WORLDQUANT UNIVERSITY

FINANCIAL ENGINEERING

GROUP WORK PROJECT

DATA PREPARATION AND ANALYSIS

Group members:

Name	Email
Quyen Ho Thanh	thquyen11@hotmail.com
Wei Hao Lew	lewweihao93@hotmail.com
Lu Liu	liulu0502@gmail.com
Truong Nguyen	nmtruong93@gmail.com
Yernur Orakbayev	yernur.orakbayev@nu.edu.kz

May 2020

Contents

1. Background

The aim of this report is to review the theory and apply the following techniques using Python

- Calculate mean and standard deviation
- Implement technical indicators (moving average, EWMA)
- Identify structural breaks using Chow test
- Apply Bera-Jarque test for returns
- Apply a cointegration test using Johansen test
- Forecast next period (t+1) asset return evolution using AR(1) model
- Provide line charts for the forecast and the selected asset

A time series is a sequence of observations taken sequentially in time. Time series modeling applied in a wide range of fields from economics, business, chemical…especial in finance. The nature of time series is the dependences between observations which mean using previous observations to forecast the future itself. From this viewpoint, we consider time series analysis as a subset of supervised learning (a machine learning subfield) when using the present observation as a label of previous one.

In this report, we show that a comprehensive of step by step how to apply time series techniques from data collecting, feature engineering to forecasting.

2. Method

2.1. Moving average and exponential weighted moving average (EWMA)

Moving Average (MA) is an indicator that is used to represent the average closing price of the market over a specified period of time. It is one of the oldest tools used by technical analysts, dating back to 1901 with the work of mathematician R. H. Hooker. It entered circulation through W. I. King's Elements of Statistical Method (1912). "Moving average" refers to a type of stochastic process is an abbreviation of H. Wold's "process of moving average" (A Study in the Analysis of Stationary Time Series (1938)).

A simple moving average is formed by computing the average (mean) price of a security over a specified number of periods. While it is possible to create moving averages from the Open, the High, and the Low data points, most moving averages are created using the closing price.

The formula for simple moving average at any point in time can be derived simply calculating the average of a certain number of periods up to that point in time. For instance, the 20-day moving average of stock price means the average of the stock price of the last 20 days. The

averages are then joined to form a smooth curving line - the moving average line. The next closing price for this new period would be added and the oldest day would be dropped. Mathematically, it is represented as,

$$Moving\ Average = (A_1 + A_2 + \cdots + A_n)\ /n$$

where A_i is the data point in the i_{th} period

In order to reduce the lag in simple moving averages, traders often use exponential moving averages (also called exponentially weighted moving averages) to reduce the lag by applying more weight to recent prices relative to older prices. Roberts (1959) introduced the exponentially weighted moving average (EWMA) control scheme and showed that the EWMA is useful for detecting small shifts in the mean of a process. P. N. (Pete) Haurlan was the first to use exponential smoothing for tracking stock prices in the early 1960s. He did not call them "exponential moving averages (EMAs)", or "exponentially weighted moving averages (EWMAs)". Instead he called them "Trend Values", and referred to them by their smoothing constants. The formula of EWMA applies more weight to recent prices relative to past prices.

The formula for EWMA is:

$$EWMA(t) = aY(t) + (1 - a)EWMA(t - 1)$$

where

- Y (t) is the stock price today
- EWMA (t-1) is the estimated value at t-1 time
- **a** (0 < **a** <1) represents the weight coefficient for historical measured values
- and **a** = 2 / (n + 1).

The closer the coefficient of **a** is to 1, the higher the weight of the current sampling value and and the lower the weight of the past measurement value will be given to the calculation. The weighting applied to the most recent price depends on the specified period of the moving average. The shorter the EMA's period, the more weight that will be applied to the most recent price.

The decision of choosing between moving average and EWMA depends on the investment style and preferences. Moving averages are lagging indicators and fits in the category of trend following indicators. Moving averages is an effective tool to identify and confirm trend, identify support and resistance levels, and develop trading systems. However, traders should identify the scenarios that are suitable for analysis with moving averages since moving averages can give misleading signals if stock prices are not trending. EWMA can capture the changes and sensitivity quicker.

2.2. Structural breaks

A lot of techniques used for detecting the structural breaks such at Quandt Likelihood ratio, CUSUM, Chow tests… For sake of simplicity, we only use Chow test.

The Chow Test

A series of data can often contain a structural break due to political and economic factors, i.e. stock market crash. Numerous studies have been undertaken on the issue of structural changes. This is evidenced by the two special volumes of this Journal edited, respectively, by Broemeling (1982), and Dufour and Ghysels (1996), as well as by a number of monographs on this subject, e.g., Poirier (1979), Kramer (1989), and Hackl and Westlund (1991). The Chow test was developed by econometrician Gregory Chow in 1960 to test for structural breaks and is a method well known in econometrics. It was originally designed to analyze whether one regression or more regressions best fit the time series data. The Chow test is an application of the F-test and the analysts usually use the Chow test to determine whether a single regression is more efficient than two separate regressions involving splitting the data into two sub-samples. The null hypothesis of Chow test is that there is no structural break. The formula in running the Chow test is:

$$F\ test = \frac{(RSS_C - (RSS_1 + RSS_2))/k}{(RSS_1 + RSS_2)/(n-2k)}$$

Where RSS_c runs the regression using all the data, before and after the structural break; RSS_1 and RSS_2 runs two separate regressions on the data before and after the structural break; This test has k and n-2k degrees of freedom with a known potential break point.

2.3. Jarque-Bera Test

The Jarque-Bera test is a type of Lagrange multiplier test which is used to test for normality - a normal distribution has a skew of zero and a kurtosis of three. Jarque-Bera Test is usually used for large data sets, because other normality tests are not reliable when n is large. Specifically, the test matches the skewness and kurtosis of data to see if it matches a normal distribution.

- The null hypothesis for the test is that the data is normally distributed
- The alternate hypothesis is that the data does not come from a normal distribution.

$$JB = \frac{N}{6}\left(W^2 + \frac{(K-3)^2}{4}\right)$$

where:

- N is the sample size,
- W is the sample skewness coefficient,

- K is the kurtosis coefficient.

2.4. Cointegration test

The two non-stationary time series X and Y are cointegrated if there exist the constants a, b in which the combined series aX + bY is stationary. In this report, we use the common method called Johansen test.

Johansen Test

The Johansen test is more flexible than the CADF which can check for multiple linear combinations of time series for forming stationary portfolios. The null hypothesis means that there is no cointegration at all.

The theoretical details of the Johansen test consider the Vector Autoregressive Models (VAR), the general form of VAR(p) model without drift is:

$$x_t = \mu + A_1 x_{t-1} + \ldots + A_p x_{t-p} + w_t$$

where:

- μ is the vector-valued mean of the series
- A_i are the coefficient matrices for each lag
- w_t is a multivariate Gaussian noise term with mean zero.

Then we form a Vector Error Correction Model (VECM) by differencing the series:

$$\Delta x_t = \mu + A x_{t-1} + \beta_1 \Delta x_{t-1} + \ldots + \beta_p \Delta x_{t-p} + w_t$$

where:

- $\Delta x_t = x_t - x_{t-1}$
- A is the coefficient matrix for the first lag
- β_i are the matrices for each differenced lag

Next, we perform the eigenvalue decomposition on matrix A in order to find the rank r. The null hypothesis of no cointegration occurs when the matrix A = 0 or r = 0. The rank r > 0 implies a cointegrating relationship between two or more time series.

2.5. Autoregression model

Autoregression model is a technique used for forecasting future values of the time series data that assumes linear combination between observations. The more distant time, the weight of the observations less affect current value. The number of previous observations greatly affects the present value called lags, denoted **p**. Thus, an autoregressive model of order **p** can be written as

$$y_t = c + \phi_1 y_1 + \phi_2 y_2 + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

where $\varepsilon_t$is white noise. We refer to this as an AR(p) model, an autoregressive model of order **p** (Hyndman et al, 2018). To determine number of lag **p**, we use autocorrelation function (ACF). Autocorrelation function measures the linear relationship between lagged values of a time series. There are several autocorrelation coefficients, corresponding to each panel in the lag plot. For example, r_1 measures the relationship between y_t and y_{t-1}, r_2 measures the relationship between y_t and y_{t-2}, and so on.

The value of r_k can be written as

$$r_k = \frac{\sum_{t=k+1}^{T}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T}(y_t - \bar{y})^2}$$

where T is the length of the time series.

For an AR(1) model:

- When $\phi_1 = 0$, $y_1$is equivalent to white noise.
- When $\phi_1 = 1$ and c = 0, $y_1$is equivalent to random walk.
- When $\phi_1 = 1$ and $c \neq 0$, $y_1$is equivalent to a random walk with drift.
- When $\phi_1 < 0$, $y_1$tends to oscillate around mean.

We normally restrict autoregressive models to stationary data, in which case some constraints on the values of the parameters are required.

For an AR(1) model: $-1 < \phi_1 < 1$

3. Results

Import vital libraries and download NASDAG index data during 6 years from 2014-05-08 to 2020-05-08.

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import yfinance as yf
import statsmodels.tsa.vector_ar.vecm as stvv
from statsmodels.tsa.ar_model import AutoReg
from sklearn.metrics import mean_squared_error
from datetime import datetime as dt
import statsmodels.api as sm
from scipy import stats
import seaborn as sns
import warnings
warnings.filterwarnings('ignore', 'statsmodels.tsa.ar_model.AR', FutureWarning)
```

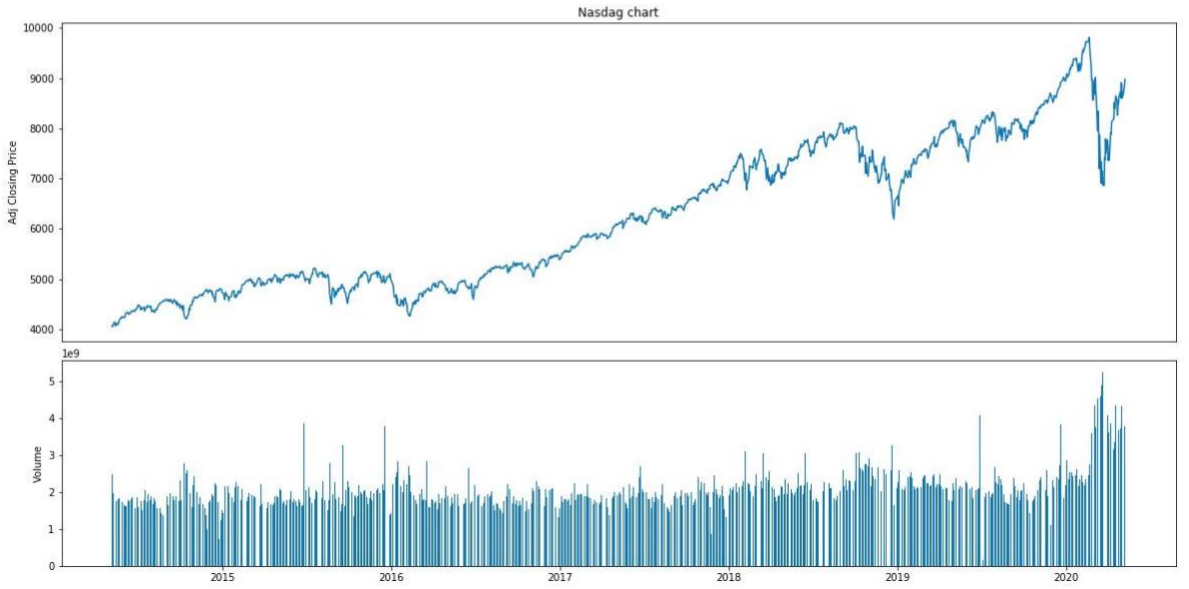
[2]:
```
start_date = '2014-05-08'
end_date = '2020-05-08'

nasdaq = yf.download('^IXIC', start=start_date, end=end_date, progress=False).rename(columns={'Adj Close': 'Adj_Close'})
nasdaq.sort_index(inplace=True)
nasdaq.head()
```

[2]:

Date	Open	High	Low	Close	Adj_Close	Volume
2014-05-07	4085.489990	4091.030029	4021.050049	4067.669922	4067.669922	2486030000
2014-05-08	4053.280029	4109.200195	4039.909912	4051.500000	4051.500000	2411940000
2014-05-09	4043.439941	4071.870117	4025.239990	4071.870117	4071.870117	1976160000
2014-05-12	4092.840088	4146.540039	4092.090088	4143.859863	4143.859863	1880020000
2014-05-13	4144.899902	4155.129883	4128.009766	4130.169922	4130.169922	1923480000

Plot close price and volume. Price tends to increase in long term and fluctuate in short term.

[3]:
```
plt.figure(figsize=(20,10))
top = plt.subplot2grid((5,4), (0, 0), rowspan=3, colspan=4)
bottom = plt.subplot2grid((5,4), (3,0), rowspan=2, colspan=4)
top.plot(nasdaq.index, nasdaq.Adj_Close)
bottom.bar(nasdaq.index, nasdaq.Volume)

# set the labels
top.axes.get_xaxis().set_visible(False)
top.set_title('Nasdaq chart')
top.set_ylabel('Adj Closing Price')
bottom.set_ylabel('Volume')
plt.show()
```



Mean and standard deviation

Mean of Nasdag close price during 6 years equals to 6241.9 and standard deviation is 1443.72. They measure the central of the data and how data spread out, respectively.

[4]:
```
close_price_mean = nasdaq.Close.mean()
close_price_std= nasdaq.Close.std()
(close_price_mean, close_price_std)
```

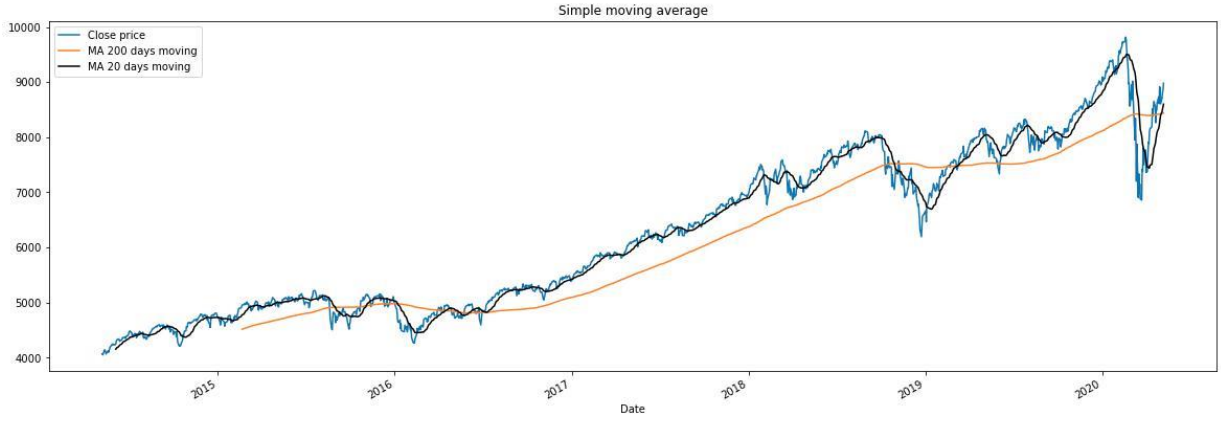
[4]:
```
(6241.90063298947, 1443.7182617447202)
```

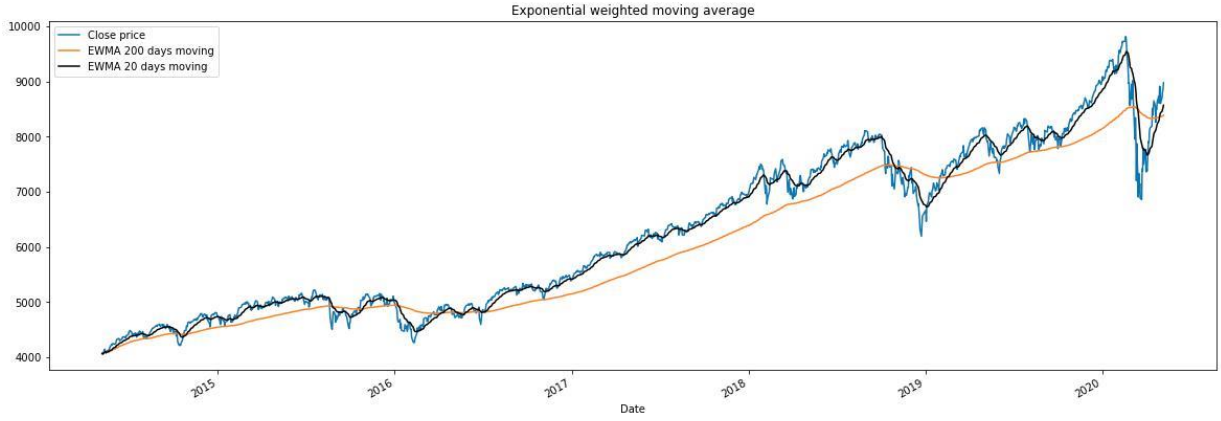
Technical indicators (Moving average, EWMA)

Some of popular moving average is implement (14, 20, 50, 100, 200 days). For convenient, we consider 20 days as short-term and 200 days as long-term indicators.

[5]:
```
# Moving average
nasdaq['ma_14'] = nasdaq.Close.rolling(window=14).mean()
nasdaq['ma_20'] = nasdaq.Close.rolling(window=20).mean()
nasdaq['ma_50'] = nasdaq.Close.rolling(window=50).mean()
nasdaq['ma_100'] = nasdaq.Close.rolling(window=100).mean()
nasdaq['ma_200'] = nasdaq.Close.rolling(window=200).mean()

# exponential weighted moving average
nasdaq['ewma_14'] = nasdaq.Close.ewm(span=14).mean()
nasdaq['ewma_20'] = nasdaq.Close.ewm(span=20).mean()
nasdaq['ewma_50'] = nasdaq.Close.ewm(span=50).mean()
nasdaq['ewma_100'] = nasdaq.Close.ewm(span=100).mean()
nasdaq['ewma_200'] = nasdaq.Close.ewm(span=200).mean()
```

[6]:
```
plt.figure(figsize=(20, 15))

# Simple moving average
plt.subplot(211)
nasdaq.Close.plot(label='Close price')
nasdaq.ma_200.plot(label='MA 200 days moving')
nasdaq.ma_20.plot(label='MA 20 days moving', c='black')
plt.title('Simple moving average')
plt.legend(loc='best')

# Exponential weighted moving average
plt.subplot(212)
nasdaq.Adj_Close.plot(label='Close price')
nasdaq.ewma_200.plot(label='EWMA 200 days moving')
nasdaq.ewma_20.plot(label='EWMA 20 days moving', c='black')
plt.title('Exponential weighted moving average')
plt.legend(loc='best')
plt.show()
```





Identify structural breaks: the Chow test

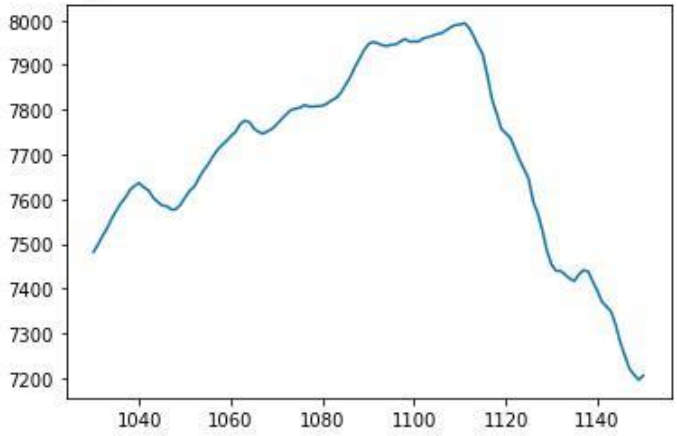
- Null hypothesis: there are no structural break
- Alternative hypothesis: there are structural break

We will test for structural break around the time of September 2018, as there was a recession at that point in time which caused a huge market dip. This corresponds to an index of 1090 in our time series. But first, we would need to do a linear regression on the time series. We will be using a window size of 60 for each splitted set, which corresponds to a time period of about 3 months, roughly the length we would expect for a given regime. Then, we perform a linear regression on each of the splitted time series. Instead of using the raw time series data, we will be using the 20-day moving average as the original series is too noisy to perform a linear regression.

Plot 3 months data that we assume having structural breaks.

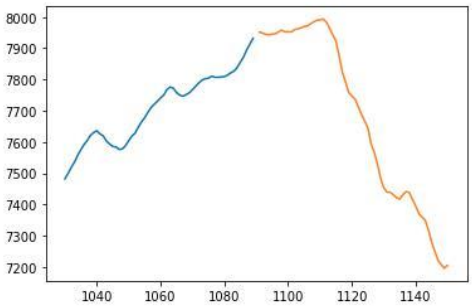
[7]:
```
breakPoint = 1090
windowSize = 60
Y_log = nasdaq.ewma_20
X = [i for i in range(0, len(Y_log))]
Y_log_trimmed = Y_log[breakPoint - windowSize: breakPoint + 1 + windowSize]
X_trimmed = X[breakPoint - windowSize: breakPoint + 1 + windowSize]
resid = sm.OLS(Y_log_trimmed, X_trimmed).fit()

print("SSR beg-end:", resid.ssr)
plt.plot(X_trimmed, Y_log_trimmed)
plt.show()
```

```
SSR beg-end: 15489466.181556512
```



Visualize the data withouth break point.

[8]:
```
Y_before = Y_log[breakPoint - windowSize:breakPoint]; X_before = X[breakPoint - windowSize:breakPoint]
Y_after = Y_log[breakPoint + 1:breakPoint + 1 + windowSize]; X_after = X[breakPoint + 1:breakPoint + 1 + windowSize]
plt.plot(X_before, Y_before)
plt.plot(X_after, Y_after)
plt.show()
```



Conduct Chow test. The p-value = 9.73e-59 is much lower than 0.01, we can safety reject the null hypothesis that there is no structural break and consider the alternative hypothesis that there is indeed a structural break in September 2018 of the time series with 99% statistical significance.

[9]:
```
resid_before = sm.OLS(Y_before, X_before).fit()
resid_after = sm.OLS(Y_after, X_after).fit()
print("SSR 2014-05-08;2018-09-05", resid_before.ssr)
print("SSR 2018-09-05;2020-05-08", resid_after.ssr)
```

```
SSR 2014-05-08;2018-09-05 68511.94302010478
SSR 2018-09-05;2020-05-08 9077394.581288755
```

[10]:
```
from scipy.stats import f

ssr_total = resid.ssr
ssr_before = resid_before.ssr
ssr_after = resid_after.ssr
numer = (ssr_total - (ssr_before + ssr_after)) / 2
denomin = (ssr_before + ssr_after) / (1510 - 2*2)
chow_test = numer / denomin
p = f.sf(chow_test, 2, len(Y_before) + len(Y_after) - 4)
print("p-value: " + str(p))
```

```
p-value: 9.726461640653348e-59
```

Bera-Jarque test for returns.

The Jarque-Bera test tests whether the sample data has the skewness and kurtosis matching a normal distribution

- Null hypothesis: the data is normally distributed
- Alternative hypothesis: the data is normally distributed.

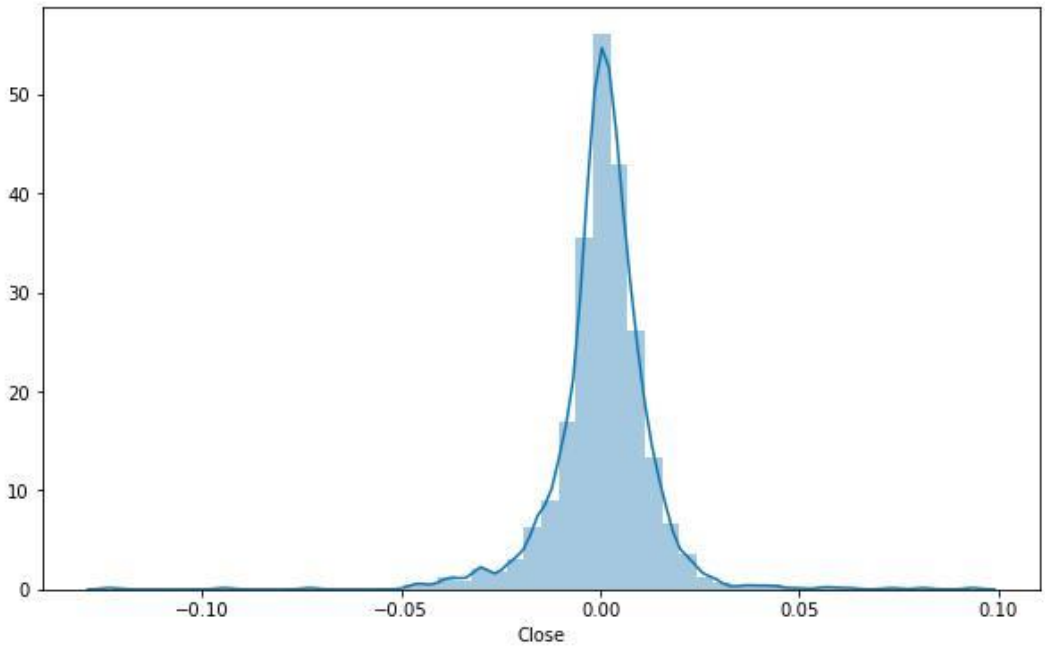
When we look at the plot of nasdaq return, we may guess that it follows the normal distribution. To assert that, the skewness, kurtosis and Bera-Jarque test.

- Skewness = -0.656 $\neq$ 0
- Kurtorsis = 14.911 $\neq$ 3
- p-value = 0.0 < 0.01, reject the null hypothesis with statistical significance 99%.

From these three result, we conclude that the data is not normal distributed.

[11]:
```
nasdaq_returns = nasdaq['Close'].pct_change()
nasdaq_returns = nasdaq_returns.dropna()
df = nasdaq_returns.reset_index()
```

[12]:
```
plt.figure(figsize=(10, 6))
sns.distplot(nasdaq_returns)
plt.show()
```



[13]:
```
print("Mean:                ", nasdaq_returns.mean())
print("Variance:            ", nasdaq_returns.var())
print("Skewness:            ", stats.skew(nasdaq_returns, bias=False))
print("Kurtosis:            ", stats.kurtosis(nasdaq_returns, bias=False))
print('The test statistic: ', stats.jarque_bera(nasdaq_returns)[0])
print('The p-value:         ', stats.jarque_bera(nasdaq_returns)[1])
```

```
Mean:                0.0006022726909184769
Variance:            0.00015523884262989713
Skewness:            -0.6552379744832914
Kurtosis:            14.910650950147442
The test statistic:  14005.453833231051
The p-value:         0.0
```

Apply cointegration test

We will be using S&P500 as an macroeconomic indicator for the US economy (in a long term, S&P500 reflect the US economic strength) to test for it's co-integration with the selected asset using Johansen.

- Null hypothesis: there is no cointegrating relationship.
- Alternative hypothesis: there is at least one cointegrating relationship.

Dowload data, calculate return and remove NaN data.

```
[14]: spy = yf.download('^GSPC', start=start_date, end=end_date, progress=False).rename(columns={'Adj Close': 'Adj_Close'})
      spy.sort_index(inplace=True)
      spy_returns = spy['Close'].pct_change()
```

```
[15]: #Clean data
      nasdaq_returns = nasdaq_returns[~np.isnan(nasdaq_returns)]
      nasdaq_returns = nasdaq_returns[~np.isinf(nasdaq_returns)]
      spy_returns = spy_returns[~np.isnan(spy_returns)]
      spy_returns = spy_returns[~np.isinf(spy_returns)]
```

The trace and max eigenvalue tests show that there is cointegration relationship between Nasdaq and S&P500 at 99% statistical significance. Hence, we should not include these both indices in a model.

```
[16]: df = pd.DataFrame({'x': nasdaq_returns, 'y': spy_returns})
      model = stvv.coint_johansen(df, 0, 1)
      colums = ['Crit-90%', 'Crit-95%', 'Crit-99%']
      index = ['r<=0', 'r<=1']
```

```
[17]: print('--------------TRACE STATISTIC--------------')
      df_trace = pd.DataFrame(data=model.cvt, columns=colums, index=index)
      df_test = pd.DataFrame(data=model.lr1, columns=['Test statistic'], index=index)
      pd.concat([df_test, df_trace], axis=1)
```

```
--------------TRACE STATISTIC--------------
```

	Test statistic	Crit-90%	Crit-95%	Crit-99%
r<=0	1227.718365	13.4294	15.4943	19.9349
r<=1	580.552333	2.7055	3.8415	6.6349

```
[18]: print('--------------TEST STATISTIC--------------')
      df_trace = pd.DataFrame(data=model.cvm, columns=colums, index=index)
      df_test = pd.DataFrame(data=model.lr2, columns=['Test statistic'], index=index)
      pd.concat([df_test, df_trace], axis=1)
```

```
--------------TEST STATISTIC--------------
```

	Test statistic	Crit-90%	Crit-95%	Crit-99%
r<=0	647.166032	12.2971	14.2639	18.5200
r<=1	580.552333	2.7055	3.8415	6.6349

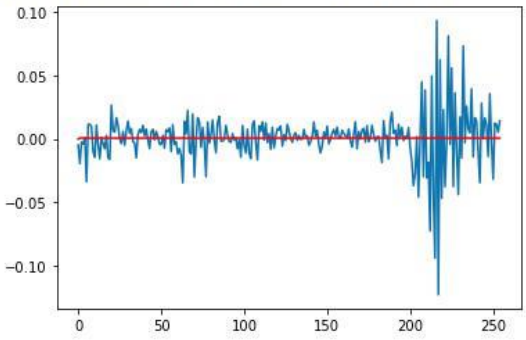
Forecast next period (t+1) asset return evolution using AR(1) model

The predicted return for the next time t+1 is - 0.000248. If this were used as a trading strategy, a short position would be best advised.

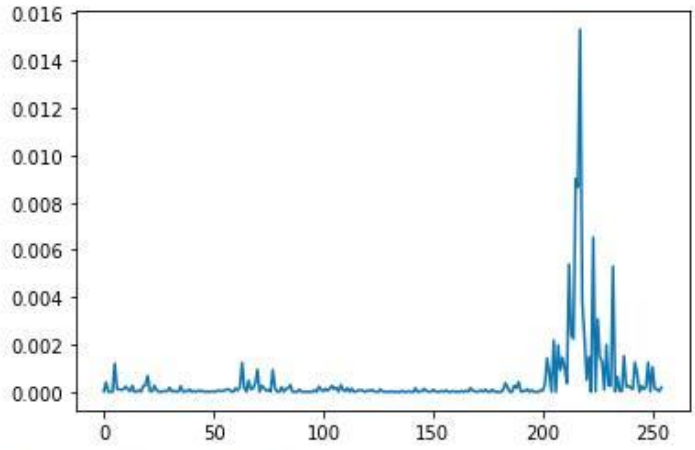
The plot below shows the forecasted period. The red line shows the mean predicted by the AR(1) model while the blue line shows the actual returns from the original time series. We can get the squared error between the prediction and actual value by plotting them.

[19]:
```
train_start ='2018-01-05'
train_end = '2019-05-05'
test_start = train_end

train, test = nasdaq_returns[train_start : train_end].values, nasdaq_returns[test_start : ].values
model = AutoReg(train, lags=1)
model_fit = model.fit()
print('Coefficients: %s' % model_fit.params)

# make predictions
predictions = model_fit.predict(start=len(train), end=len(train) + len(test) - 1, dynamic=False)
print("Prediction at t+1: " + str(predictions[0]))

plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

```
Coefficients: [ 0.00050488 -0.04757815]
Prediction at t+1: -0.00024832957713594454
```



[20]:
```
plt.plot(((predictions - test))**2)
plt.show()

print("MSE:" + str(mean_squared_error(predictions,test)))
```




```
MSE:0.0004266898942335333
```

The mean square error is quite large when data fluctuating.

Conclusion

- In this report, we presented a simple way to deal with financial time series from data preparation, feature engineering, modeling and forecasting data. In the real world, the problems may not as simple as this, but we should try simple methods first, then complex and the last is complicated.
- Moving average indicator with different window sizes is powerful tool that is indispensable in trading strategy. It is usually used as a part of ARIMA family models, denoising techniques… in algorithmic trading. Besides, this indicator is usually combined with price and volume in finding super stocks for individual investors.
- The structural breaks are strutures causing trend discontinuity and trend reversal that require us treat them wisely and carefully. The typical testing methods are the Chow, Quandt Likelihood Ratio and CUSUM tests.
- In many time series models (autoregressive, moving average, ARIMA, …), they often assume stationary in the data. The nature of stationary with constant of mean, variance and covariance help us predict the future more simple and reliable. Therefore, we have to look into the assumptions of each model before apply them. The Bera-Jarque allows us test whether the dataset have stationary property. However, nowadays, the time series modern techniques such as Long-Short Term Memory (LSTM) and its variances can treat datasets that are not stationary with high accuracy.
- Feature engineering is most important phase in data analysis. Garbage in, garbage out is always truth in data modeling. Co-integration testing is approaches helping model bias removal. However, some modern methods help us save time without co-integration testing like ensemble (Random forest, Adaboost, Gradient Boosting…), neural network…
- Autoregressive model is one of most simple and popular time series model that helps us consider dependence between adjacent observations. Between the two simple and complex methods, if they both solve the same problem well, we should choose the simpler method to save costs. Hence, any simple method should not be ignored.

References

Achelis, S.B. (2001) Technical analysis from A to Z, McGraw-Hill, New York, 2nd edition.

Andrews, D.W.K., 1993. Testing for structural instability and structural change with unknown change point. Econometrica 61, 821-856.

Atanasova, C.V. and Hudson, R.S. (2010) Technical trading rules and calendar anomalies –are they the same phenomena?, Economics Letters, 106(2), 128–130.

Bai, J., Perron, P., 1998. Estimating and testing for multiple structural changes in linear models. Econometrica 66, 47}78.

Brock, W., Lakonishok, J. and LeBaron, B. (1992) Simple technical trading rules and thestochastic properties of stock returns, Journal of Finance, 47(5), 1731–1764.

Brock, W., Lakonishok, J. and LeBaron, B. (1992) Simple technical trading rules and the stochastic properties of stock returns, Journal of Finance, 47(5), 1731–1764.

Brock, W., Lakonishok, J. and LeBaron, B. (1992) Simple technical trading rules and the stochastic properties of stock returns, Journal of Finance, 47(5), 1731–1764.

George E.P.Box, Gwilym M.Jenkins, Gregory C.Reinsel, Reta M.Ljung. (2016). Time series analysis*: Forecasting and Control* (5th ed.). Wiley.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice*. Melbourne: OTexts.

Jarque-Bera Test. (2017, December 09). Retrieved from https://www.statisticshowto.datasciencecentral.com/jarque-bera-test/

Kim, S., Cho, S. & Lee, S. (2000). On the cusum test for parameter changes in GARCH(1,1) models. Commun. Statist. Theory Methods 29, 445–462

Quandt, R. E. (1960) Tests of the hypothesis that a linear regression system obeys two separate regimes. J. Amer. Statist. Assoc. 55, 324–330.

Stock, James H., and Mark Watson. 2011. Introduction to Econometrics. 3rd ed. Boston: Pearson Education/Addison Wesley.

Tsay, R.S. (2010) 'Analysis of Financial Time Series'. Wiley