機械学習・データマイニングの入門

Introduction to “Machine Learning” and “Data Mining”

芝浦工業大学　無線研究部
Shibaura Institute of Technology, Ham radio club

1. 動機

近年，機械学習が注目を集めており，”AI”，”人工知能”，”ディープラーニング”などと呼ばれ様々な分野からその活用法について関心が高まっている．

既に多くの文献が存在し，ソフトウェアについても，”TensorFlow “，”PySpark”，”scikit-learn”，”Keras”など多くのライブラリが存在している．

私の配属された研究室は，画像処理関連であるがイメージセンサから得られた情報を処理するために機械学習を活用するケースがある．また，車載関係を扱っている会社のインターンシップに参加した際も，カメラやレーダーなどで状況を識別するような安全支援装備を動作させるためのエッジコンピュータ開発や，防犯カメラから得られた情報を取得し不審者を識別するためのシステムのために，機械学習を実装したコーディングが進められているようである．

このように様々な方面から注目されている機械学習に触れて，実際に自分でコードを書き，コンピュータから何か結果を得てみたいと考え，本研究に取り組むこととした．

2. 目的

本研究の目的は，実際に先ほど述べたライブラリや何らかのプログラミング言語を用いて機械学習に関するコードを書き，何かしらの結果をコンピュータから得ること．そしてその得られた結果に関して，その合理性について評価・検討をする．

今回はその題材として，過去の株価の変動をコンピュータに学習させ，未来の株価についての結果を出力させることに取り組むこととした．

競馬やボートレースなど様々な分野でも活用が期待できると思うが，今回は銘柄ごとの過去の株価が容易に入手でき，また平日 9:00~11:30（前場），12:30~15:00（後場）で確実に取引が行われその変動が手軽に取得できる株式に注目した．

これは，学習時に必要である膨大な統計データを入手する上で有利であるものであると考えたからである．

研究は主に図 1 に示すスケジュールに沿って進めていくこととした．

投資家は，”テクニカル分析”と”ファンダメンタル分析”の主に 2 つの分析をして投資判断をするようである．テクニカル分析は価格や売買高などの需給や投資家の行動パターンに注目するもの，ファンダメンタル分析は，売上高や利益などの業績，財務状況など企業の本質的な価値に注目するものである．

本研究では，テクニカル分析を機械に行わせることが目的である．

4 月より研究前の事前調査を開始し，5 月より，学習用の標本データを取得するために，Python を用いたテキストマイニングを行うプログラムの作成に取り組んできた．このコードは既に完成し，実績を得ているため本中間発表で示すものとする．また並行して，機械学習を行うためのコードを作成しているが，こちらは未完成であるため，12 月に行われる最終発表に回すものとする．

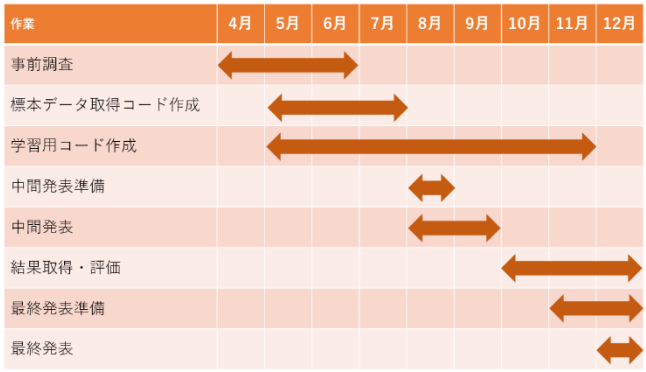

図 1 作業計画

3. 製作

機械学習を用いた分析は様々な手段が考えられる．例えば Twitter や Facebook などの SNS 上の投稿をテキストマイニングすることで，投稿の内容を見て株価を予測することもできるかもしれないし，ニュースなどと関連付けて予測することも可能かもしれない．そのような処理を組み合わせることでより精度を高めることも可能となるかもしれないが，今回は，一番手軽に取り組むことができると考えられる，過去の株価をインターネット上よりスクレイピングし，そのデータを活用して未来の株価を予測する方針で取り組んでいくこととする．

過去の株価は，

株式投資メモ　(https://kabuoji3.com/stock/)

から取得することとした．

理由として，まず無料で利用できるからである．このようなデータは本研究で行っているように分析することで利益を出すことが可能となるようなものであるので，他サイトや証券会社サイトでは有料コンテンツとして配信されている．このサイトは有難いことに無料で利用できるので使わせていただくこととする．

次に，スクレイピングが許可されていることである．

スクレイピングの許可はサイトの robots.txt で確認することができる．サイトの robots.txt (https://kabuoji3.com/robots.txt)の内容は以下の通りである．

```
User-Agent: *
Allow:/
Sitemap:http://kabuoji3.com/sitemap/sitemap-index.xml
```

このように，Allow:/ となっているため，root からスクレイピングが許可されていることがわかる．

スクレイピングは膨大な情報量をサーバから取得する行為であり，サーバサイドにとても負荷がかかる．厳密に法律で規制されているわけではないが，図書館の蔵書検索システムのデータベースをスクレイピングするために執拗にリクエストを送り付けたものとして偽計業務妨害の容疑で逮捕された経歴もある．

スクレイピングが許可されていることを確認すると同時に，許可されていてもある程度時間を空けてリクエストを送るなどの配慮は必要となる．

このサイトでは，次の URL でリクエストを送ることで株価データを得ることができる．

https://kabuoji3.com/stock/”銘柄コード”/”年度”

これでリクエストを送ると，情報が返ってくるが，返ってきた情報の中から必要な情報を分析する必要がある．

今回のサイトでは，分析の結果 tr タグ内の td に日付，始値，高値，安値，終値，出来高の順で格納されていることが分かった．よって，これらの体裁を整えることで，データを取得することが可能であることがわかる．下に一部抜粋したデータを示す．

```
<table class="stock_table stock_data_table">
<thead>
<tr>
<th>日付</th>
<th>始値</th>
<th>高値</th>
<th>安値</th>
<th>終値</th>
<th>出来高</th>
<th>終値調整</th>
</tr>
</thead>
<tbody>
    <tr>
        <td>2020-01-06</td>
        <td>2445</td>
        <td>2460</td>
        <td>2426</td>
        <td>2450</td>
        <td>1643600</td>
        <td>2450</td>
    </tr>
</tbody>
<tbody>
    <tr>
        <td>2020-01-07</td>
        <td>2469</td>
        <td>2506</td>
        <td>2447</td>
        <td>2504</td>
        <td>1122200</td>
        <td>2504</td>
    </tr>
</tbody>
```

これらを処理できるライブラリを用いて python で作成したコードが以下の通りである．

```
# -*- coding: utf-8 -*-

#各種ライブラリを宣言
from bs4 import BeautifulSoup
import pandas as pd
import requests
from datetime import datetime
import sys

#コマンドライン引数の設定
args = sys.argv

#株価を取得する
def get_dfs(stock_number):
    dfs = []
    year = [] #取得したい年を指定
    for y in year:
        try: #存在しない年は例外処理
            print(y)
            url = 'https://kabuoji3.com/stock/{}/{}/'.format(stock_number,y)
            #print(url)
            headers = {}#ユーザーエージョントの入力
            soup = BeautifulSoup(requests.get(url, headers = headers).content, 'html.parser')
            #print(soup)
            tag_tr = soup.find_all('tr')
            head = [h.text for h in tag_tr[0].find_all('th')]
            data = []
            for i in range(1,len(tag_tr)):#データをタグから取得
                data.append([d.text for d in tag_tr[i].find_all('td')])
            df = pd.DataFrame(data, columns = head)

            col = ['始値','高値','安値','終値','出来高','終値調整']
            for c in col:
                df[c] = df[c].astype(float)
            df['日付'] = [datetime.strptime(i,'%Y-%m-%d') for i in df['日付']]
            dfs.append(df)
        except IndexError:
            print('No data')
```

```
        return dfs

    #体裁を整える
    def concatenate(dfs):
        data = pd.concat(dfs,axis=0)
        data = data.reset_index(drop=True)
        col = ['始値','高値','安値','終値','出来高','終値調
整']
        for c in col:
            data[c] = data[c].astype(float)
        return data

    #作成したコードリストから必要銘柄を読み込む
    code_list = pd.read_csv(args[1])

    #複数のデータフレームを CSV で保存
    for i in range(len(code_list)):
        k = code_list.loc[i,'code']
        v = code_list.loc[i,'name']
        print(k,v)
        dfs = get_dfs(k)
        data = concatenate(dfs)
      data.to_csv('{}-{}.csv'.format(k,v))
```

フローチャートは以下の通りである.

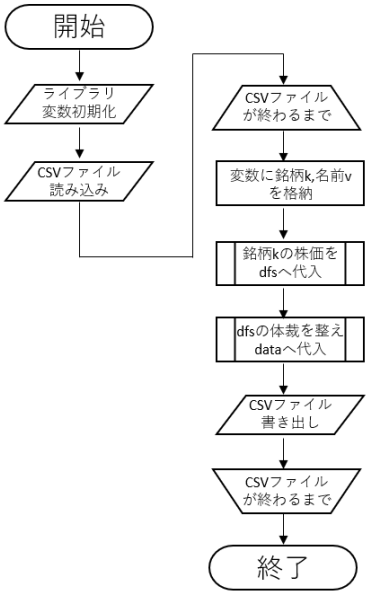

図 2 メイン関数のフローチャート

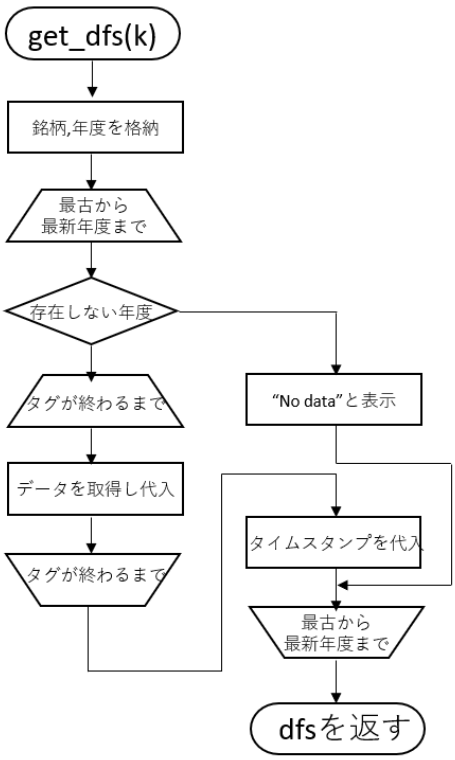

図 3 get_dfs(k)のフローチャート

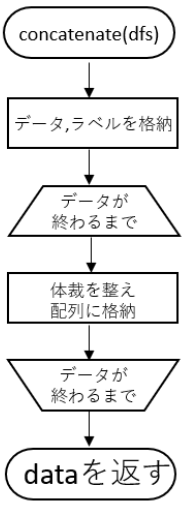

図 4 concatenate(dfs)のフローチャート

今回取得する銘柄としては，7 月 6 日時点での日経平均株価と日経 500 平均の二つの指標に用いられている銘柄，計 502 銘柄である．

取得したデータは CSV で保存されるものとしている．

以下のような内容で得られる．いずれもサイト上で入手できる最古のデータから最新（7 月 6 日）までのデータが入っている．

製作に使用した環境は以下の通りである．

```
OS : Debian GNU/Linux 10.4

CPU : Intel(R) CPU Atom N270     @ 1.60GHz
MEMORY : DDR2 2GB
HDD : 300GB

Python : 3.7.3
bs4 : 0.0.1
pandas : 0.24.2
requests : 2.23.0
DateTime : 4.3
```

4. 結果

実際に計 502 銘柄を取得した結果が以下のとおりである．いずれもサイト上で入手できる最古のデータから最新（7 月 6 日）までのデータが入っている．

以下に取得したデータを一部抜粋したものである．1332-日水のデータである．1983 年 1 月 4 日がサイト上の最古のデータであり，それぞれ取得した値が入っていることがわかる．

```
,日付,始値,高値,安値,終値,出来高,終値調整
0,1983-01-04,266.0,266.0,263.0,266.0,214000.0,246.3
1,1983-01-05,266.0,269.0,264.0,267.0,921000.0,247.2
2,1983-01-06,270.0,274.0,267.0,270.0,2481000.0,250.0
3,1983-01-07,271.0,272.0,267.0,270.0,1240000.0,250.0
4,1983-01-08,270.0,270.0,266.0,268.0,508000.0,248.2
5,1983-01-10,266.0,268.0,265.0,265.0,252000.0,245.4
6,1983-01-11,264.0,265.0,261.0,265.0,233000.0,245.4
7,1983-01-12,263.0,264.0,261.0,263.0,339000.0,243.5
8,1983-01-13,263.0,264.0,261.0,263.0,263000.0,243.5
9,1983-01-14,268.0,269.0,265.0,268.0,649000.0,248.2
```

5. 考察

今回の中間発表では，5 月から 7 月にかけて取り組んできた標本用データの取得法として，テキストマイニングのためのスクレイピングについて取り組んだ．

CSV 形式で取得したい銘柄を入力しておき，そのファイルをコマンドラインより引数で指定することにより，取得したい銘柄の一番古いデータから最近のデータまでを自動で取得するようなコードを作成することに成功した．

使用したコンピュータはそれほどスペックの高くないもので，32bit で Debian が動作しているものを SSH サーバとしてネットワーク上で接続しターミナル上から操作したが，インターネット上から情報を取得し保存するという単純な作業であったため問題なくデータを取得し保存することができた．

6. まとめ・展望

今回の中間発表までの研究で, 機械学習を行うための標本となるデータを取得することに成功したため，今後は取得したデータを用いて，実際に機械学習のアルゴリズムを使い学習させていく段階へと移行しようと計画している．

最終発表までに, 機械学習のコードを実装する作業を終わらせ，結果を評価する段階まで進めればよいと思う．

7. 参考文献

1. Yuxi(Hayden) Liu, 2020, Python 機械学習, 朝倉書店

2. 巣籠 悠輔, 2019, 詳解 ディープラーニング [第 2 版] TensorFlow/Keras・PyTorch による時系列データ処理, マイナビ出版

3. 中井 悦司, 2019, TensorFlow と Keras で動かしながら学ぶディープラーニングの仕組み：畳み込みニューラルネットワーク徹底解説, マイナビ出版

4. 株式投資メモ, https://kabuoji3.com/, 2020/09/03 アクセス