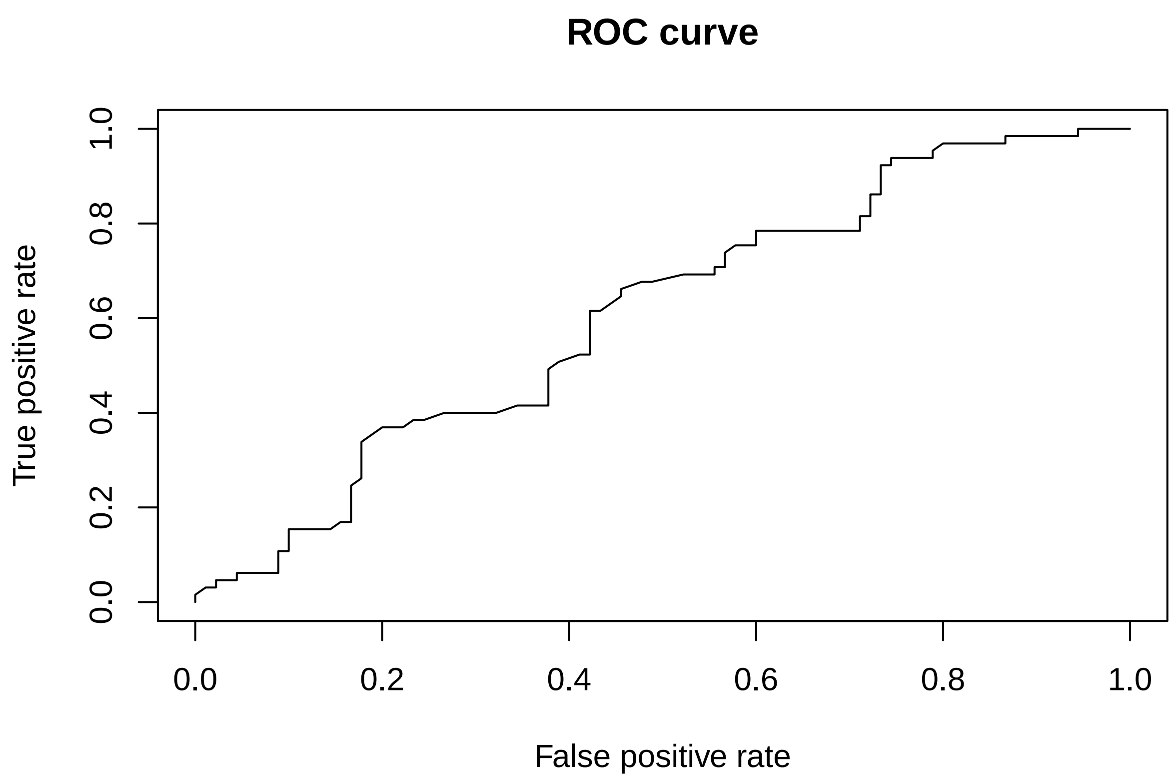
```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
ROC <- prediction(pred, Y_train)
ROC_perf <- performance(ROC, 'tpr','fpr')
plot(ROC_perf, main = "ROC curve")
```



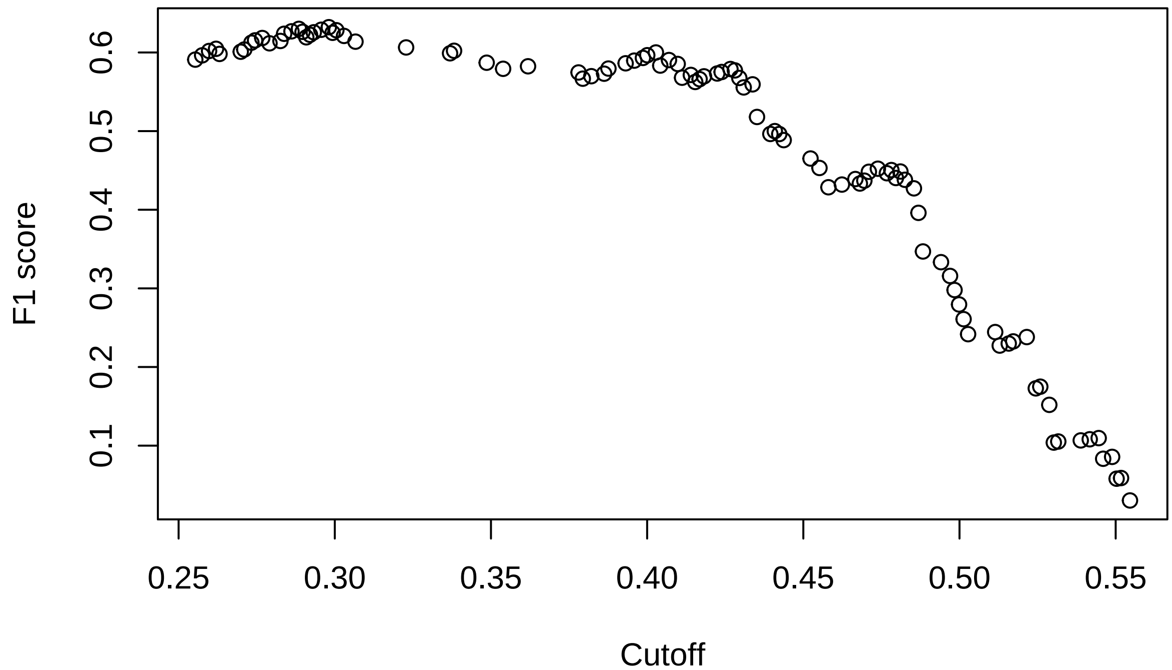
The higher the AUC value is, the more accurate the model is. In this case, the AUC is 0.6073504.

```
AUC <- performance(ROC, "auc")
AUC@y.values[[1]]
```

```
## [1] 0.6073504
```

The below F1 score plot showed F1 score at every cutoff point .

```
F1 <- performance(ROC, 'f')
plot(F1@y.values[[1]][-1]~F1@x.values[[1]][-1], xlab="Cutoff", ylab="F1 score")
```




```r
optCut <- F1@x.values[[1]][-1][which.max(F1@y.values[[1]][-1])]
```

When the cutoff point was 0.2980931, the f1 score was the highest. At this cutoff point, the confusion matrix was like below. we could classify the label 1 very well.

```r
table(Y_train, pred >= optCut)
```

```
##
## Y_train FALSE TRUE
##       0    23   67
##       1     4   61
```