# Python 实现 KDJ 的超买和超卖信息提示

KDJ 指标又叫随机指标，是一种相当新颖、实用的技术分析指标。它起先用于期货市场的分析，后被广泛用于股市的中短期趋势分析，是期货和股票市场上最常用的技术分析工具。随机指标 KDJ 一般是用于股票分析的统计体系，根据统计学原理，通过一个特定的周期（常为 9 日、9 周等）内出现过的最高价、最低价及最后一个计算周期的收盘价及这三者之间的比例关系，来计算最后一个计算周期的未成熟随机值 RSV，然后根据平滑移动平均线的方法来计算 K 值、D 值与 J 值，并绘成曲线图来研判股票走势。

KDJ 的计算比较复杂，首先要计算周期（n 日、n 周等）的 RSV 值，即未成熟随机指标值，然后再计算 K 值、D 值、J 值等。以 n 日 KDJ 数值的计算为例：

（1）n 日 RSV=（Cn－Ln）/（Hn－Ln）×100

公式中，Cn 为第 n 日收盘价；Ln 为 n 日内的最低价；Hn 为 n 日内的最高价。

（2）其次，计算 K 值与 D 值：

当日 K 值=2/3×前一日 K 值+1/3×当日 RSV

当日 D 值=2/3×前一日 D 值+1/3×当日 K 值

若无前一日 K 值与 D 值，则可分别用 50 来代替。

（3）J 值=3*当日 K 值-2*当日 D 值

KDJ 的基本使用方法：

K 线是快速确认线——数值在 90 以上为超买，数值在 10 以下为超卖；

D 线是慢速主干线——数值在 80 以上为超买，数值在 20 以下为超卖；

J 线为方向敏感线，当 J 值大于 100，特别是连续 5 天以上，股价至少会形成短期头部，反之 J 值小于 0 时，特别是连续数天以上，股价至少会形成短期底部。

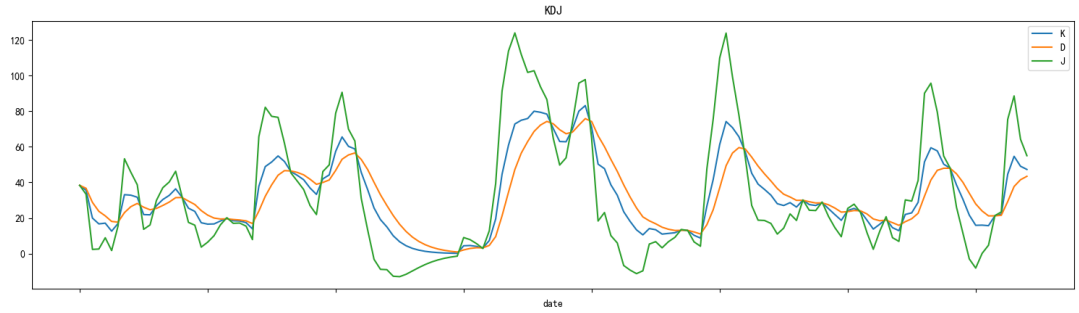
接下来，我将通过 python 调用 baostock（baostock 是免费证券数据的 python 接口，具体信息参考：www.baostock.com）的日 K 线数据实现 KDJ 的金叉死叉信息提示。

```
import baostock as bs
import pandas as pd
import matplotlib.pyplot as plt


def computeKDJ(code, startdate, enddate):
    login_result = bs.login(user_id='anonymous', password='123456')
    print(login_result.error_msg)

    # 获取股票日K线数据
    rs = bs.query_history_k_data(code,
                                 "date,code,high,close,low,tradeStatus",
                                 start_date=startdate, end_date=enddate,
                                 frequency="d", adjustflag="3")
    # 打印结果集
    result_list = []
```

```
    while (rs.error_code == '0') & rs.next():
        # 获取一条记录，将记录合并在一起
        result_list.append(rs.get_row_data())
    df_init = pd.DataFrame(result_list, columns=rs.fields)
    # 剔除停盘数据
    df_status = df_init[df_init['tradeStatus'] == '1']

    low = df_status['low'].astype(float)
    del df_status['low']
    df_status.insert(0, 'low', low)
    high = df_status['high'].astype(float)
    del df_status['high']
    df_status.insert(0, 'high', high)
    close = df_status['close'].astype(float)
    del df_status['close']
    df_status.insert(0, 'close', close)

    # 计算KDJ指标,前9个数据为空
    low_list = df_status['low'].rolling(window=9).min()
    high_list = df_status['high'].rolling(window=9).max()

    rsv = (df_status['close'] - low_list) / (high_list - low_list) *
100
    df_data = pd.DataFrame()
    df_data['K'] = rsv.ewm(com=2).mean()
    df_data['D'] = df_data['K'].ewm(com=2).mean()
    df_data['J'] = 3 * df_data['K'] - 2 * df_data['D']
    df_data.index = df_status['date'].values
    df_data.index.name = 'date'
    # 删除空数据
    df_data = df_data.dropna()
    # 计算KDJ指标金叉、死叉情况
    df_data['KDJ_金叉死叉'] = ''
    kdj_position = df_data['K'] > df_data['D']
    df_data.loc[kdj_position[(kdj_position == True) &
(kdj_position.shift() == False)].index, 'KDJ_金叉死叉'] = '金叉'
    df_data.loc[kdj_position[(kdj_position == False) &
(kdj_position.shift() == True)].index, 'KDJ_金叉死叉'] = '死叉'

    df_data.plot(title='KDJ')

    plt.show()
    bs.logout()
    return(df_data)
```

```
if __name__ == '__main__':
    code = 'sz.300104'
    startdate = '2017-01-01'
    enddate = '2018-07-01'
    df = computeKDJ(code, startdate, enddate)
    # 保存到文件中
    df.to_csv("D:/KDJ.csv", encoding='gbk')
```

KDJ 的折线图如下：




文件的结果内容如下：

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | date | K | D | J | KDJ_金叉死叉 | |
| 2 | 2017/2/3 | 38.36565 | 38.36565 | 38.36565 | | |
| 3 | 2017/2/6 | 35.5819 | 36.6954 | 33.3549 | | |
| 4 | 2017/2/7 | 19.98568 | 28.78027 | 2.396501 | | |
| 5 | 2017/2/8 | 16.71663 | 23.76922 | 2.61144 | | |
| 6 | 2017/2/9 | 17.12399 | 21.2182 | 8.935551 | | |
| 7 | 2017/2/10 | 12.63646 | 18.08232 | 1.744741 | | |
| 8 | 2017/2/13 | 16.96891 | 17.68811 | 15.5305 | | |
| 9 | 2017/2/14 | 33.11208 | 23.03818 | 53.25986 | 金叉 | |
| 10 | 2017/2/15 | 32.82755 | 26.38845 | 45.70574 | | |
| 11 | 2017/2/16 | 31.68987 | 28.18678 | 38.69604 | | |
| 12 | 2017/2/17 | 21.94964 | 26.08342 | 13.6821 | 死叉 | |
| 13 | 2017/2/20 | 21.80498 | 24.64619 | 16.12254 | | |
| 14 | 2017/2/21 | 26.90463 | 25.40289 | 29.9081 | 金叉 | |
| 15 | 2017/2/22 | 30.39563 | 27.07286 | 37.04118 | | |
| 16 | 2017/2/23 | 32.71631 | 28.95832 | 40.23231 | | |
| 17 | 2017/2/24 | 36.3734 | 31.43378 | 46.25265 | | |
| 18 | 2017/2/27 | 31.68755 | 31.51846 | 32.02573 | | |
| 19 | 2017/2/28 | 25.53263 | 29.52183 | 17.55423 | 死叉 | |
| 20 | 2017/3/1 | 23.72028 | 27.58711 | 15.98663 | | |
| 21 | 2017/3/2 | 17.34535 | 24.17216 | 3.69172 | | |
| 22 | 2017/3/3 | 16.57766 | 21.64015 | 6.452665 | | |
| 23 | 2017/3/6 | 16.69281 | 19.99082 | 10.0968 | | |

有了如上结果，但是我们还是需要注意如下几点：

1. 计算 KDJ，结果的前 9 个值为 NAN；但是程序中我已经删除了该空值数据。
2. 输入的数据量尽量足够大（推荐包含追溯你真实计算 KDJ 的前 2 个月数据）例如，你需要计算 2017-04-01 到 2017-05-01 的 KDJ，那么你需要输入的数据日期应该是 2017-02-01 到 2017-05-01；
3. 计算出来的 K，D，J 等方法和行情软件的计算方法是一样的。但是行情软件提供的 KDJ 一般是指股票上市以来计算的，若你抽取一段时间来计算，数值上会有少量差别的，但是不影响 KDJ 的基本判断；
4. KDJ 的金叉和死叉提示日期可能在真实金叉和死叉日期之后，也就是可能会延后一日。