Spark-Stock

葉津源、陳冠甫

predicting realtime stock price
with deep-learning model in spark

什麼資料

值得預測？


SELL
BUY

goal:

predict whether next close value will be higher or lower

data:

10 consecutive days of 15 stock close prices

try: tech, etf*, random

etf: Exchange-traded fund

Training

Stock RealTime API:
Alpha Vantage

ALPHA VANTAGE

TensorFlow

Neural Network Model

daily value

README.md

SparkFlow

This is an implementation of TensorFlow on Spark.
in using TensorFlow on Spark. With SparkFlow, you
Pipeline. Underneath, SparkFlow uses a parameter
Through the api, the user can specify the style of t

build passing license MIT

pyspark model

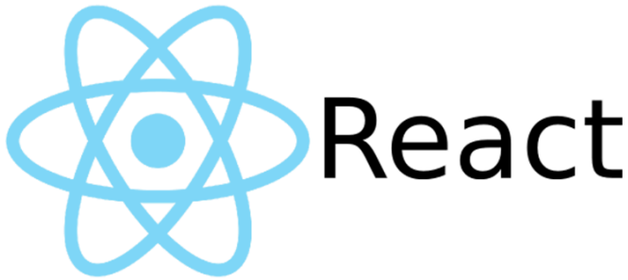
TSM.csv ... → **dataframe** ← --- **.fit( )** --- **pyspark Pipeline**

(datetime, close, open, high, low, volumn)

(is next close value higher or lower than current?)

PySpark

Predicting



Stock RealTime API:
Alpha Vantage

web visualize

realtime value

realtime datastore

TSM.csv ...

(datetime, close,
open, high,
low, volumn)

pyspark model **predict( )** **result**

collect data with alpha_vantage

```
targets_tech = ['GOOGL','FB','MSFT','AAPL','INTC','TSM','ORCL',
            'IBM','NVDA','ADBE','TXN','AVGO','ACN','CRM','QCOM']
targets_etf = ['MS','VFH','TSM','IXG','RYF','UYG','DFNL','PSCF',
            'IAK','KBWP','CHIX','BDCS','KBWR','PFI','JHMF']
targets_rand = ['CY','KHC','AMAT','EBAY','URBN','ROST','ADI',
            'LRCX','RRGB', 'MCD', 'TER', 'ACGL', 'TSCO', 'TIVO', 'TSM']


datadir = 'data'

def main_collect():
    print('check data')
    if not os.path.exists(datadir):
        os.mkdir(datadir)
    if len(os.listdir(datadir)) == 0:
        print('     ## collecting all data##     ')
        for targets in [targets_tech, targets_etf, targets_rand]:
            collect_data(datadir, targets)
```

← list of stocks

```
from alpha_vantage.timeseries import TimeSeries
```

```
def collect_data(datadir, targets):
    for symbol in tqdm(targets):
        ts = TimeSeries(key='PCAG643Q0SFCE6WI',output_format='csv')
        data, meta_data = ts.get_daily(symbol, outputsize='full')

        filepath = os.path.join(datadir, symbol+'.csv')

        with open(filepath, 'w') as f:
            writer = csv.writer(f)
            writer.writerows(data)

        time.sleep(14)
```

preprocess data with pyspark

```
#collect and process data from csvs of one target list
print('==start to preprocess for %s ==')%task)

schema = StructType([
    StructField('timestamp', StringType(), True),
    StructField('open', StringType(), True),
    StructField('high', StringType(), True),
    StructField('low', StringType(), True),
    StructField('close', StringType(), True),
    StructField('volume', StringType(), True)
    ])
```

```
for symbol in targets:
    try:
        filepath = os.path.join(datadir, symbol+'.csv')
        raw_data = sc.textFile(filepath)
        rdd = raw_data.map(lambda x: x.split(",")).filter(lambda x: x[0]!='timestamp')
        df = sqlContext.createDataFrame(rdd, schema)
        df = df.drop('open').drop('high').drop('low').drop('volume')
        df = df.withColumn('close', df['close'].cast(DoubleType()))
        dic = list(map(lambda row: row.asDict(), df.collect()))
        dic_list.append(dic)
        if symbol == 'TSM':
            tsm_dic = dic
        if len(dic) < min_count:
            min_count = len(dic)
        print(symbol, 'done')
```

```
process_dic = list()
for i in range(11,min_count):
    doc = list()
    doc.append(dic_list[0][i]['timestamp'])
    for dic in dic_list:
        for j in range(i-10,i):
            doc.append(dic[j]['close'])
    if tsm_dic[i-10]['close'] > tsm_dic[i-11]['close']:
        doc.append(0)
    else:
        doc.append(1)

    process_dic.append(doc)

final_schema = [StructField('timestamp', StringType(), True)]
for i in range(0,150):
    final_schema.append(StructField(str(i), DoubleType(), True))
final_schema.append(StructField('result', IntegerType(), True))
final_schema = StructType(final_schema)

final_rdd = sc.parallelize(process_dic)
final_df = sqlContext.createDataFrame(final_rdd, final_schema)
```

convert tensorflow model to spark model with sparkflow,
train with spark pipeline

```python
import tensorflow as tf

def small_model():
    x = tf.placeholder(tf.float64, shape=[None, 150], name='x')
    y = tf.placeholder(tf.float32, shape=[None, 2], name='y')
    layer1 = tf.layers.dense(x, 256, activation=tf.nn.relu)
    layer2 = tf.layers.dense(layer1, 256, activation=tf.nn.relu)
    out = tf.layers.dense(layer2, 2)
    z = tf.argmax(out, 1, name='out')
    loss = tf.losses.softmax_cross_entropy(y, out)
    return loss
```

```python
mg = build_graph(small_model)
#Assemble and one hot encode
va = VectorAssembler(inputCols=final_df.columns[1:151], outputCol='features')
encoded = OneHotEncoder(inputCol='result', outputCol='labels', dropLast=False)
adam_config = build_adam_config(learning_rate=0.001, beta1=0.9, beta2=0.999)

spark_model = SparkAsyncDL(
    inputCol='features',
    tensorflowGraph=mg,
    tfInput='x:0',
    tfLabel='y:0',
    tfOutput='out:0',
    tfLearningRate=.001,
    iters=200,
    predictionCol='predicted',
    labelCol='labels',
    verbose=1,
    optimizerOptions=adam_config
)

print('save model in ckptpath')
p = Pipeline(stages=[va, encoded, spark_model]).fit(final_df)
p.write().overwrite().save(ckptdir)

print('task all done')
```

column in dataframe

variable in tf model

150 -> 256

relu activation

256 -> 256

relu activation

256 -> 2

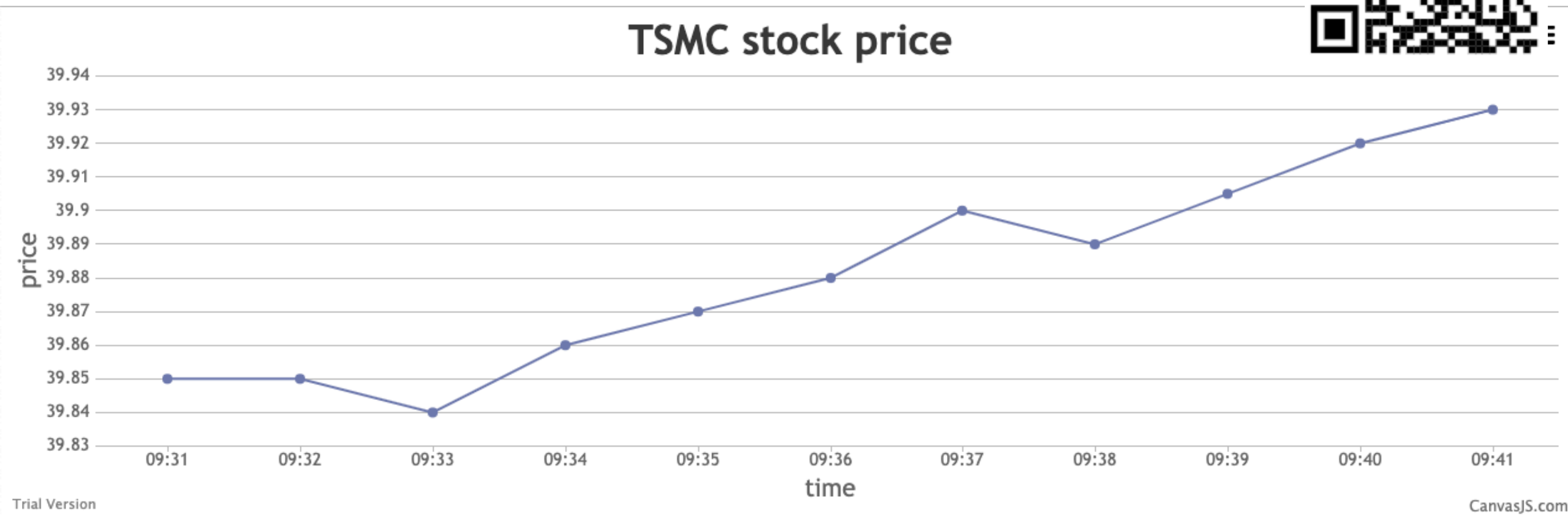
training screenshots

```
* Serving Flask app "sparkflow.HogwildSparkModel" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
```

```
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 0, Loss: 371.149323
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 0, Loss: 46.221939
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 0, Loss: 143.617676
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 0, Loss: 35.406776
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 1, Loss: 139.247696
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 1, Loss: 197.460190
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 1, Loss: 96.722649
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 1, Loss: 36.886974
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 2, Loss: 96.562553
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 2, Loss: 121.998642
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 2, Loss: 70.031273
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 2, Loss: 106.950455
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 3, Loss: 82.900932
```

```
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 90, Loss: 0.694443
[Stage 16:>                                                         (0 + 4) /
ration: 91, Loss: 0.894650
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 92, Loss: 1.335120
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 91, Loss: 0.692730
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 91, Loss: 1.139917
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 92, Loss: 1.000871
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 93, Loss: 1.236592
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 92, Loss: 0.685727
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 92, Loss: 0.841245
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 93, Loss: 0.952797
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 94, Loss: 1.342754
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 93, Loss: 0.686798
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 93, Loss: 1.385706
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 94, Loss: 1.059915
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 95, Loss: 1.179318
```

basic features now publicly available on
https://cloud-computing-final-6bc36.firebaseapp.com/



tech

時間	09:35	09:36	09:37	09:38	09:39	09:40	09:41
實際	漲	漲	漲	跌	漲	漲	--
預測	漲	漲	漲	漲	漲	漲	漲

for more features donate $$ to
r06921105@ntu.edu.tw or r07943149@ntu.edu.tw