Data Science Capstone Project

Prediction of gold price by time series algorithms

Instructor: Dr.Abdi Awl

Changhao Ying, Jialei Chen

12/09/2020

Table of Contents

Problem Statement

Gold price and the trend of the dollar affect each other. Biased estimation of the gold price will lead to wrong investment decisions and cause huge economic losses. Predicting gold price helps companies formulate appropriate investment plans.

Problem Elaboration

Try different models including linear regression, simple exponential smoothing, Holt winter's seasonal method, ARMA, ARIMA, neural networks, robust prediction, LSTM and iterate model quickly to find the model with best performance.

Literature Review

SARIMA Introduction
https://www.statsmodels.org/dev/examples/notebooks/generated/statespace_sarimax_stata.html
ARIMA and SARIMA forecast principle
https://towardsdatascience.com/time-series-in-python-exponential-smoothing-and-arima-processes-2c67f2a52788
Time series textbook by R language
https://otexts.com/fpp2/advanced.html
STL decomposition Source Code
https://github.com/jrmontag/STLDecompose
Long Short Term Memory
https://doi.org/10.1162/neco.1997.9.8.1735

Tools and Packages Used

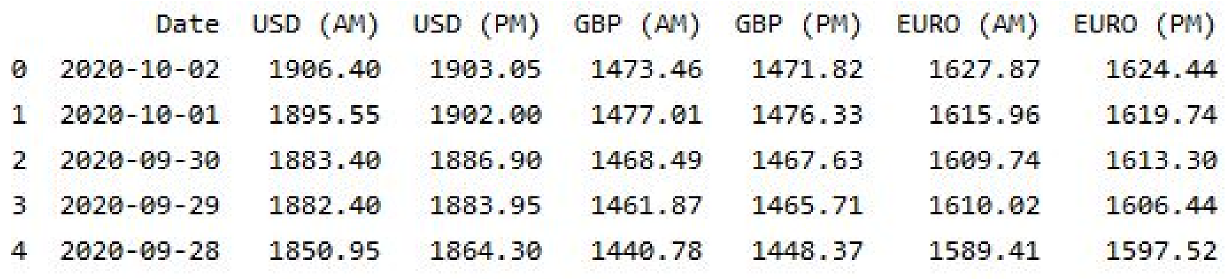
Tools: Github, Pycharm, Google Colab
Computer language: Python
Github:
https://github.com/olokojoh/Data-Science-Capstone-DATS6501
Google Colab:
https://colab.research.google.com/drive/1wDaaHMV_THbDSQ5Vel-ZvNKuX9zsqoj3?usp=sharing
https://colab.research.google.com/drive/1x6v5Ud67gKbuNgCg970PhFyBfEDROUBE?usp=sharing
Packages: statsmodels, adfuller test, STL, Keras, Sklearn
Version: Python 3, Tensorflow 2.3.0

Dataset Introduction

The dataset is about gold price per day variation by London Bullion Market Association (LBMA). Fixing levels are set per troy ounce. (https://www.quandl.com/data/LBMA/GOLD-Gold-Price-London-Fixing)

The dataset includes 13335 daily gold price data from 1968.1.2 to 2020.10.2. The Gold price in London is set twice a day by five LBMA Market Makers who comprise the London Gold Market Fixing Limited (LGMFL). The process starts with the announcement from the Chairman of the LGMFL to the other members of the LBMA Market Makers, then relayed to the dealing rooms where customers can express their interest as buyers or sellers and also the quantity they wish to trade. The gold fixing price is then set by collating bids and offers until the supply and demand are matched. At this point the price is announced as the 'Fixed' price for gold and all business is conducted on the basis of that price. The Gold price is set in USD, GBP and EURO.

The dataset includes 7 columns: dataset and prices:

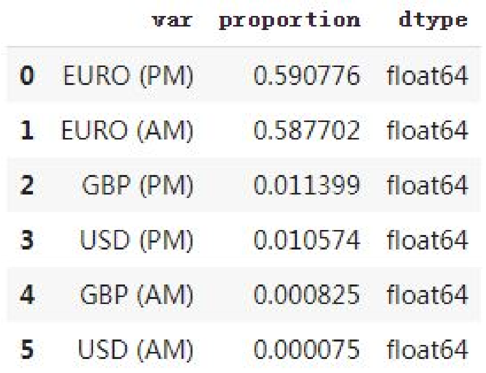


```
        Date  USD (AM)  USD (PM)  GBP (AM)  GBP (PM)  EURO (AM)  EURO (PM)
0  2020-10-02   1906.40   1903.05   1473.46   1471.82    1627.87    1624.44
1  2020-10-01   1895.55   1902.00   1477.01   1476.33    1615.96    1619.74
2  2020-09-30   1883.40   1886.90   1468.49   1467.63    1609.74    1613.30
3  2020-09-29   1882.40   1883.95   1461.87   1465.71    1610.02    1606.44
4  2020-09-28   1850.95   1864.30   1440.78   1448.37    1589.41    1597.52
```

EDA & Visualization

We firstly checked the missing values in the dataset with this function:

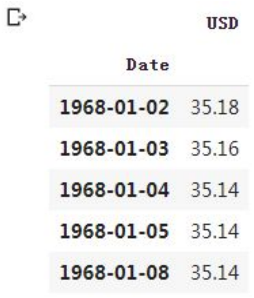
```
def nan_checker(df):
    df_nan = pd.DataFrame([[var, df[var].isna().sum() / df.shape[0],
df[var].dtype]
                           for var in df.columns if df[var].isna().sum() > 0],
                          columns=['var', 'proportion', 'dtype'])
    df_nan = df_nan.sort_values(by='proportion', ascending=False)
    return df_nan
df_nan = nan_checker(df)
df_nan.reset_index(drop=True)
```



	var	proportion	dtype
0	EURO (PM)	0.590776	float64
1	EURO (AM)	0.587702	float64
2	GBP (PM)	0.011399	float64
3	USD (PM)	0.010574	float64
4	GBP (AM)	0.000825	float64
5	USD (AM)	0.000075	float64

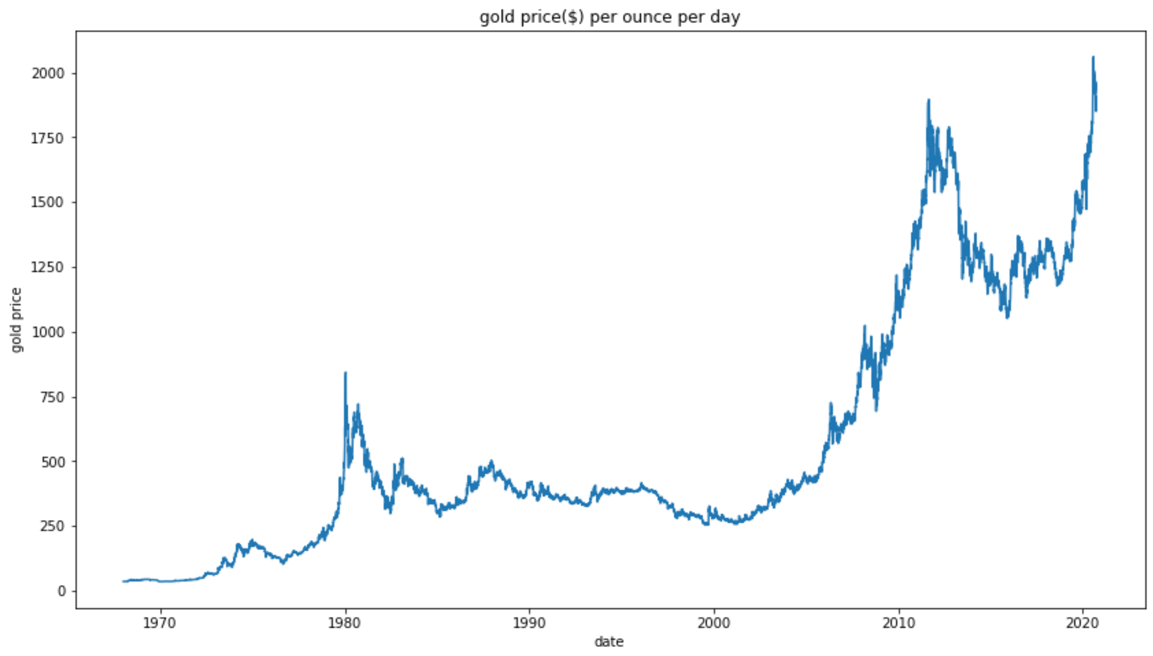
The result shows that USD(AM) has the lowest missing values, so we deleted the missing values(only 1) in that series and picked that as our target series.

Then we reset the index to datetime and sort the index with ascending=True.

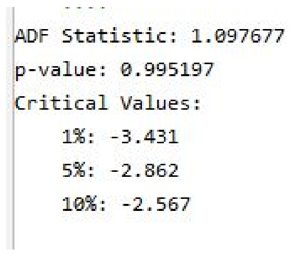


	USD
Date	
1968-01-02	35.18
1968-01-03	35.16
1968-01-04	35.14
1968-01-05	35.14
1968-01-08	35.14

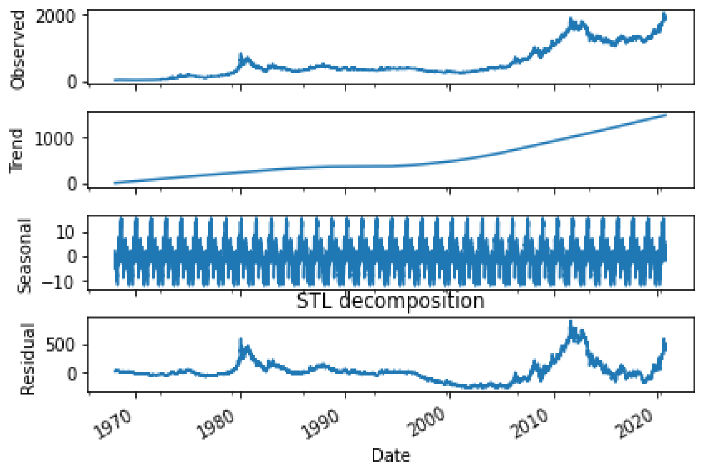
Firstly, we took a look at the target series(USD(AM)):



And we also did an ADF test to check if the series is stationary or not and used the STL method to decompose the series.

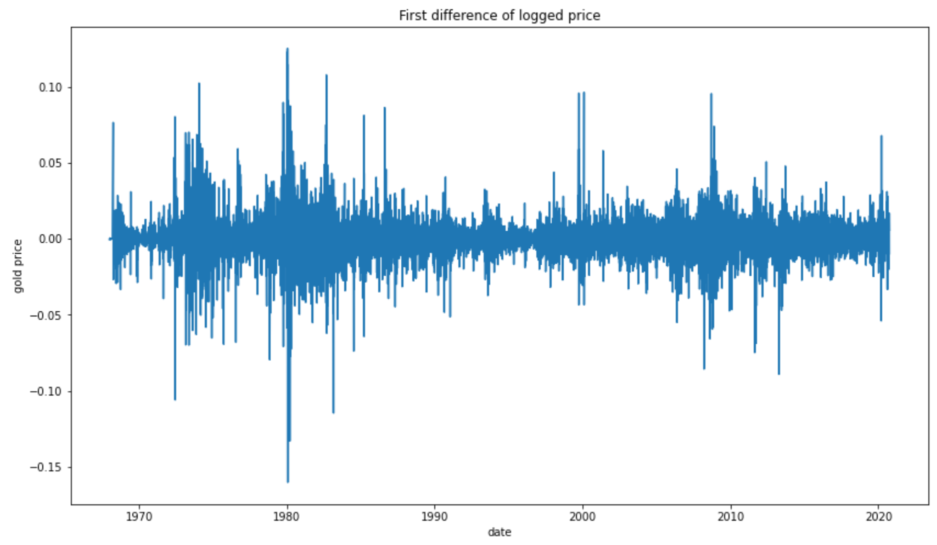


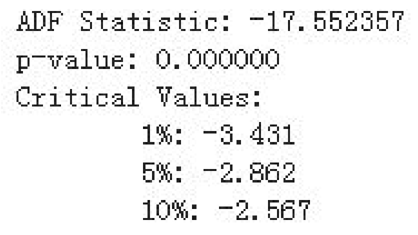
```
ADF Statistic: 1.097677
p-value: 0.995197
Critical Values:
    1%: -3.431
    5%: -2.862
    10%: -2.567
```



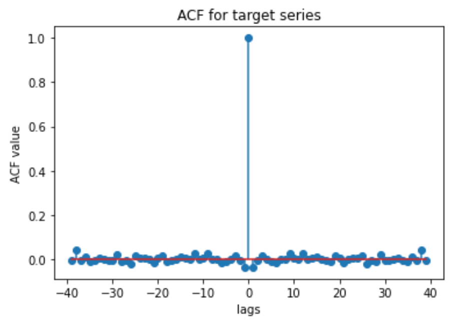
The p-value which is 0.995 from the ADF test and from the plot, we could see that the target series is very non-stationary. The STL decomposition shows the series has an increasing trend and the seasonality looks monthly to me.

To build linear methods of time series, we needed a stationary series, so I firstly log transformed the series with Base-10 logarithm. After that, I used the first difference to transform the series. For example, the raw price of 01/01/2020 is 200 and 01/02/2020 is 300, after logged transformation, the price will be 5.298 and 5.704 roundly. After the first difference transformation, the price of 01/02/2020 will be 5.704-5.298=0.4057 roundly.
Then I plot the transformed series and apply an ADF test to check if this series is stationary or not. I also plot the autocorrelation plot with 40 lags to check if the series is autocorrelated or not.





```
ADF Statistic: -17.552357
p-value: 0.000000
Critical Values:
        1%: -3.431
        5%: -2.862
        10%: -2.567
```



The p-value of ADF test is about 0 and the plot also shows a constant mean and variance through the time. So the transformed series is stationary now, and the autocorrelation plot shows the series is not correlated with each other.

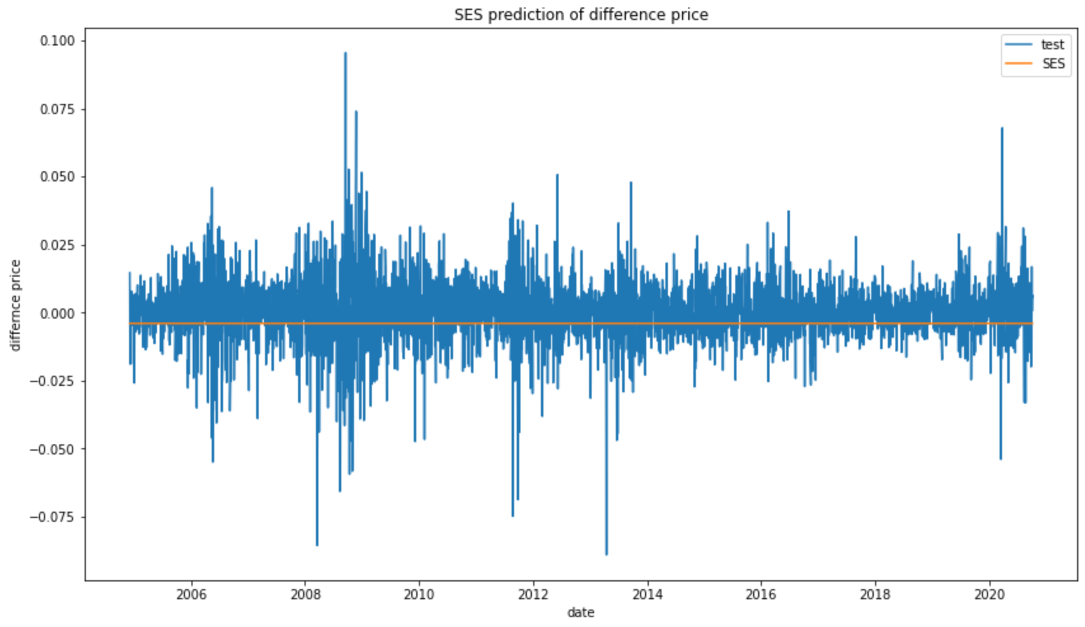
Data modeling methods

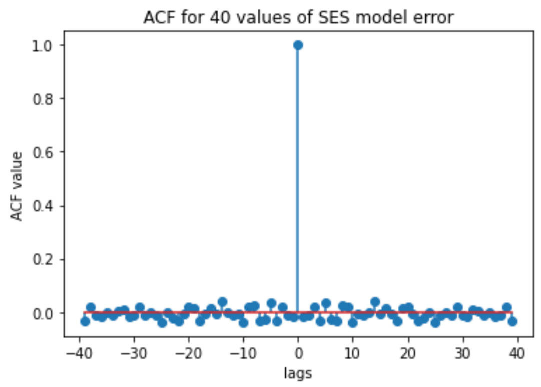
- Split train(70%), test(30%) series
- Linear(transformed series)
  - Simple exponential smoothing
  - Holt Linear
  - Holt Winter
  - ARMA, ARIMA, SARIMA
- Nonlinear(Raw price of USD(AM) & USD(PM))
  - LSTM, GRU

Simple exponential smoothing

$$s_t = \alpha x_t + (1-\alpha)s_{t-1} = s_{t-1} + \alpha(x_t - s_{t-1}).$$

The algorithm of simple exponential smoothing is above. I set alpha which is smoothing level to be 0.5 for my series.
Here is the prediction of the SES model and autocorrelation plot with 40 lags of prediction error and error evaluation results.





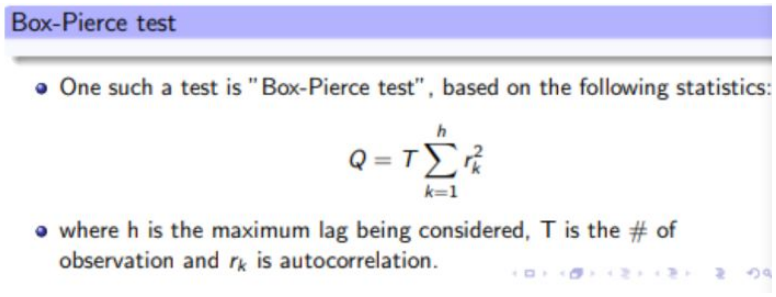
```
The Q value of SES model is: 69.43888557773676
The variance of SES model is: 0.00013199612908907233
The mse of SES model is: 0.00015099578690296733
The mean of SES model error is: 0.0043588596919257455
RMSE of SES error is: 0.012288034297761678
```

Mean squared error

$$MSE = mean(e_t^2)$$

Root Mean squared error

$$RMSE = \sqrt{mean(e_t^2)}$$



Box-Pierce test

- One such a test is "Box-Pierce test", based on the following statistics:

$$Q = T\sum_{k=1}^{h} r_k^2$$

- where h is the maximum lag being considered, T is the # of observation and r_k is autocorrelation.

The Q-value is the sum of the autocorrelation of errors. Low Q-value means the error is close to a white noise. From the autocorrelation plot, we can see that the error is not autocorrelated to each other and the Q-value is not high which means the model performs good.

Holt Linear Method

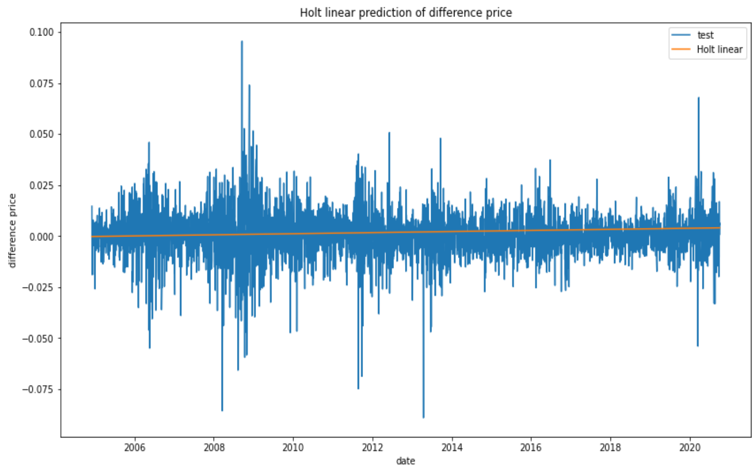
$$(1)\, Forecast\ p_{t+h|t} = \underbrace{l_t}_{level} + \underbrace{hb_t}_{trend}$$

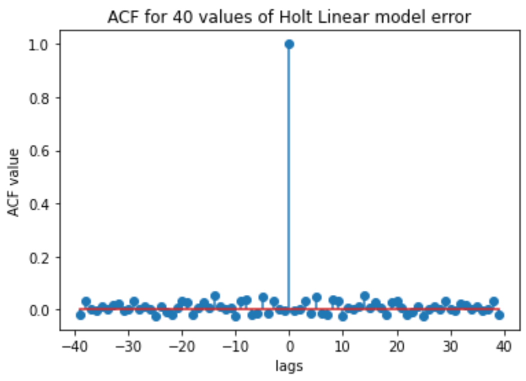
$$(2)\, l_t = \alpha y_t + (1-\alpha) l_{t-1}\ (\alpha = smoothing\, level)$$

$$(3)\, b_t = \beta(l_t - l_{t-1}) + (1-\beta) b_{t-1}\ (\beta = smoothing\, slope)$$

Here is the algorithm of the Holt linear model. It is usually used to predict a series with trend and no seasonality. Compared to SES, the Holt linear method has a smoothing slope more than smoothing level.

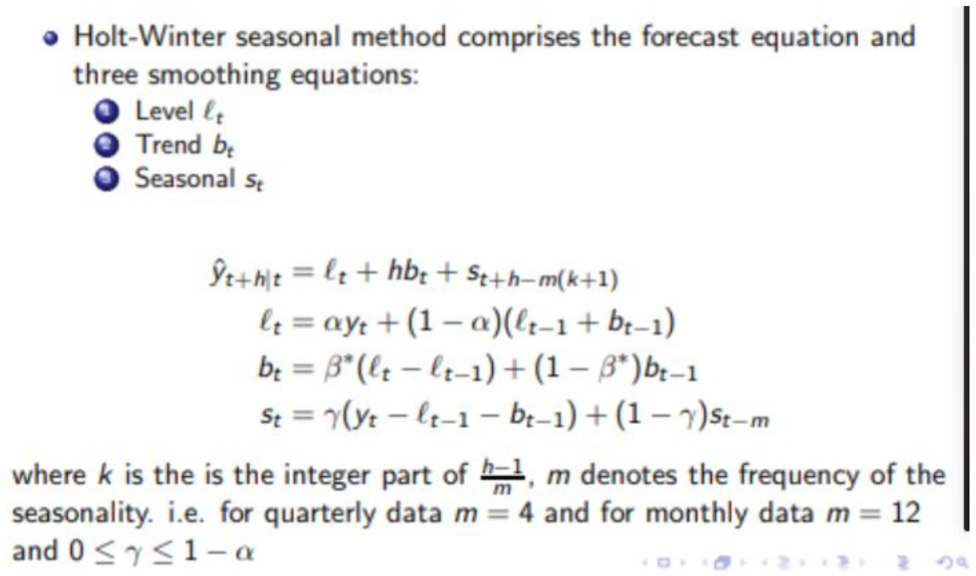
Because the series does not have a trend after transformation, so I set smoothing_level = 0.005, smoothing_slope = 0.4.





```
The Q value of Holt Winter model is: 73.47307322520471
The variance of Holt Winter model is: 0.00013386008728222222
The mse of Holt Winter model is: 0.0001361808272383433
The mean of Holt Winter model error is: -0.001523397504304479
RMSE of Holt Winter error is: 0.011669654118196619
```

Holt Winter method

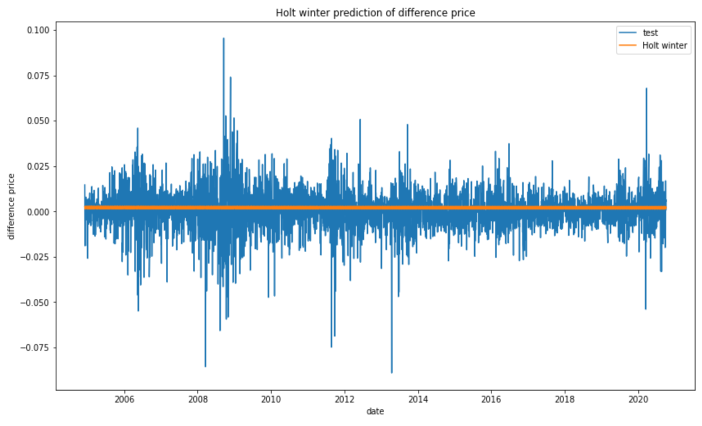


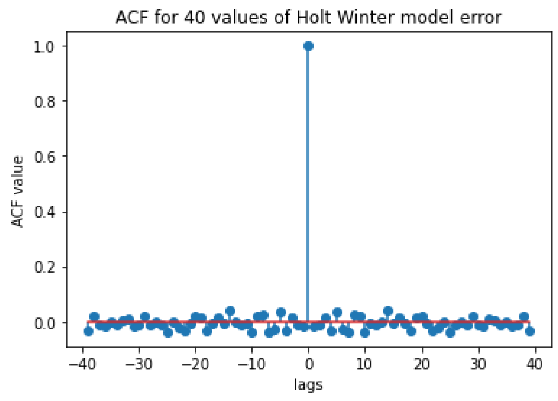
- Holt-Winter seasonal method comprises the forecast equation and three smoothing equations:
  1. Level ℓ_t
  2. Trend b_t
  3. Seasonal s_t

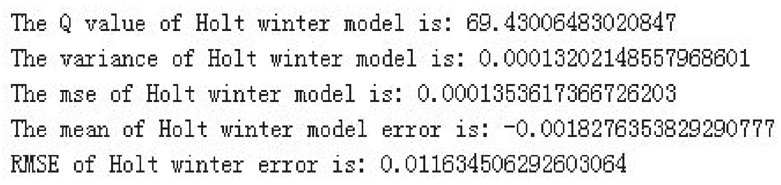
$$\begin{aligned}\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)}\\ \ell_t &= \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})\\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}\\ s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}\end{aligned}$$

where k is the is the integer part of $\frac{h-1}{m}$, m denotes the frequency of the seasonality. i.e. for quarterly data $m = 4$ and for monthly data $m = 12$ and $0 \le \gamma \le 1 - \alpha$

Compared to Holt linear method, Holt winter method has one more component of seasonal. The STL decomposition method shows a monthly seasonality to me so I set seasonal_periods=12. The trend from STL decomposition is increasing, the proportion of the increase of seasonality is not increasing, so I set trend=additive, seasonal=additive but not multiplicative. Because there are some negative values in the transformed series, so I did not use box cox transformation while fitting.

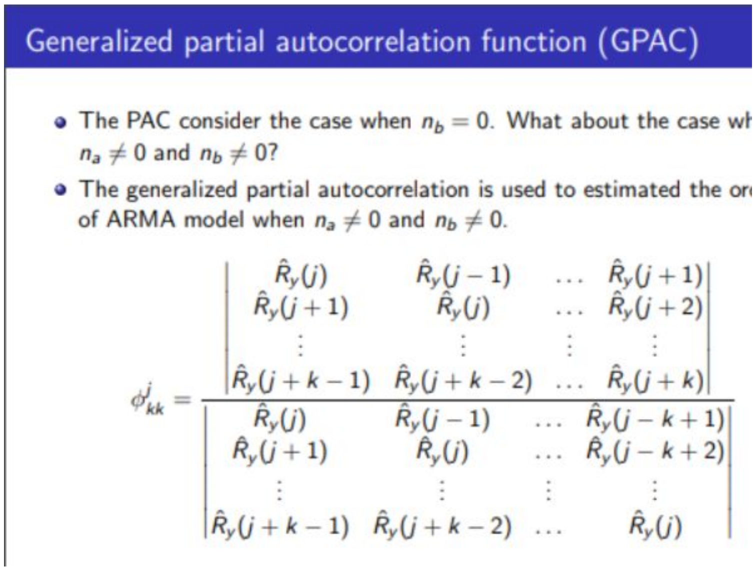






```
The Q value of Holt winter model is: 69.43006483020847
The variance of Holt winter model is: 0.00013202148557968601
The mse of Holt winter model is: 0.0001353617366726203
The mean of Holt winter model error is: -0.0018276353829290777
RMSE of Holt winter error is: 0.011634506292603064
```

ARMA Model



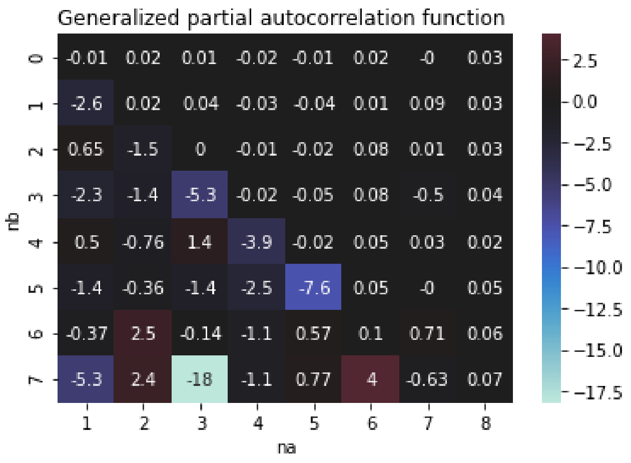
Generalized partial autocorrelation function (GPAC)

- The PAC consider the case when $n_b = 0$. What about the case wh $n_a \neq 0$ and $n_b \neq 0$?
- The generalized partial autocorrelation is used to estimated the or of ARMA model when $n_a \neq 0$ and $n_b \neq 0$.

$$\phi^{j}_{kk} = \frac{\begin{vmatrix} \hat{R}_y(j) & \hat{R}_y(j-1) & \dots & \hat{R}_y(j+1) \\ \hat{R}_y(j+1) & \hat{R}_y(j) & \dots & \hat{R}_y(j+2) \\ \vdots & \vdots & \vdots & \vdots \\ \hat{R}_y(j+k-1) & \hat{R}_y(j+k-2) & \dots & \hat{R}_y(j+k) \end{vmatrix}}{\begin{vmatrix} \hat{R}_y(j) & \hat{R}_y(j-1) & \dots & \hat{R}_y(j-k+1) \\ \hat{R}_y(j+1) & \hat{R}_y(j) & \dots & \hat{R}_y(j-k+2) \\ \vdots & \vdots & \vdots & \vdots \\ \hat{R}_y(j+k-1) & \hat{R}_y(j+k-2) & \dots & \hat{R}_y(j) \end{vmatrix}}$$

I will check the GPAC table of the dependent variable to determine the orders of the ARMA model. Autoregressive moving average (ARMA(na, nb)) models are the combination of AR(na) and MA(nb) models:

$$y(t) + a_1 y(t-1) + a_2 y(t-2) + \dots + a_{n_a} y(t-n_a) = \epsilon(t) + b_1 \epsilon(t-1) + b_2 \epsilon(t-2) + \dots + b_{n_b} \epsilon(t-n_b)$$

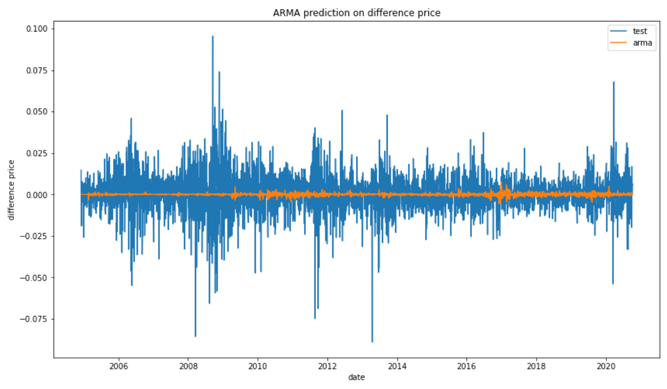


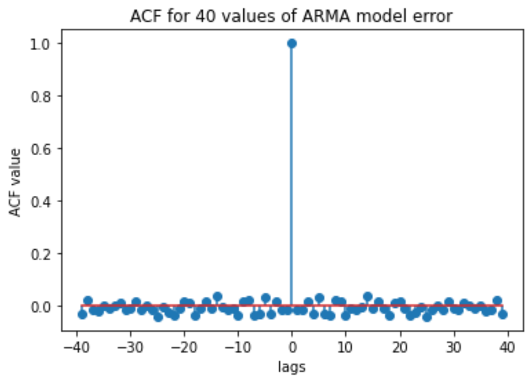
nb \ na	1	2	3	4	5	6	7	8
0	-0.01	0.02	0.01	-0.02	-0.01	0.02	-0	0.03
1	-2.6	0.02	0.04	-0.03	-0.04	0.01	0.09	0.03
2	0.65	-1.5	0	-0.01	-0.02	0.08	0.01	0.03
3	-2.3	-1.4	-5.3	-0.02	-0.05	0.08	-0.5	0.04
4	0.5	-0.76	1.4	-3.9	-0.02	0.05	0.03	0.02
5	-1.4	-0.36	-1.4	-2.5	-7.6	0.05	-0	0.05
6	-0.37	2.5	-0.14	-1.1	0.57	0.1	0.71	0.06
7	-5.3	2.4	-18	-1.1	0.77	4	-0.63	0.07

Here is the heatmap of the GPAC table. Na is the order of AR and nb is the order of MA. When na reaches 3 or 1 there is a high value in the table. So I pick na=2 and when nb=1, the partial autocorrelation is not high. So I set AR=2 and MA=1.

```
                              ARMA Model Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 9333
Model:                     ARMA(2, 1)   Log Likelihood               27390.626
Method:                       css-mle   S.D. of innovations              0.013
Date:                Wed, 25 Nov 2020   AIC                         -54773.251
Time:                        05:18:38   BIC                         -54744.686
Sample:                             0   HQIC                        -54763.549

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1.y       -0.3670      0.486     -0.755      0.450      -1.319       0.585
ar.L2.y       -0.0229      0.022     -1.065      0.287      -0.065       0.019
ma.L1.y        0.3245      0.486      0.668      0.504      -0.628       1.277
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -3.4807           +0.0000j            3.4807            0.5000
AR.2          -12.5402           +0.0000j           12.5402            0.5000
MA.1           -3.0813           +0.0000j            3.0813            0.5000
-----------------------------------------------------------------------------
```

Here is the summary of the ARMA(2,1) model. There is not constant and disp=0. The coefficients can pass the zero/pole cancellation.

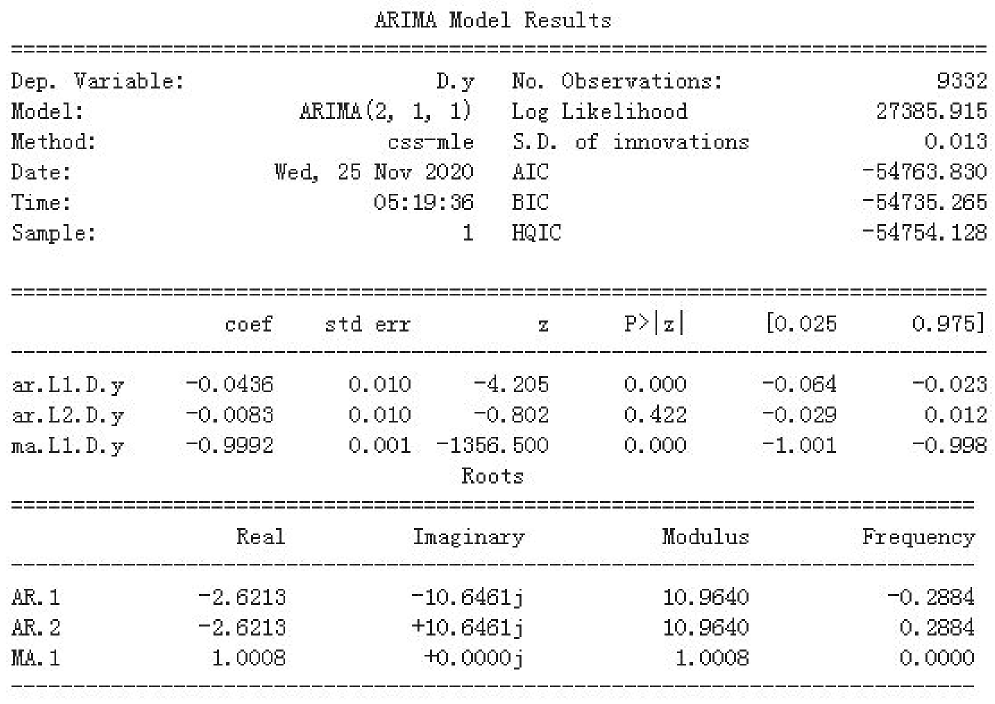




```
The Q value of ARMA model is: 68.49179302790853
The variance of ARMA model is: 0.00013275410802O9147
The mse of ARMA model is: 0.0001329069316310536
The mean of ARMA model error is: 0.0003909266045422025
RMSE of ARMA error is: 0.011528526863006113
```

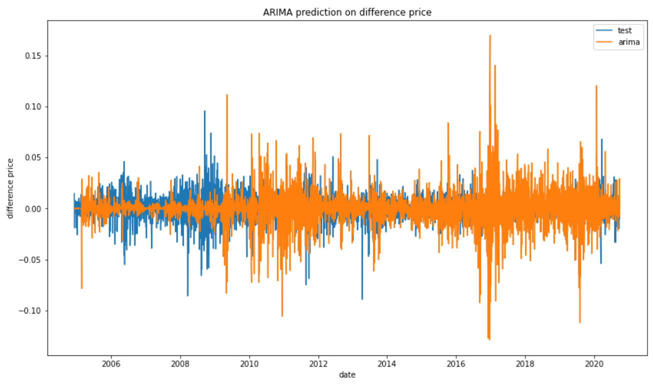
The Q-value of ARMA model error is about 68. I used a whiteness test(chi-square test) with n=40(lags)-3(orders) and alpha=0.001. The Q-value is smaller than the threshold, so the residual is a white noise.

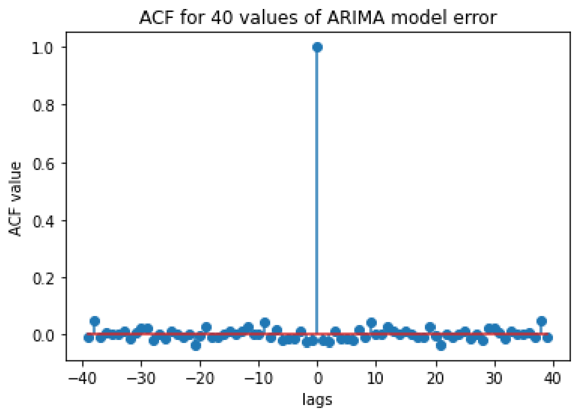
ARIMA Model

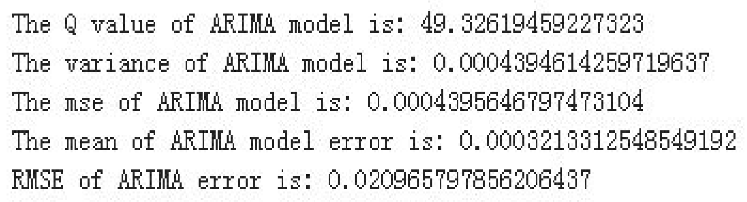


```
                              ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                 9332
Model:                 ARIMA(2, 1, 1)   Log Likelihood               27385.915
Method:                       css-mle   S.D. of innovations              0.013
Date:                Wed, 25 Nov 2020   AIC                         -54763.830
Time:                        05:19:36   BIC                         -54735.265
Sample:                             1   HQIC                        -54754.128

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1.D.y     -0.0436      0.010     -4.205      0.000      -0.064      -0.023
ar.L2.D.y     -0.0083      0.010     -0.802      0.422      -0.029       0.012
ma.L1.D.y     -0.9992      0.001  -1356.500      0.000      -1.001      -0.998
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -2.6213          -10.6461j           10.9640           -0.2884
AR.2           -2.6213          +10.6461j           10.9640            0.2884
MA.1            1.0008           +0.0000j            1.0008            0.0000
-----------------------------------------------------------------------------
```

I used the first difference to the series, so the order of I in ARIMA is 1. There is not constant and disp=0





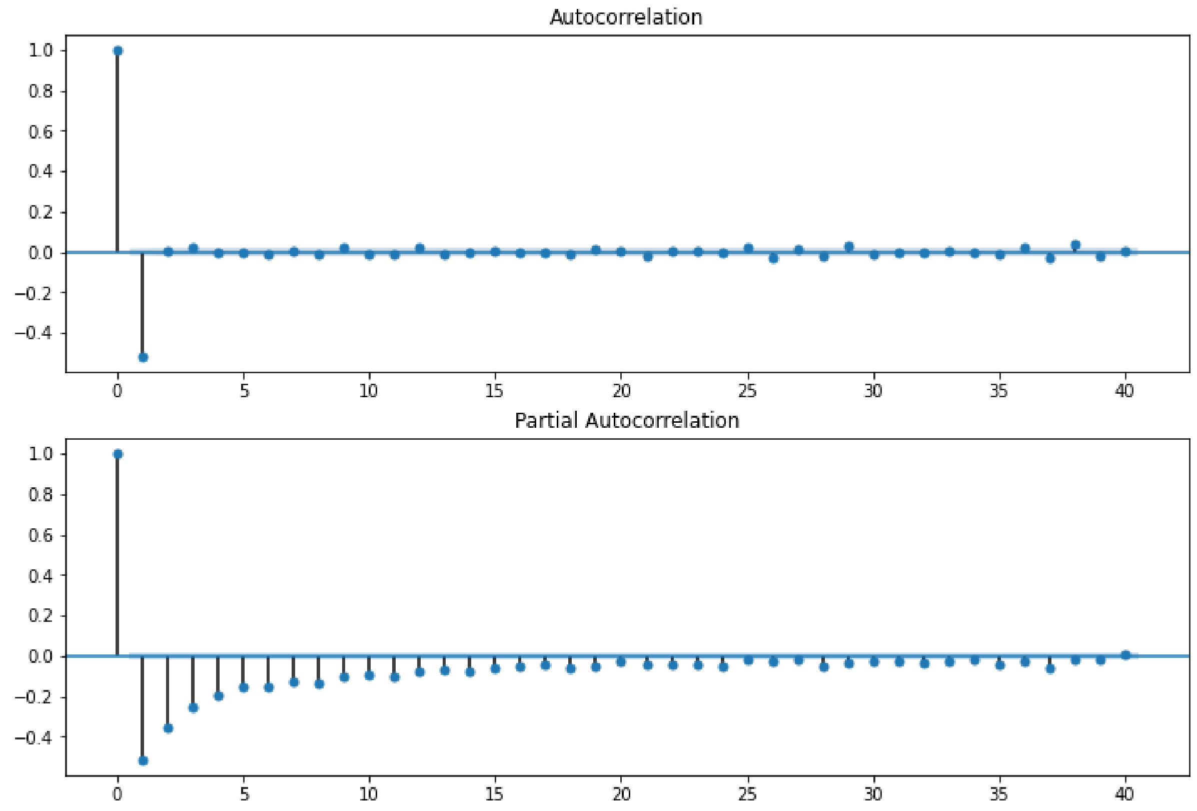


```
The Q value of ARIMA model is: 49.32619459227323
The variance of ARIMA model is: 0.00043946142597l9637
The mse of ARIMA model is: 0.0004395646797473104
The mean of ARIMA model error is: 0.0003213312548549192
RMSE of ARIMA error is: 0.020965797856206437
```

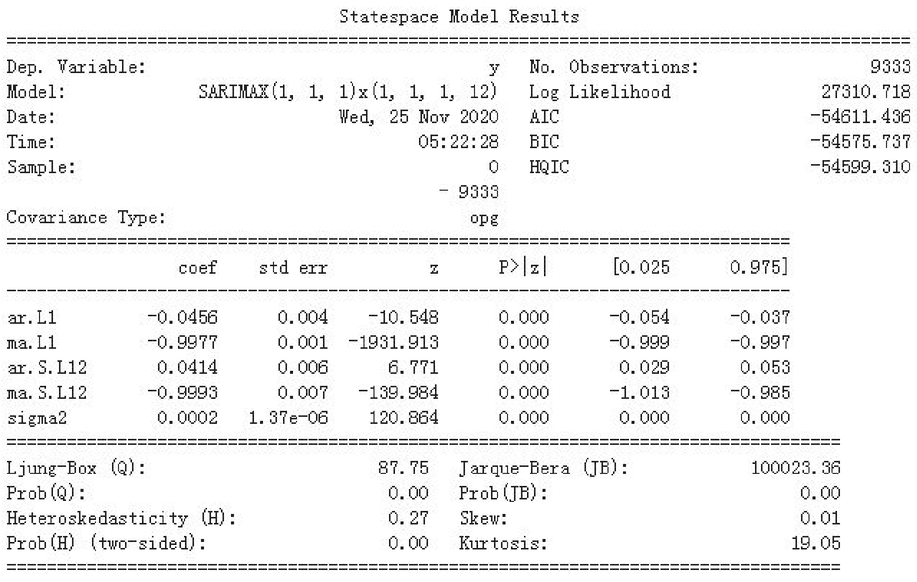
The Q-value of the ARIMA model is about 49.3 which can pass the whiteness test.

SARIMA Model

For the SARIMA model, I used ACF and PACF plots to determine the seasonal order of SARIMA.

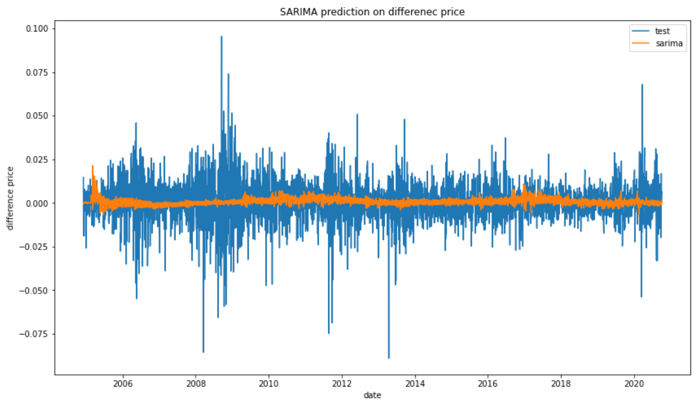


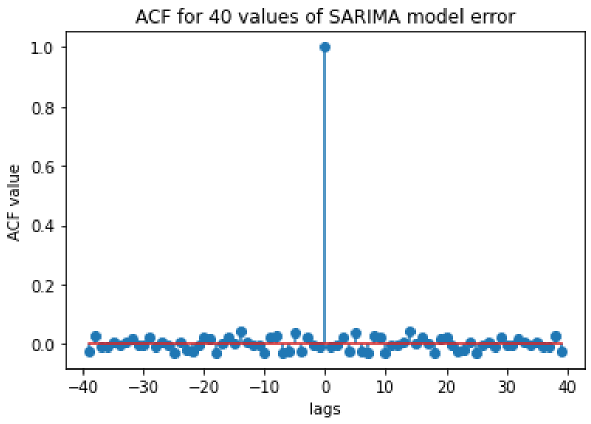
When lag is 1, the autocorrelation and partial autocorrelation is decreasing. So I set the seasonal order to be (1,1,1,12) as the series's seasonality is monthly.

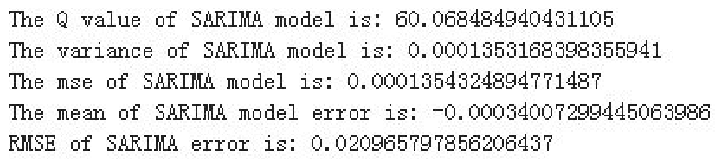


```
                                      Statespace Model Results
==========================================================================================
Dep. Variable:                                 y   No. Observations:                 9333
Model:             SARIMAX(1, 1, 1)x(1, 1, 1, 12)   Log Likelihood               27310.718
Date:                            Wed, 25 Nov 2020   AIC                         -54611.436
Time:                                    05:22:28   BIC                         -54575.737
Sample:                                         0   HQIC                        -54599.310
                                           - 9333
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.0456      0.004    -10.548      0.000      -0.054      -0.037
ma.L1         -0.9977      0.001  -1931.913      0.000      -0.999      -0.997
ar.S.L12       0.0414      0.006      6.771      0.000       0.029       0.053
ma.S.L12      -0.9993      0.007   -139.984      0.000      -1.013      -0.985
sigma2         0.0002   1.37e-06    120.864      0.000       0.000       0.000
===================================================================================
Ljung-Box (Q):                       87.75   Jarque-Bera (JB):             100023.36
Prob(Q):                              0.00   Prob(JB):                          0.00
Heteroskedasticity (H):               0.27   Skew:                              0.01
Prob(H) (two-sided):                  0.00   Kurtosis:                         19.05
===================================================================================
```

There is not constant in the model and I enforce_stationarity=True enforce_invertibility=True.

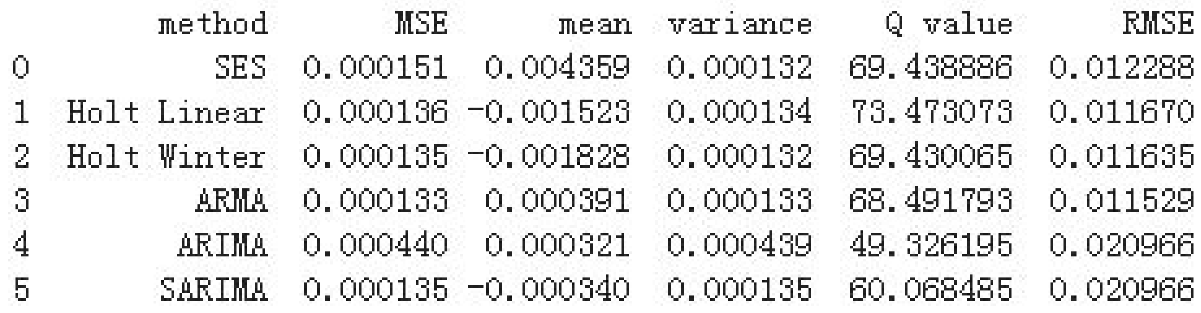






```
The Q value of SARIMA model is: 60.068484940431105
The variance of SARIMA model is: 0.0001353168398355941
The mse of SARIMA model is: 0.0001354324894771487
The mean of SARIMA model error is: -0.00034007299445063986
RMSE of SARIMA error is: 0.020965797856206437
```

Prediction Comparison of linear methods



```
        method       MSE      mean  variance    Q value      RMSE
0          SES  0.000151  0.004359  0.000132  69.438886  0.012288
1  Holt Linear  0.000136 -0.001523  0.000134  73.473073  0.011670
2  Holt Winter  0.000135 -0.001828  0.000132  69.430065  0.011635
3         ARMA  0.000133  0.000391  0.000133  68.491793  0.011529
4        ARIMA  0.000440  0.000321  0.000439  49.326195  0.020966
5       SARIMA  0.000135 -0.000340  0.000135  60.068485  0.020966
```

Compared to using MSE and RMSE(which are small) as criteria, I prefer using Q values of the residuals. The ARIMA(2,1,1) model which has the lowest Q value is best for me. That means the residual of the ARIMA model is closest to white noise.

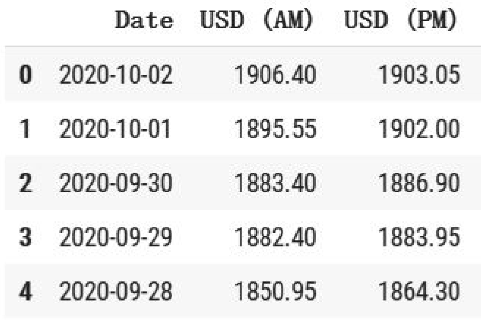
Neural Network

Long Short Term Memory

For Neural networks, we try Long Short Term Memory. The algorithm is shown below.

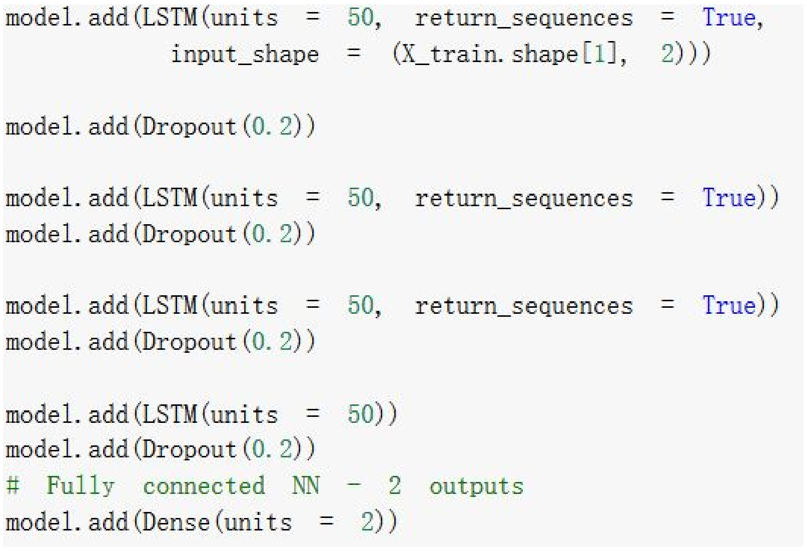
$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t\ 1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t\ 1} + b_c) \\
c_t &= f_t \circ c_{t\ 1} + i_t \circ \tilde{c}_t \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

For data processing, use the data USD(AM) and USD(PM) because they have least Na values.



	Date	USD (AM)	USD (PM)
0	2020-10-02	1906.40	1903.05
1	2020-10-01	1895.55	1902.00
2	2020-09-30	1883.40	1886.90
3	2020-09-29	1882.40	1883.95
4	2020-09-28	1850.95	1864.30

For the input, to predict the gold price on day X. We choose the sequence of past N days' prices as the input of the network.
The structure of the LSTM model is shown below. The network has 4 layers and each has 50 neurons.

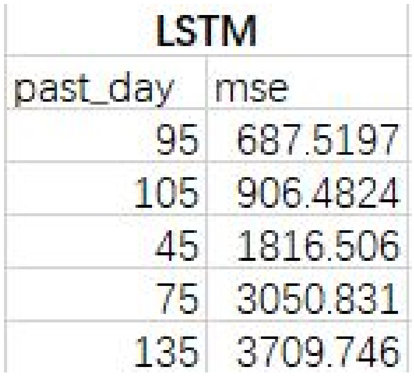


```
model.add(LSTM(units = 50, return_sequences = True,
          input_shape = (X_train.shape[1], 2)))

model.add(Dropout(0.2))

model.add(LSTM(units = 50, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units = 50, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units = 50))
model.add(Dropout(0.2))
# Fully connected NN - 2 outputs
model.add(Dense(units = 2))
```

We also tried other structure:

1. 4 layers with 100 neurons each layer. The MSE increases about 9.6% in average compared with the above network.
2. 6 layers with 50 neurons each layer. The MSE increases about 2% on average but the training process takes much longer. So we don't use this structure.

To determine N, we do grid research by using MSE for evaluation.



LSTM	
past_day	mse
95	687.5197
105	906.4824
45	1816.506
75	3050.831
135	3709.746

As we can see, when N equals to 95, the network has the lowest MSE. Thus we use the sequence of past 95 days' prices as the input sequence of LSTM.

Gated Recurrent Unit

The Gated Recurrent Unit is a simplified version of LSTM. It can also process sequences of data. The algorithm of GRU is shown below.

$$
\begin{aligned}
z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z)\\
r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r)\\
\hat{h}_t &= \phi_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_{\hat{h}})\\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t
\end{aligned}
$$

For the input, use the same input with LSTM - USE(AM) and USD(PM).

For the structure of the network, the same problem happened.
We tried other structure:

3. 4 layers with 100 neurons each layer. The MSE increases about 4.9% in average compared with the original network.
4. 6 layers with 50 neurons each layer. The MSE increases about 12% on average.

Thus, we remain the original structure. (4 layers * 50 neurons)
Similarly, do grid research to determine the N. The result is shown below.

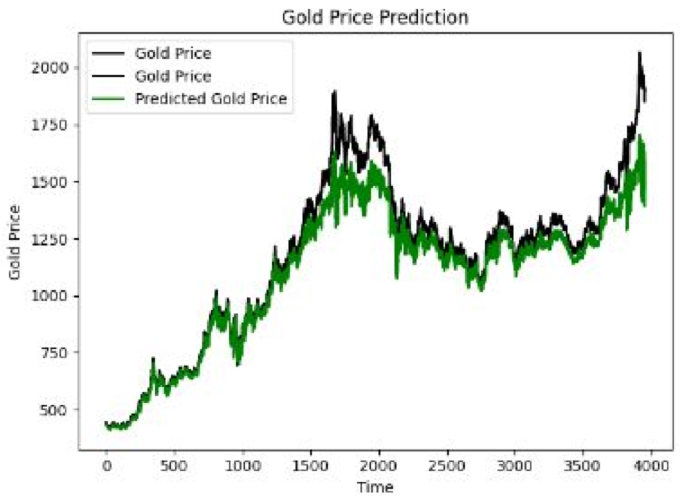
GRU	
past_day	mse
95	3547.471
125	4677.879
100	9162.113
120	12662.72
85	16382.43

Same to LSTM, the GRU has the lowest MSE when N equals 95. Thus we use the sequence of past 95 days' prices as the input sequence of GRU.

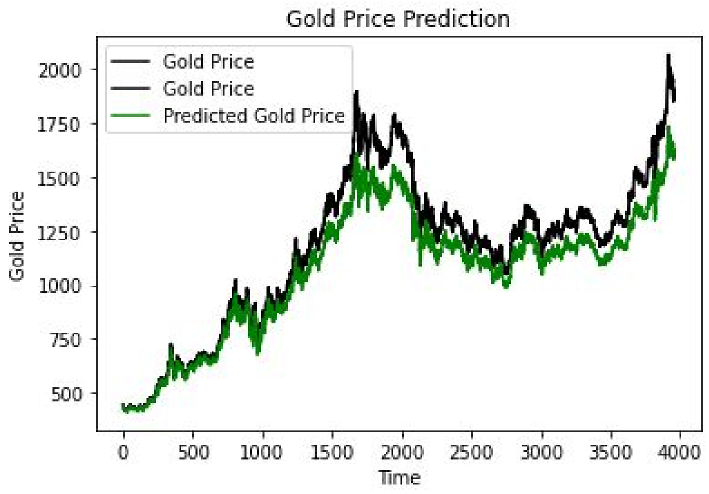
Result Comparison

LSTM performs better than GRU with less MSE.

LSTM:



GRU:



Conclusion

The ARIMA(2,1,1) model which has the lowest Q value is best of the linear methods to predict the preprocessed series.

LSTM performs best in Non-linear Methods. And for LSTM sequences of past 95 days' prices get the best prediction.

Limitation

First of all, the dataset's range is big and data is not stationary most of the time. Also, the residual of linear methods cannot pass the whiteness test on 0.05 alpha, which could be a problem.

Last but not least, the gold price will be affected by many unforecast factors which will cause the prediction to be inaccurate. For example, the gold price is stationary between the period from 2010 to 2019 but rapidly increases in the year of 2020.

Improvement

For linear models, we may use AIC and BIC to determine the order of AR&MA. And use bootstrap and bagging on linear models to achieve higher accuracy. And we can try grid research for order of na and nb.

For Non-linear models, we can try large scale Grid Research for parameter tuning (learning rate). We may try different structures of the network like to make it much deeper and wider to see if there is any improvement of performance.

Reference

- Dr.Reza Jafari's Slides: https://www.investopedia.com/terms/a/autocorrelation.asp#:~:text=Autocorrelation%20represents%20the%20degree%20of,value%20and%20its%20past%20values
- https://towardsdatascience.com/time-series-in-python-exponential-smoothing-and-arima-processes-2c67f2a52788
- https://www.statsmodels.org/dev/examples/notebooks/generated/statespace_sarimax_stata.html
- https://otexts.com/fpp2/advanced.html
- https://www.analyticsvidhya.com/blog/2018/02/time-series-forecasting-methods/
- https://www.hindawi.com/journals/ijap/2013/402842/

Github: **https://github.com/olokojoh/Data-Science-Capstone-DATS6501**