# Quant Strategy

## 经济学院

## 数据清洗，针对下一期的涨跌幅：

## 涨跌幅低于-1%记为-2; 涨跌幅高于-1%，低于 0%，记为 -1;

## 涨跌幅高于 1%记为 2; 涨跌幅低于-1%，高于 0%，记为 1;

## 读文件，lm 回归判断哪些解释变量是有用的，减少噪音。确定自变量和因变量

```
data_sample <- read.zoo("./data/HS300_5.csv",sep=",",header=T,format =
"%Y-%m-%d")
data_sample <- na.omit(data_sample)
fit <- lm(ret~open+close+high+low+volume+mv10+mv20+vol10+vol20+rsi5+rsi
14+macd.macd1+signal.macd1+macd.macd2+signal.macd2+dn+mavg+up+pctB,data
 = data_sample)
summary(fit)
```

```
##
## Call:
## lm(formula = ret ~ open + close + high + low + volume + mv10 +
##     mv20 + vol10 + vol20 + rsi5 + rsi14 + macd.macd1 + signal.macd1
+
##     macd.macd2 + signal.macd2 + dn + mavg + up + pctB, data = data_s
ample)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.092163 -0.007645  0.000413  0.008760  0.090727
##
## Coefficients: (3 not defined because of singularities)
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.606e-03  2.297e-03   2.005 0.045037 *
## open         -6.408e-05  1.751e-05  -3.659 0.000258 ***
## close        -3.926e-05  1.579e-05  -2.487 0.012936 *
## high          8.806e-05  2.023e-05   4.352  1.4e-05 ***
## low           1.283e-05  1.653e-05   0.776 0.437755
## volume       -7.027e-14  4.726e-12  -0.015 0.988138
## mv10          1.195e-05  1.981e-05   0.603 0.546551
## mv20         -1.151e-05  1.473e-05  -0.781 0.434633
## vol10         5.009e-06  1.221e-05   0.410 0.681690
## vol20        -6.301e-06  8.962e-06  -0.703 0.482050
```

```
## rsi5         -2.334e-05  2.446e-05  -0.954 0.340036
## rsi14        -4.370e-05  5.207e-05  -0.839 0.401356
## macd.macd1    1.609e-03  1.266e-03   1.271 0.203903
## signal.macd1 -4.636e-04  5.239e-04  -0.885 0.376194
## macd.macd2   -3.872e-04  8.115e-04  -0.477 0.633314
## signal.macd2 -8.485e-04  1.548e-03  -0.548 0.583625
## dn                   NA         NA      NA       NA
## mavg                 NA         NA      NA       NA
## up                   NA         NA      NA       NA
## pctB          6.473e-03  2.955e-03   2.190 0.028582 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01806 on 2849 degrees of freedom
## Multiple R-squared:  0.01937,    Adjusted R-squared:  0.01386
## F-statistic: 3.517 on 16 and 2849 DF,  p-value: 2.566e-06
```

## 确定训练样本和测试样本

```
x <- data_sample[,-c(4:21)]
y <- data_sample[,20]
insams<- "2005-01-01"
insame<- "2016-12-31"
osams<- "2017-01-01"
osame<- "2017-12-31"
inrow <- which(index(data_sample) >= insams & index(data_sample) <= ins
ame)
outrow <- which(index(data_sample) >= osams & index(data_sample) <= osa
me)
```

## 计算 SVM 在 2 种分类机，4 种核函数下模型的错误次数

```
type <- c("C-classification","nu-classification")
kernel <- c("linear","polynomial","radial","sigmoid")
accuracy <- matrix(0,2,4)
for (i in 1:2)
{
  for ( j in 1:4)
  {
    model <- svm(x[inrow,],y[inrow],type=type[i],kernel = kernel[j])
    pred_temp <- predict(model,x[outrow])
    accuracy[i,j] <- sum(pred_temp!=as.vector(y[outrow]))
  }
}
dimnames(accuracy) <- list(type,kernel)
accuracy

##                    linear polynomial radial sigmoid
## C-classification      113        112    145     213
## nu-classification     117        218    166     145
```

由以上结果可知，使用 SVM 进行实验，type="C-classification",kernel = "polynomial"的模型最优。

```
model1 <- svm(x[inrow,],y[inrow],type="C-classification",kernel = "polynomial")
pred1 <- predict(model1,x[outrow,])
#table(pred1,y[outrow])
outresult_out<- confusionMatrix(pred1,y[outrow])
outresult_out

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  -2  -1   1   2
##         -2   0   0   0   0
##         -1   0   0   0   0
##         1    8  91 114  13
##         2    0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5044
##                  95% CI : (0.4373, 0.5714)
##     No Information Rate : 0.5044
##     P-Value [Acc > NIR] : 0.5266
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: -2 Class: -1 Class: 1 Class: 2
## Sensitivity             0.0000    0.0000   1.0000  0.00000
## Specificity             1.0000    1.0000   0.0000  1.00000
## Pos Pred Value             NaN       NaN   0.5044      NaN
## Neg Pred Value          0.9646    0.5973      NaN  0.94248
## Prevalence              0.0354    0.4027   0.5044  0.05752
## Detection Rate          0.0000    0.0000   0.5044  0.00000
## Detection Prevalence    0.0000    0.0000   1.0000  0.00000
## Balanced Accuracy       0.5000    0.5000   0.5000  0.50000
```

简单回测展示 2017 年收益情况 预测为 1 或 2 开多仓，预测为 -2 开空仓

```
signal <- ifelse( pred1==1 | pred1==2,1,ifelse(pred1==-2 ,-1,0))
simreturn <- data_sample$ret[outrow]
cost <- 0
strategy_return <- Lag(simreturn)*Lag(signal)-cost
cumm_return<- Return.cumulative(strategy_return)
```

```r
annual_return <- Return.annualized(strategy_return)
charts.PerformanceSummary(strategy_return)
```