中国石化

[1] Read data ohlcv data path ../d\_price/600028\_SH\_1min.csv Shape\_raw: (168266, 5)

(163726, 316)

[163726 rows x 294 columns]

get\_HT\_pp df.shape: (163726, 294) len\_right: 72 len\_left: 36

[3.1]target\_Y.shape: (163726, 3)

[3.2] DEBUG All technical indicators Extracted Count: 294

[4.2] Select the best technical patterns to train with features\_W3 Extracted Count: 91 Names : mtum\_MINUS\_DM,mtum\_PVOs\_12\_26\_9,sti\_VAR,olap\_SAREXT,mtum\_td\_seq\_sig,mtum\_PLUS\_DM,sti\_BETA,ti\_vortex\_pos\_14,mtum\_DX,mtum\_PLUS\_DI,ma\_KAMA\_5,mtum\_K\_9\_3,cycl\_SINE\_lead,mtum\_QQEl\_14\_5\_4236,ichi\_senkou\_b,day\_minute,demark\_r1,mtum\_ADX,clas\_s1,mtum\_QQEs\_14\_5\_4236,mtum\_ADXR,ma\_TRIMA\_10,tend\_renko\_TR,ma\_WMA\_10,cycl\_DCPHASE,mtum\_WILLIAMS\_R,mtum\_TRIX,sti\_STDDEV,ma\_WMA\_100,wood\_s2,tend\_VHF\_28,vola\_TRANGE,ma\_T3\_20,ichi\_senkou\_a,mtum\_MACD\_signal,mtum\_PVOh\_12\_26\_9,mtum\_STOCH\_d,ti\_mass\_index\_9\_25,olap\_PWMA\_10,ti\_coppock\_14\_11\_10,tend\_PSARr\_002\_02,tend\_LDECAY\_5,vola\_ATR,volu\_NVI\_1,mtum\_BOP,wood\_s3,olap\_JMA\_7\_0,ti\_donchian\_upper\_20,mtum\_MINUS\_DI,cama\_s2,ti\_supertrend\_20,mtum\_J\_9\_3,cycl\_DCPERIOD,vola\_THERMOl\_20\_2\_05,sti\_CORREL,mtum\_MACD,fibo\_s1,mtum\_FISHERTs\_9\_1,close,cycl\_SINE\_sine,ti\_kelt\_20\_upper,mtum\_MFI,wood\_r1,volu\_PVI\_1,mtum\_AR\_26,cycl\_PHASOR\_quad,mtum\_ER\_10,low,demark\_s1,mtum\_CTI\_12,mtum\_STOCH\_Fa\_kd,volu\_Chaikin\_AD,mtum\_FISHERT\_9\_1,wood\_r2,sti\_LINEARREG\_INTERCEPT,mtum\_STOCH\_kd,clas\_r1,olap\_SAR,mtum\_PVO\_12\_26\_9,open,mtum\_STC\_10\_12\_26\_05,mtum\_AO\_5\_34,volu\_PVOL,ti\_choppiness\_14,mtum\_AROON\_up,mtum\_RSX\_14,vola\_HWL,fibo\_r2,vola\_THERMOma\_20\_2\_05,vola\_NATR,vola\_THERMO\_20\_2\_05

[4.3] features\_W3 index Dates: from 2020-09-01 11:26:00+00:00 to 2023-08-01 14:56:00+00:00 Shape: (163726, 91)

[5] For correct training, for correct training the values must be normalised to between 0 and 1

[5.1] Normalise\_data path: data/scalers/600028\_SH\_\_1min\_\_test\_A1\_\_scalex.pkl

[6] Currently you have for each target\_Y value a row of technical indicators, you add a 'window' to make the decision to predict whether the +-48 rows above will be taken (about 4 hours of previous indicators, splited in 5min).

[6.1] get\_window\_data Shapes Y: (163697, 3) X: (163697, 30, 91)

[7] data split between training and validation

[7.1] Shapes: X\_train: (130957, 30, 91) y\_train: (130957, 3) index\_train: (130957,)

[8] Ground True data are unbalanced Given that there is a lot of 'do nothing' 0, and very little 'do buy' 1 or 'do sell' 2, weight balancing is required, to give more importance to the minorities.

[8.1] Class weight path: data/class\_weight/600028\_SH\_\_1min\_\_test\_A1\_\_class\_weight.pkl Dict: {0: 0.4043080665876309, 1: 5.02154990605468, 2: 3.053464838649506}

[9] Creation of the TF model architecture. must respect the input\_shape and output\_shape and the 'softmax' , from there EXPERIMENT combinations

[9.1] arrays to use the TF model , input\_shape: (30, 91) output\_shape: 3

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 30, 144) 13248

dense\_1 (Dense) (None, 30, 72) 10440

dropout (Dropout) (None, 30, 72) 0

dense\_2 (Dense) (None, 30, 36) 2628

dropout\_1 (Dropout) (None, 30, 36) 0

dense\_3 (Dense) (None, 30, 12) 444

flatten (Flatten) (None, 360) 0

dense\_4 (Dense) (None, 3) 1083

=================================================================

Total params: 27843 (108.76 KB)

Trainable params: 27843 (108.76 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

[9.2] optimizer config : {'name': 'Adam', 'weight\_decay': None, 'clipnorm': None, 'global\_clipnorm': None, 'clipvalue': None, 'use\_ema': False, 'ema\_momentum': 0.99, 'ema\_overwrite\_frequency': None, 'jit\_compile': False, 'is\_legacy\_optimizer': False, 'learning\_rate': 0.001, 'beta\_1': 0.9, 'beta\_2': 0.999, 'epsilon': 1e-07, 'amsgrad': False}

You must install pydot (`pip install pydot`) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) for plot\_model to work.

[9.3] print diagram of the TF model Path: outputs/plots/600028\_SH\_\_1min\_\_test\_A1\_\_buysell.png

[10] Start training TF model outputs/600028\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

Epoch 1/90

5484/5484 [==============================] - 22s 4ms/step - loss: 0.7624 - categorical\_accuracy: 0.7190 - val\_loss: 0.6592 - val\_categorical\_accuracy: 0.7324

Epoch 2/90

5484/5484 [==============================] - 22s 4ms/step - loss: 0.6078 - categorical\_accuracy: 0.7447 - val\_loss: 0.5818 - val\_categorical\_accuracy: 0.7522

Epoch 3/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.5622 - categorical\_accuracy: 0.7543 - val\_loss: 0.6077 - val\_categorical\_accuracy: 0.7493

Epoch 4/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.5410 - categorical\_accuracy: 0.7582 - val\_loss: 0.7378 - val\_categorical\_accuracy: 0.6731

Epoch 5/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.5232 - categorical\_accuracy: 0.7614 - val\_loss: 0.6353 - val\_categorical\_accuracy: 0.7268

Epoch 6/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.5114 - categorical\_accuracy: 0.7643 - val\_loss: 0.6384 - val\_categorical\_accuracy: 0.7370

Epoch 7/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.4999 - categorical\_accuracy: 0.7650 - val\_loss: 0.5659 - val\_categorical\_accuracy: 0.7694

Epoch 8/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.4922 - categorical\_accuracy: 0.7688 - val\_loss: 0.6091 - val\_categorical\_accuracy: 0.7487

Epoch 9/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4798 - categorical\_accuracy: 0.7727 - val\_loss: 0.7357 - val\_categorical\_accuracy: 0.7112

Epoch 10/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4707 - categorical\_accuracy: 0.7755 - val\_loss: 0.5569 - val\_categorical\_accuracy: 0.7737

Epoch 11/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4579 - categorical\_accuracy: 0.7789 - val\_loss: 0.6179 - val\_categorical\_accuracy: 0.7527

Epoch 12/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.4500 - categorical\_accuracy: 0.7796 - val\_loss: 0.6576 - val\_categorical\_accuracy: 0.7252

Epoch 13/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4403 - categorical\_accuracy: 0.7828 - val\_loss: 0.7277 - val\_categorical\_accuracy: 0.6968

Epoch 14/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4352 - categorical\_accuracy: 0.7870 - val\_loss: 0.6130 - val\_categorical\_accuracy: 0.7493

Epoch 15/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4239 - categorical\_accuracy: 0.7894 - val\_loss: 0.6891 - val\_categorical\_accuracy: 0.7228

Epoch 16/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.4123 - categorical\_accuracy: 0.7935 - val\_loss: 0.7005 - val\_categorical\_accuracy: 0.7321

Epoch 17/90

5484/5484 [==============================] - 23s 4ms/step - loss: 0.4110 - categorical\_accuracy: 0.7931 - val\_loss: 0.5700 - val\_categorical\_accuracy: 0.7783

Epoch 18/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.4028 - categorical\_accuracy: 0.7997 - val\_loss: 0.6733 - val\_categorical\_accuracy: 0.7231

Epoch 19/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.3957 - categorical\_accuracy: 0.7998 - val\_loss: 0.6449 - val\_categorical\_accuracy: 0.7382

Epoch 20/90

5484/5484 [==============================] - 24s 4ms/step - loss: 0.3855 - categorical\_accuracy: 0.8034 - val\_loss: 0.5796 - val\_categorical\_accuracy: 0.7733

Epoch 20: early stopping

[10.1] Model initial\_weights saved : data/initial\_weights/600028\_SH\_\_1min\_\_test\_A1\_\_initial\_weights.tf

[11] Do a predict and eval. Load path: outputs/600028\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

[11.1] array to predict x\_test.shape: (32740, 30, 91)

5457/5457 [==============================] - 5s 895us/step

[11.2] read the evaluation: (32740, 30, 91)

(32740,) (32740,)

precision recall f1-score support

0 0.97 0.90 0.93 30202

1 0.31 0.59 0.40 1152

2 0.34 0.65 0.44 1386

accuracy 0.87 32740

macro avg 0.54 0.71 0.59 32740

weighted avg 0.92 0.87 0.89 32740

outputs/model\_info/600028\_SH\_\_1min\_\_test\_A1\_.csv

count per%

y\_pred y\_test

0 0 27056 82.64

2 0 1646 5.03

1 0 1500 4.58

2 2 894 2.73

1 1 677 2.07

0 2 453 1.38

1 352 1.08

2 1 123 0.38

1 2 39 0.12

outputs/model\_info/600028\_SH\_\_1min\_\_test\_A1\_\_.info.txt

包钢股份

(168266, 5)

[2] Calculating Technical indicators. stock: 600010\_SH

[2.1] Calculated Technical indicators. stock: 600010\_SH Tech indicator count: (25103, 316)

[3] Calculate the target\_Y (ground true) ,what is the target to detect? the highest and lowest peaks stock: 600010\_SH

get\_HT\_pp df.shape: (25103, 294) len\_right: 120 len\_left: 60

[3.1]target\_Y.shape: (25103, 3)

[3.2] DEBUG All technical indicators Extracted Count: 294

[4] Calculation of correlation strength. What are the +-100 best technical indicators and which are noise

[4.1]uncorrelate\_selection path: data/columns\_select/600010\_SH\_\_1min\_\_test\_A1\_\_columns.pkl

created\_json\_feature\_selection path: data/columns\_select/600010\_SH\_\_1min\_\_test\_A1\_\_corr.json

data/columns\_select/600010\_SH\_\_1min\_\_test\_A1\_\_corr.json

[4.2] Select the best technical patterns to train with features\_W3 Extracted Count: 86 Names : mtum\_AO\_5\_34,mtum\_QQEl\_14\_5\_4236,mtum\_FISHERTs\_9\_1,ichi\_senkou\_a,ti\_supertrend\_20,cycl\_PHASOR\_quad,cama\_r3,ma\_WMA\_50,olap\_BBAND\_UPPER,tend\_PSARr\_002\_02,wood\_pp,cycl\_DCPHASE,mtum\_CFO\_9,mtum\_CTI\_12,ma\_DEMA\_5,tend\_PSARl\_002\_02,volu\_PVT,mtum\_ADX,mtum\_ULTOSC,sti\_STDDEV,mtum\_FISHERT\_9\_1,mtum\_ADXR,demark\_r1,day\_minute,clas\_r1,tend\_VHF\_28,wood\_r3,mtum\_STOCH\_d,ma\_TEMA\_5,cycl\_DCPERIOD,cycl\_SINE\_sine,ma\_EMA\_100,mtum\_WILLIAMS\_R,cama\_s2,volu\_PVOL,sti\_BETA,clas\_r2,olap\_SAREXT,vola\_THERMOma\_20\_2\_05,day\_week,mtum\_ER\_10,mtum\_MINUS\_DM,perf\_ha,ma\_SMA\_10,mtum\_MINUS\_DI,ti\_kelt\_20\_upper,mtum\_PVOs\_12\_26\_9,mtum\_BEARP\_13,fibo\_s2,ti\_konk\_bl,vola\_ATR,mtum\_td\_seq\_sig,ti\_choppiness\_14,ti\_kelt\_20\_lower,mtum\_D\_9\_3,mtum\_STCstoch\_10\_12\_26\_05,wood\_r1,olap\_ALMA\_10\_60\_085,vola\_NATR,wood\_s1,olap\_SAR,mtum\_PVO\_12\_26\_9,mtum\_K\_9\_3,mtum\_AR\_26,sti\_CORREL,mtum\_MACD\_fix\_signal,volu\_NVI\_1,olap\_BBAND\_dif,mtum\_PLUS\_DI,mtum\_PLUS\_DM,ti\_vortex\_pos\_5,clas\_pp,olap\_HT\_TRENDLINE,ma\_WMA\_100,mtum\_STC\_10\_12\_26\_05,mtum\_DX,clas\_r3,ma\_KAMA\_10,clas\_s2,cama\_s3,mtum\_MACD\_signal,volume,mtum\_PVOh\_12\_26\_9,sti\_LINEARREG\_ANGLE,sti\_VAR,clas\_s3

[4.3] features\_W3 index Dates: from 2020-09-01 11:26:00+00:00 to 2021-02-22 10:26:00+00:00 Shape: (25103, 86)

[5] For correct training, for correct training the values must be normalised to between 0 and 1

[5.1] Normalise\_data path: data/scalers/600010\_SH\_\_1min\_\_test\_A1\_\_scalex.pkl

[6] Currently you have for each target\_Y value a row of technical indicators, you add a 'window' to make the decision to predict whether the +-48 rows above will be taken (about 4 hours of previous indicators, splited in 5min).

[6.1] get\_window\_data Shapes Y: (25074, 3) X: (25074, 30, 86)

[7] data split between training and validation

[7.1] Shapes: X\_train: (20059, 30, 86) y\_train: (20059, 3) index\_train: (20059,)

[8] Ground True data are unbalanced Given that there is a lot of 'do nothing' 0, and very little 'do buy' 1 or 'do sell' 2, weight balancing is required, to give more importance to the minorities.

[8.1] Class weight path: data/class\_weight/600010\_SH\_\_1min\_\_test\_A1\_\_class\_weight.pkl Dict: {0: 0.5436043360433604, 1: 5.115786789084417, 2: 1.0363194874974169}

[9] Creation of the TF model architecture. must respect the input\_shape and output\_shape and the 'softmax' , from there EXPERIMENT combinations

[9.1] arrays to use the TF model , input\_shape: (30, 86) output\_shape: 3

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 30, 144) 12528

dense\_1 (Dense) (None, 30, 72) 10440

dropout (Dropout) (None, 30, 72) 0

dense\_2 (Dense) (None, 30, 36) 2628

dropout\_1 (Dropout) (None, 30, 36) 0

dense\_3 (Dense) (None, 30, 12) 444

flatten (Flatten) (None, 360) 0

dense\_4 (Dense) (None, 3) 1083

=================================================================

Total params: 27123 (105.95 KB)

Trainable params: 27123 (105.95 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

[9.2] optimizer config : {'name': 'Adam', 'weight\_decay': None, 'clipnorm': None, 'global\_clipnorm': None, 'clipvalue': None, 'use\_ema': False, 'ema\_momentum': 0.99, 'ema\_overwrite\_frequency': None, 'jit\_compile': False, 'is\_legacy\_optimizer': False, 'learning\_rate': 0.001, 'beta\_1': 0.9, 'beta\_2': 0.999, 'epsilon': 1e-07, 'amsgrad': False}

You must install pydot (`pip install pydot`) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) for plot\_model to work.

[9.3] print diagram of the TF model Path: outputs/plots/600010\_SH\_\_1min\_\_test\_A1\_\_buysell.png

[10] Start training TF model outputs/600010\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

Epoch 1/90

840/840 [==============================] - 5s 4ms/step - loss: 0.9333 - categorical\_accuracy: 0.6152 - val\_loss: 0.9977 - val\_categorical\_accuracy: 0.5370

Epoch 2/90

840/840 [==============================] - 3s 4ms/step - loss: 0.8253 - categorical\_accuracy: 0.6534 - val\_loss: 0.9204 - val\_categorical\_accuracy: 0.6240

Epoch 3/90

840/840 [==============================] - 3s 4ms/step - loss: 0.7775 - categorical\_accuracy: 0.6743 - val\_loss: 0.8931 - val\_categorical\_accuracy: 0.5801

Epoch 4/90

840/840 [==============================] - 4s 4ms/step - loss: 0.7450 - categorical\_accuracy: 0.6905 - val\_loss: 0.8563 - val\_categorical\_accuracy: 0.6337

Epoch 5/90

840/840 [==============================] - 4s 4ms/step - loss: 0.7119 - categorical\_accuracy: 0.6992 - val\_loss: 0.9809 - val\_categorical\_accuracy: 0.5462

Epoch 6/90

840/840 [==============================] - 4s 4ms/step - loss: 0.6834 - categorical\_accuracy: 0.7101 - val\_loss: 0.7402 - val\_categorical\_accuracy: 0.6801

Epoch 7/90

840/840 [==============================] - 4s 4ms/step - loss: 0.6508 - categorical\_accuracy: 0.7288 - val\_loss: 0.9162 - val\_categorical\_accuracy: 0.6514

Epoch 8/90

840/840 [==============================] - 4s 4ms/step - loss: 0.6091 - categorical\_accuracy: 0.7415 - val\_loss: 0.9044 - val\_categorical\_accuracy: 0.6335

Epoch 9/90

840/840 [==============================] - 4s 4ms/step - loss: 0.5439 - categorical\_accuracy: 0.7594 - val\_loss: 0.7593 - val\_categorical\_accuracy: 0.6943

Epoch 10/90

840/840 [==============================] - 4s 4ms/step - loss: 0.5103 - categorical\_accuracy: 0.7754 - val\_loss: 0.8786 - val\_categorical\_accuracy: 0.6434

Epoch 11/90

840/840 [==============================] - 4s 5ms/step - loss: 0.4878 - categorical\_accuracy: 0.7865 - val\_loss: 0.8628 - val\_categorical\_accuracy: 0.6486

Epoch 12/90

840/840 [==============================] - 4s 4ms/step - loss: 0.4232 - categorical\_accuracy: 0.8131 - val\_loss: 0.9307 - val\_categorical\_accuracy: 0.6502

Epoch 13/90

840/840 [==============================] - 4s 4ms/step - loss: 0.4111 - categorical\_accuracy: 0.8187 - val\_loss: 0.9869 - val\_categorical\_accuracy: 0.6029

Epoch 14/90

840/840 [==============================] - 4s 5ms/step - loss: 0.3633 - categorical\_accuracy: 0.8412 - val\_loss: 1.1130 - val\_categorical\_accuracy: 0.6085

Epoch 15/90

840/840 [==============================] - 4s 4ms/step - loss: 0.3404 - categorical\_accuracy: 0.8510 - val\_loss: 1.0724 - val\_categorical\_accuracy: 0.6168

Epoch 16/90

840/840 [==============================] - 4s 4ms/step - loss: 0.3156 - categorical\_accuracy: 0.8644 - val\_loss: 1.2480 - val\_categorical\_accuracy: 0.6319

Epoch 16: early stopping

[10.1] Model initial\_weights saved : data/initial\_weights/600010\_SH\_\_1min\_\_test\_A1\_\_initial\_weights.tf

[11] Do a predict and eval. Load path: outputs/600010\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

[11.1] array to predict x\_test.shape: (5015, 30, 86)

836/836 [==============================] - 1s 867us/step

[11.2] read the evaluation: (5015, 30, 86)

(5015,) (5015,)

precision recall f1-score support

0 0.92 0.81 0.86 3585

1 0.13 0.18 0.15 261

2 0.54 0.69 0.61 1169

accuracy 0.75 5015

macro avg 0.53 0.56 0.54 5015

weighted avg 0.79 0.75 0.76 5015

outputs/model\_info/600010\_SH\_\_1min\_\_test\_A1\_.csv

count per%

y\_pred y\_test

0 0 2894 57.71

2 2 811 16.17

0 557 11.11

1 2 180 3.59

0 2 178 3.55

1 0 134 2.67

2 1 127 2.53

0 1 88 1.75

1 1 46 0.92

outputs/model\_info/600010\_SH\_\_1min\_\_test\_A1\_\_.info.txt

END

进程已结束，退出代码为 0

片仔癀

##茅台：

[4.2] 选择最佳技术模式来训练特征，提取计数：88

mtum\_CTI\_12,cycl\_SINE\_lead,vola\_HWU,mtum\_STCmacd\_10\_12\_26\_05,clas\_s3,mtum\_MINUS\_DM,demark\_s1,mtum\_CG\_10,open,cycl\_PHASOR\_inph,mtum\_TRIX,vola\_NATR,tend\_renko\_TR,tend\_LDECAY\_5,volu\_PVT,ma\_SMA\_50,sti\_BETA,olap\_MIDPRICE,mtum\_QQEs\_14\_5\_4236,high,low,ti\_mass\_index\_9\_25,ichi\_senkou\_b,demark\_pp,mtum\_DX,mtum\_PVOs\_12\_26\_9,cama\_r3,mtum\_STOCH\_Fa\_kd,olap\_SAREXT,mtum\_MFI,vola\_THERMOma\_20\_2\_05,mtum\_CFO\_9,mtum\_PVO\_12\_26\_9,ma\_SMA\_100,mtum\_MACD\_fix,mtum\_QQEl\_14\_5\_4236,vola\_ATR,mtum\_FISHERT\_9\_1,tend\_PSARl\_002\_02,ma\_SMA\_20,vola\_THERMOl\_20\_2\_05,volu\_NVI\_1,sti\_VAR,day\_minute,mtum\_td\_seq,mtum\_INERTIA\_20\_14,mtum\_MACD,volume,close,fibo\_s1,ma\_TEMA\_5,cama\_s3,vola\_KCUe\_20\_2,mtum\_ER\_10,cycl\_SINE\_sine,vola\_TRANGE,tend\_PSARs\_002\_02,sti\_LINEARREG\_INTERCEPT,cama\_s2,vola\_KCLe\_20\_2,mtum\_BOP,vola\_THERMO\_20\_2\_05,ma\_WMA\_10,cycl\_PHASOR\_quad,volu\_PVOL,sti\_STDDEV,cycl\_DCPERIOD,ti\_kelt\_20\_lower,sti\_CORREL,mtum\_STOCH\_kd,olap\_PWMA\_10,mtum\_ADXR,ichi\_senkou\_a,tend\_VHF\_28,volu\_PVI\_1,olap\_BBAND\_dif,olap\_SINWMA\_14,tend\_PSARr\_002\_02,olap\_ALMA\_10\_60\_085,ti\_konk\_bl,sti\_LINEARREG\_ANGLE,perf\_ha,mtum\_PLUS\_DM,mtum\_PVOh\_12\_26\_9,mtum\_ADX,cycl\_DCPHASE,ti\_choppiness\_14,olap\_BBAND\_LOWER

茅台：89 Names : cycl\_SINE\_sine,mtum\_STOCH\_Fa\_kd,mtum\_DX,ichi\_senkou\_a,ma\_TEMA\_5,mtum\_ER\_10,open,vola\_HWU,vola\_THERMOma\_20\_2\_05,fibo\_s1,ti\_kelt\_20\_lower,cama\_s2,ma\_SMA\_50,high,cycl\_DCPHASE,volu\_PVOL,tend\_renko\_TR,cycl\_DCPERIOD,mtum\_td\_seq,mtum\_MACD\_fix,vola\_TRANGE,tend\_PSARl\_002\_02,cama\_r3,mtum\_MACD,mtum\_MFI,tend\_PSARr\_002\_02,vola\_KCUe\_20\_2,mtum\_ADXR,sti\_STDDEV,mtum\_PLUS\_DM,volu\_NVI\_1,close,volu\_PVI\_1,mtum\_CG\_10,mtum\_PVOs\_12\_26\_9,vola\_KCLe\_20\_2,sti\_BETA,mtum\_BOP,olap\_PWMA\_10,vola\_THERMOl\_20\_2\_05,mtum\_FISHERT\_9\_1,demark\_s1,ti\_konk\_bl,mtum\_MINUS\_DM,vola\_ATR,sti\_CORREL,ma\_WMA\_10,ichi\_tenkan\_sen,mtum\_PVO\_12\_26\_9,ti\_choppiness\_14,clas\_s3,volu\_PVT,day\_minute,demark\_pp,mtum\_ADX,mtum\_QQEs\_14\_5\_4236,tend\_LDECAY\_5,ti\_mass\_index\_9\_25,olap\_ALMA\_10\_60\_085,olap\_SAREXT,ichi\_senkou\_b,cdl\_BELTHOLD,mtum\_PVOh\_12\_26\_9,olap\_MIDPRICE,sti\_VAR,ma\_SMA\_100,cama\_s3,vola\_THERMO\_20\_2\_05,cycl\_SINE\_lead,mtum\_CTI\_12,mtum\_INERTIA\_20\_14,mtum\_TRIX,mtum\_QQEl\_14\_5\_4236,vola\_NATR,tend\_PSARs\_002\_02,tend\_VHF\_28,mtum\_STOCH\_kd,sti\_LINEARREG\_INTERCEPT,olap\_BBAND\_LOWER,olap\_BBAND\_dif,mtum\_CFO\_9,cycl\_PHASOR\_quad,low,mtum\_STCmacd\_10\_12\_26\_05,olap\_SINWMA\_14,ma\_SMA\_20,sti\_LINEARREG\_ANGLE,cycl\_PHASOR\_inph,volume

precision recall f1-score support

0 0.99 0.97 0.98 32573

1 0.38 0.80 0.51 452

2 0.48 0.74 0.58 456

accuracy 0.97 33481

macro avg 0.62 0.83 0.69 33481

weighted avg 0.98 0.97 0.97 33481

outputs/model\_info/600519\_SH\_\_1min\_\_test\_A1\_.csv

count per%

y\_pred y\_test

0 0 31622 94.45

1 0 588 1.76

2 0 363 1.08

1 1 360 1.08

2 2 336 1.00

0 2 116 0.35

1 92 0.27

1 2 4 0.01

outputs/model\_info/600519\_SH\_\_1min\_\_test\_A1\_\_.info.txt

##山西汾酒

89 Names :

[1] Read data ohlcv data path ../d\_price/688111\_SH\_1min.csv Shape\_raw: (167613, 5)

[2] Calculating Technical indicators. stock: 688111\_SH

[2.1] Calculated Technical indicators. stock: 688111\_SH Tech indicator count: (165569, 316)

[3] Calculate the target\_Y (ground true) ,what is the target to detect? the highest and lowest peaks stock: 688111\_SH

get\_HT\_pp df.shape: (165569, 294) len\_right: 72 len\_left: 36

[3.1]target\_Y.shape: (165569, 3)

[3.2] DEBUG All technical indicators Extracted Count: 294

[4] Calculation of correlation strength. What are the +-100 best technical indicators and which are noise

[4.1]uncorrelate\_selection path: data/columns\_select/688111\_SH\_\_1min\_\_test\_A1\_\_columns.pkl

created\_json\_feature\_selection path: data/columns\_select/688111\_SH\_\_1min\_\_test\_A1\_\_corr.json

data/columns\_select/688111\_SH\_\_1min\_\_test\_A1\_\_corr.json

[4.2] Select the best technical patterns to train with features\_W3 Extracted Count: 90 Names : vola\_THERMO\_20\_2\_05,mtum\_PVO\_12\_26\_9,cycl\_SINE\_lead,vola\_KCUe\_20\_2,cycl\_SINE\_sine,mtum\_STOCH\_Fa\_kd,perf\_ha,mtum\_STC\_10\_12\_26\_05,fibo\_s2,low,volu\_PVT,mtum\_QQEs\_14\_5\_4236,mtum\_STOCH\_kd,wood\_s3,volume,mtum\_PVOh\_12\_26\_9,sti\_ZS\_30,olap\_BBAND\_dif,high,day\_minute,cycl\_PHASOR\_quad,mtum\_PLUS\_DI,mtum\_PLUS\_DM,clas\_s2,volu\_PVI\_1,mtum\_BOP,volu\_NVI\_1,demark\_pp,cama\_r2,ichi\_senkou\_a,wood\_r3,ma\_TRIMA\_10,tend\_PSARr\_002\_02,olap\_MIDPRICE,ti\_choppiness\_14,vola\_ATR,ti\_mass\_index\_9\_25,sti\_STDDEV,cama\_s2,sti\_CORREL,volu\_PVOL,ichi\_senkou\_b,mtum\_STOCH\_d,ma\_WMA\_100,mtum\_td\_seq,olap\_SAR,ti\_donchian\_upper\_20,cycl\_PHASOR\_inph,vola\_NATR,mtum\_MINUS\_DM,mtum\_CTI\_12,sti\_BETA,sti\_LINEARREG\_INTERCEPT,close,mtum\_RSI,cama\_r1,tend\_VHF\_28,mtum\_PVOs\_12\_26\_9,mtum\_MACD\_fix\_signal,mtum\_MACD\_list,ti\_hma\_20,fibo\_r1,mtum\_ADXR,vola\_TRANGE,mtum\_SMI\_5\_20\_5,ma\_EMA\_100,fibo\_s1,demark\_r1,vola\_THERMOma\_20\_2\_05,mtum\_ER\_10,vola\_KCLe\_20\_2,sti\_TSF,mtum\_AR\_26,sti\_VAR,ma\_SMA\_100,vola\_THERMOl\_20\_2\_05,olap\_BBAND\_LOWER,mtum\_MACD\_fix\_list,ma\_EMA\_50,fibo\_r2,cycl\_DCPERIOD,mtum\_MFI,mtum\_MINUS\_DI,olap\_ALMA\_10\_60\_085,mtum\_STCmacd\_10\_12\_26\_05,cycl\_DCPHASE,demark\_s1,mtum\_CFO\_9,cama\_s3,tend\_renko\_TR

[4.3] features\_W3 index Dates: from 2020-09-01 11:26:00+00:00 to 2023-08-01 14:56:00+00:00 Shape: (165569, 90)

[5] For correct training, for correct training the values must be normalised to between 0 and 1

[5.1] Normalise\_data path: data/scalers/688111\_SH\_\_1min\_\_test\_A1\_\_scalex.pkl

[6] Currently you have for each target\_Y value a row of technical indicators, you add a 'window' to make the decision to predict whether the +-48 rows above will be taken (about 4 hours of previous indicators, splited in 5min).

[6.1] get\_window\_data Shapes Y: (165540, 3) X: (165540, 30, 90)

[7] data split between training and validation

[7.1] Shapes: X\_train: (132432, 30, 90) y\_train: (132432, 3) index\_train: (132432,)

[8] Ground True data are unbalanced Given that there is a lot of 'do nothing' 0, and very little 'do buy' 1 or 'do sell' 2, weight balancing is required, to give more importance to the minorities.

[8.1] Class weight path: data/class\_weight/688111\_SH\_\_1min\_\_test\_A1\_\_class\_weight.pkl Dict: {0: 0.3423076923076923, 1: 26.13617525162818, 2: 24.758272574312954}

[9] Creation of the TF model architecture. must respect the input\_shape and output\_shape and the 'softmax' , from there EXPERIMENT combinations

[9.1] arrays to use the TF model , input\_shape: (30, 90) output\_shape: 3

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 30, 144) 13104

dense\_1 (Dense) (None, 30, 72) 10440

dropout (Dropout) (None, 30, 72) 0

dense\_2 (Dense) (None, 30, 36) 2628

dropout\_1 (Dropout) (None, 30, 36) 0

dense\_3 (Dense) (None, 30, 12) 444

flatten (Flatten) (None, 360) 0

dense\_4 (Dense) (None, 3) 1083

=================================================================

Total params: 27699 (108.20 KB)

Trainable params: 27699 (108.20 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

[9.2] optimizer config : {'name': 'Adam', 'weight\_decay': None, 'clipnorm': None, 'global\_clipnorm': None, 'clipvalue': None, 'use\_ema': False, 'ema\_momentum': 0.99, 'ema\_overwrite\_frequency': None, 'jit\_compile': False, 'is\_legacy\_optimizer': False, 'learning\_rate': 0.001, 'beta\_1': 0.9, 'beta\_2': 0.999, 'epsilon': 1e-07, 'amsgrad': False}

You must install pydot (pip install pydot) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) for plot\_model to work.

[9.3] print diagram of the TF model Path: outputs/plots/688111\_SH\_\_1min\_\_test\_A1\_\_buysell.png

[11.1] array to predict x\_test.shape: (33108, 30, 90)

5518/5518 [==============================] - 5s 857us/step

[11.2] read the evaluation: (33108, 30, 90)

(33108,) (33108,)

precision recall f1-score support

0 1.00 0.94 0.97 32246

1 0.25 0.88 0.38 432

2 0.30 0.88 0.44 430

accuracy 0.93 33108

macro avg 0.51 0.90 0.60 33108

weighted avg 0.98 0.93 0.95 33108

outputs/model\_info/688111\_SH\_\_1min\_\_test\_A1\_.csv

count per%

y\_pred y\_test

0 0 30175 91.14

1 0 1173 3.54

2 0 898 2.71

1 1 382 1.15

2 2 378 1.14

0 1 49 0.15

2 49 0.15

1 2 3 0.01

2 1 1 0.00

北向资金对大盘涨跌的指示性作用

选股策略回测

REITs基金净值研究报告

大模型使用指南

C:\Users\msc\Desktop\stocks-prediction-Machine-learning-RealTime-TensorFlow\venv\Scripts\python.exe C:\Users\msc\Desktop\stocks-prediction-Machine-learning-RealTime-TensorFlow\Tutorial\RUN\_buy\_sell\_Tutorial\_3W\_5min\_RT.py

cuda\_malloc\_async

[1] Read data ohlcv data path ../d\_price/600000\_SH\_1min.csv Shape\_raw: (168266, 5)

[2] Calculating Technical indicators. stock: 600000\_SH

[2.1] Calculated Technical indicators. stock: 600000\_SH Tech indicator count: (161816, 316)

open high ... tend\_renko\_brick tend\_renko\_change

Date ...

2020-09-01 11:26:00+00:00 0.000000 0.0 ... 0.0 0

2020-09-01 11:27:00+00:00 0.000000 0.0 ... 0.0 0

2020-09-01 11:28:00+00:00 0.000000 0.0 ... 0.0 0

2020-09-01 11:29:00+00:00 -0.008629 0.0 ... 0.0 0

2020-09-01 13:00:00+00:00 0.008629 0.0 ... 0.0 0

... ... ... ... ... ...

2023-08-01 14:52:00+00:00 -0.010000 0.0 ... 0.0 0

2023-08-01 14:53:00+00:00 0.010000 0.0 ... 0.0 0

2023-08-01 14:54:00+00:00 0.000000 0.0 ... 0.0 0

2023-08-01 14:55:00+00:00 0.000000 0.0 ... 0.0 0

2023-08-01 14:56:00+00:00 0.000000 0.0 ... 0.0 0

[161816 rows x 294 columns]

[3] Calculate the target\_Y (ground true) ,what is the target to detect? the highest and lowest peaks stock: 600000\_SH

get\_HT\_pp df.shape: (161816, 294) len\_right: 72 len\_left: 36

[3.1]target\_Y.shape: (161816, 3)

[3.2] DEBUG All technical indicators Extracted Count: 294

[4] Calculation of correlation strength. What are the +-100 best technical indicators and which are noise

[4.1]uncorrelate\_selection path: data/columns\_select/600000\_SH\_\_1min\_\_test\_A1\_\_columns.pkl

created\_json\_feature\_selection path: data/columns\_select/600000\_SH\_\_1min\_\_test\_A1\_\_corr.json

data/columns\_select/600000\_SH\_\_1min\_\_test\_A1\_\_corr.json

[4.2] Select the best technical patterns to train with features\_W3 Extracted Count: 88 Names : clas\_s3,olap\_SAREXT,mtum\_D\_9\_3,mtum\_QQEl\_14\_5\_4236,cycl\_EBSW\_40\_10,mtum\_STOCH\_d,mtum\_MACD\_list,mtum\_MINUS\_DM,volu\_NVI\_1,cama\_r2,ti\_choppiness\_14,mtum\_AR\_26,tend\_renko\_TR,ichi\_senkou\_b,clas\_pp,day\_minute,sti\_LINEARREG\_INTERCEPT,mtum\_SMI\_5\_20\_5,mtum\_STOCH\_Fa\_kd,ma\_SMA\_100,sti\_STDDEV,mtum\_PVO\_12\_26\_9,mtum\_PVOh\_12\_26\_9,ichi\_kijun\_sen,vola\_TRANGE,cycl\_DCPERIOD,mtum\_FISHERT\_9\_1,ti\_coppock\_14\_11\_10,volu\_PVI\_1,olap\_MIDPRICE,cycl\_PHASOR\_inph,mtum\_PLUS\_DM,ma\_SMA\_50,mtum\_td\_seq\_sig,clas\_r2,perf\_ha,mtum\_ADX,vola\_THERMOs\_20\_2\_05,mtum\_STOCH\_RSI\_k,fibo\_r2,mtum\_ER\_10,tend\_PSARs\_002\_02,vola\_THERMOl\_20\_2\_05,tend\_VHF\_28,volu\_PVT,mtum\_CG\_10,demark\_s1,mtum\_MFI,cycl\_PHASOR\_quad,high,ti\_mass\_index\_9\_25,ma\_WMA\_100,sti\_VAR,mtum\_FISHERTs\_9\_1,sti\_CORREL,tend\_PSARr\_002\_02,sti\_BETA,volu\_PVOL,vola\_THERMO\_20\_2\_05,olap\_SSF\_10\_2,ti\_konk\_bl,vola\_RVI\_14,olap\_ALMA\_10\_60\_085,olap\_BBAND\_dif,ichi\_tenkan\_sen,cama\_s2,mtum\_DX,mtum\_QQE\_14\_5\_4236\_RSIMA,mtum\_MINUS\_DI,mtum\_TRIX,cycl\_SINE\_sine,cycl\_SINE\_lead,olap\_SAR,mtum\_ADXR,mtum\_PGO\_14,volu\_Chaikin\_AD,cama\_s3,ti\_konk\_bro,vola\_THERMOma\_20\_2\_05,close,mtum\_INERTIA\_20\_14,low,vola\_NATR,clas\_r3,ichi\_senkou\_a,vola\_ATR,mtum\_PVOs\_12\_26\_9,olap\_PWMA\_10

[4.3] features\_W3 index Dates: from 2020-09-01 11:26:00+00:00 to 2023-08-01 14:56:00+00:00 Shape: (161816, 88)

[5] For correct training, for correct training the values must be normalised to between 0 and 1

[5.1] Normalise\_data path: data/scalers/600000\_SH\_\_1min\_\_test\_A1\_\_scalex.pkl

[6] Currently you have for each target\_Y value a row of technical indicators, you add a 'window' to make the decision to predict whether the +-48 rows above will be taken (about 4 hours of previous indicators, splited in 5min).

[6.1] get\_window\_data Shapes Y: (161787, 3) X: (161787, 30, 88)

[7] data split between training and validation

[7.1] Shapes: X\_train: (129429, 30, 88) y\_train: (129429, 3) index\_train: (129429,)

[8] Ground True data are unbalanced Given that there is a lot of 'do nothing' 0, and very little 'do buy' 1 or 'do sell' 2, weight balancing is required, to give more importance to the minorities.

[8.1] Class weight path: data/class\_weight/600000\_SH\_\_1min\_\_test\_A1\_\_class\_weight.pkl Dict: {0: 0.35931240682596133, 1: 9.071278385197646, 2: 9.3748370273794}

[9] Creation of the TF model architecture. must respect the input\_shape and output\_shape and the 'softmax' , from there EXPERIMENT combinations

[9.1] arrays to use the TF model , input\_shape: (30, 88) output\_shape: 3

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 30, 144) 12816

dense\_1 (Dense) (None, 30, 72) 10440

dropout (Dropout) (None, 30, 72) 0

dense\_2 (Dense) (None, 30, 36) 2628

dropout\_1 (Dropout) (None, 30, 36) 0

dense\_3 (Dense) (None, 30, 12) 444

flatten (Flatten) (None, 360) 0

dense\_4 (Dense) (None, 3) 1083

=================================================================

Total params: 27411 (107.07 KB)

Trainable params: 27411 (107.07 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

[9.2] optimizer config : {'name': 'Adam', 'weight\_decay': None, 'clipnorm': None, 'global\_clipnorm': None, 'clipvalue': None, 'use\_ema': False, 'ema\_momentum': 0.99, 'ema\_overwrite\_frequency': None, 'jit\_compile': False, 'is\_legacy\_optimizer': False, 'learning\_rate': 0.001, 'beta\_1': 0.9, 'beta\_2': 0.999, 'epsilon': 1e-07, 'amsgrad': False}

You must install pydot (`pip install pydot`) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) for plot\_model to work.

[9.3] print diagram of the TF model Path: outputs/plots/600000\_SH\_\_1min\_\_test\_A1\_\_buysell.png

[10] Start training TF model outputs/600000\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

Epoch 1/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.6332 - categorical\_accuracy: 0.6818 - val\_loss: 0.6394 - val\_categorical\_accuracy: 0.7730

Epoch 2/90

5420/5420 [==============================] - 21s 4ms/step - loss: 0.4140 - categorical\_accuracy: 0.7597 - val\_loss: 1.1394 - val\_categorical\_accuracy: 0.4893

Epoch 3/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3807 - categorical\_accuracy: 0.7737 - val\_loss: 0.6487 - val\_categorical\_accuracy: 0.7782

Epoch 4/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3620 - categorical\_accuracy: 0.7826 - val\_loss: 0.6592 - val\_categorical\_accuracy: 0.7844

Epoch 5/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3431 - categorical\_accuracy: 0.7889 - val\_loss: 0.6234 - val\_categorical\_accuracy: 0.7820

Epoch 6/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3424 - categorical\_accuracy: 0.7895 - val\_loss: 0.5092 - val\_categorical\_accuracy: 0.8278

Epoch 7/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3371 - categorical\_accuracy: 0.7954 - val\_loss: 0.6093 - val\_categorical\_accuracy: 0.8032

Epoch 8/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3303 - categorical\_accuracy: 0.7996 - val\_loss: 0.5449 - val\_categorical\_accuracy: 0.8061

Epoch 9/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3269 - categorical\_accuracy: 0.7982 - val\_loss: 0.4963 - val\_categorical\_accuracy: 0.8056

Epoch 10/90

5420/5420 [==============================] - 24s 4ms/step - loss: 0.3238 - categorical\_accuracy: 0.7994 - val\_loss: 0.5839 - val\_categorical\_accuracy: 0.8033

Epoch 11/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3135 - categorical\_accuracy: 0.8041 - val\_loss: 0.6300 - val\_categorical\_accuracy: 0.7895

Epoch 12/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3201 - categorical\_accuracy: 0.8020 - val\_loss: 0.4846 - val\_categorical\_accuracy: 0.8357

Epoch 13/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3134 - categorical\_accuracy: 0.8022 - val\_loss: 0.3968 - val\_categorical\_accuracy: 0.8474

Epoch 14/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3089 - categorical\_accuracy: 0.8048 - val\_loss: 0.5102 - val\_categorical\_accuracy: 0.8235

Epoch 15/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3106 - categorical\_accuracy: 0.8048 - val\_loss: 0.5067 - val\_categorical\_accuracy: 0.8208

Epoch 16/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3122 - categorical\_accuracy: 0.8039 - val\_loss: 0.3959 - val\_categorical\_accuracy: 0.8511

Epoch 17/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3096 - categorical\_accuracy: 0.8053 - val\_loss: 0.4999 - val\_categorical\_accuracy: 0.8278

Epoch 18/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.3012 - categorical\_accuracy: 0.8084 - val\_loss: 0.7470 - val\_categorical\_accuracy: 0.7352

Epoch 19/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3024 - categorical\_accuracy: 0.8087 - val\_loss: 0.4606 - val\_categorical\_accuracy: 0.8315

Epoch 20/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3058 - categorical\_accuracy: 0.8064 - val\_loss: 0.4899 - val\_categorical\_accuracy: 0.8230

Epoch 21/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.3026 - categorical\_accuracy: 0.8084 - val\_loss: 0.3821 - val\_categorical\_accuracy: 0.8514

Epoch 22/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2967 - categorical\_accuracy: 0.8104 - val\_loss: 0.4738 - val\_categorical\_accuracy: 0.8224

Epoch 23/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.2976 - categorical\_accuracy: 0.8113 - val\_loss: 0.5265 - val\_categorical\_accuracy: 0.7963

Epoch 24/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.2881 - categorical\_accuracy: 0.8111 - val\_loss: 0.5670 - val\_categorical\_accuracy: 0.8039

Epoch 25/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2930 - categorical\_accuracy: 0.8130 - val\_loss: 0.4658 - val\_categorical\_accuracy: 0.8443

Epoch 26/90

5420/5420 [==============================] - 22s 4ms/step - loss: 0.2933 - categorical\_accuracy: 0.8146 - val\_loss: 0.4468 - val\_categorical\_accuracy: 0.8472

Epoch 27/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2854 - categorical\_accuracy: 0.8166 - val\_loss: 0.4622 - val\_categorical\_accuracy: 0.8379

Epoch 28/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2850 - categorical\_accuracy: 0.8166 - val\_loss: 0.5829 - val\_categorical\_accuracy: 0.8039

Epoch 29/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2840 - categorical\_accuracy: 0.8126 - val\_loss: 0.3778 - val\_categorical\_accuracy: 0.8539

Epoch 30/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2819 - categorical\_accuracy: 0.8147 - val\_loss: 0.4244 - val\_categorical\_accuracy: 0.8478

Epoch 31/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2827 - categorical\_accuracy: 0.8164 - val\_loss: 0.5466 - val\_categorical\_accuracy: 0.8009

Epoch 32/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2834 - categorical\_accuracy: 0.8147 - val\_loss: 0.4470 - val\_categorical\_accuracy: 0.8465

Epoch 33/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2805 - categorical\_accuracy: 0.8176 - val\_loss: 0.4500 - val\_categorical\_accuracy: 0.8300

Epoch 34/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2811 - categorical\_accuracy: 0.8131 - val\_loss: 0.3572 - val\_categorical\_accuracy: 0.8630

Epoch 35/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2824 - categorical\_accuracy: 0.8179 - val\_loss: 0.4878 - val\_categorical\_accuracy: 0.8284

Epoch 36/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2710 - categorical\_accuracy: 0.8205 - val\_loss: 0.3725 - val\_categorical\_accuracy: 0.8575

Epoch 37/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2776 - categorical\_accuracy: 0.8157 - val\_loss: 0.3641 - val\_categorical\_accuracy: 0.8652

Epoch 38/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2742 - categorical\_accuracy: 0.8183 - val\_loss: 0.3162 - val\_categorical\_accuracy: 0.8730

Epoch 39/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2709 - categorical\_accuracy: 0.8211 - val\_loss: 0.4028 - val\_categorical\_accuracy: 0.8564

Epoch 40/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2756 - categorical\_accuracy: 0.8176 - val\_loss: 0.4192 - val\_categorical\_accuracy: 0.8462

Epoch 41/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2703 - categorical\_accuracy: 0.8201 - val\_loss: 0.4237 - val\_categorical\_accuracy: 0.8416

Epoch 42/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2644 - categorical\_accuracy: 0.8230 - val\_loss: 0.4636 - val\_categorical\_accuracy: 0.8306

Epoch 43/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2707 - categorical\_accuracy: 0.8217 - val\_loss: 0.4256 - val\_categorical\_accuracy: 0.8481

Epoch 44/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2713 - categorical\_accuracy: 0.8176 - val\_loss: 0.3963 - val\_categorical\_accuracy: 0.8562

Epoch 45/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2734 - categorical\_accuracy: 0.8215 - val\_loss: 0.3979 - val\_categorical\_accuracy: 0.8494

Epoch 46/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2665 - categorical\_accuracy: 0.8234 - val\_loss: 0.3688 - val\_categorical\_accuracy: 0.8595

Epoch 47/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2658 - categorical\_accuracy: 0.8202 - val\_loss: 0.5208 - val\_categorical\_accuracy: 0.8283

Epoch 48/90

5420/5420 [==============================] - 23s 4ms/step - loss: 0.2656 - categorical\_accuracy: 0.8236 - val\_loss: 0.3696 - val\_categorical\_accuracy: 0.8602

Epoch 48: early stopping

[10.1] Model initial\_weights saved : data/initial\_weights/600000\_SH\_\_1min\_\_test\_A1\_\_initial\_weights.tf

[11] Do a predict and eval. Load path: outputs/600000\_SH\_\_1min\_\_test\_A1\_\_buysell.h5

[11.1] array to predict x\_test.shape: (32358, 30, 88)

5393/5393 [==============================] - 5s 876us/step

[11.2] read the evaluation: (32358, 30, 88)

(32358,) (32358,)

precision recall f1-score support

0 0.98 0.86 0.92 29928

1 0.31 0.80 0.45 1239

2 0.35 0.83 0.49 1191

accuracy 0.86 32358

macro avg 0.55 0.83 0.62 32358

weighted avg 0.93 0.86 0.89 32358

outputs/model\_info/600000\_SH\_\_1min\_\_test\_A1\_.csv

count per%

y\_pred y\_test

0 0 25879 79.98

1 0 2198 6.79

2 0 1851 5.72

1 1 996 3.08

2 2 990 3.06

0 1 236 0.73

0 2 195 0.60

2 1 7 0.02

1 2 6 0.02

outputs/model\_info/600000\_SH\_\_1min\_\_test\_A1\_\_.info.txt

END

进程已结束，退出代码为 0