FIT5186 Intelligent Systems

Predicting the Crude Oil Price Based on Neural Networks

Group 4

Ke Huishu 2819****

Guo Xuechun 2819****

29 May 2017

SOUTHEASTUNIVERSITY - MONASH UNIVERSITY

JOINTGRADUATE SCHOOL (SUZHOU)

Abstract

Crude oil plays a crucial role in the development of world economy and politics. In this research, we aim to take advantage of various neural networks to predict the tomorrow crude oil price. First, we give a brief introduction of crude oil. Then, we clarify the data resource and how to preprocess the data. After this, we discuss the inputs and training issues of the research. Moreover, we conduct the experiments and display the result of these experiments. Finally, we give the limitations and conclusions of our research.

1. Introduction

Crude oil is one of the indispensable energy in the development of world economy and politics. It has an important influence on national economy. Crude oil price predicting is an important part of economic management, which has positive and important guiding significance for the government, banks, enterprises and individuals and other economic or management entities.

Currently, there are three crude oil markets, namely Brent, West Texas Intermediate (WTI) and Organization of Petroleum Exporting Countries (OPEC) (Charles and Darné, 2014). There are mainly three benchmarks: Brent Blend, WTI and Dubai Crude. Different with WTI and Dubai Crude, the liquidity of Brent is not restricted and Brent can be exported to the overseas market. As a result, the trading volume in Brent market is greater and Brent is more sensitive to the situation in the Middle East and North Africa. At present, more than 50% crude oil transactions refer to the Brent. Therefore, we choose Brent as our research object.

The crude oil price is not only influenced by the situation of economy and diplomacy, but also affected by international military politics and affairs at the same time. Therefore, the price of crude oil has some characters such as uncertainty and non-linear (Su et al., 2017). Neural Networks (NNs) have strong capability of non-linear reflecting and may improve the prediction accuracy of the crude oil price effectively. This research will apply different architectures and techniques of NNs to better achieve the prediction of the crude oil price.

2. Data Sets

In this part, the resource of data and preprocess of data are clarified.

2.1 Data Resource

For the research, the data is from Federal Reserve Economic Data (FRED), which records more than 470 thousand US and international time series from 84 sources. In this database, data can be browsed by tags, categories or sources. In order to build artificial neural network model for the crude oil price, the data of daily crude oil prices from 14th March 2016 to 13th March 2017 will be selected for prediction. Some tests show that gold price and exchange rate may impact oil price in the short term (Chang et al., 2013). As a result, the industrial production of crude oil, US dollar index (USDX) and gold process from 14th March 2016 to 13th March 2017 are also collected as contributing factors from this database. In order to improve the performance of the model, different combinations of factors will be experimented.

2. 2 Data Preprocess

For the input, we need to consider how to select the factors that may affect the crude oil price. Actually, the nature of neural network is to imitate the way of human thinking. Therefore, we should choose factors that a human expert would consider when make a precision about crude oil prices. However, some political events, policies and other factors would be hard to be described in numerical data. In addition, some factors cannot be directly used as inputs. Therefore, we need to make best of these contributing factors by preprocessing. In this case, apart from raw data, some other factors such as industrial production of crude oil, US dollar index (USDX) and gold prices are considered. Besides, the moving average and day lag of daily crude oil price would affect tomorrow's crude oil prices as well. In this situation, we chose some related input candidates as follows:

1. Today's crude oil price
2. 1-day lag of crude oil price
3. 2-day lag of crude oil price
4. 3-day lag of crude oil price
5. 5-day moving average of daily crude oil price
6. 7-day moving average of daily crude oil price
7. Today's industrial production of crude oil
8. Today's US dollar index
9. Today's gold price

Apart from processing the contributing factors, the noise of data should be removed. The pointless data like date or abnormal data need to be filtered to improve accuracy of prediction.

3. Training Issues

3.1 Architectures

There are various architectures provided within NeuroShell2. Backpropagation (BP) has high non-linear reflecting capability and generalization capability. These characteristics can improve the performance of classifying and predicting. Many kinds of backpropagation networks are available as well. The self–organizing feature map neural network (Kohonen, 1988) is trained using unsupervised learning and using a neighborhood function to preserve the topological properties of the input. Probabilistic Neural Network (PNN) is good at categorizing the input data. General Regression Neural Network (GRNN) is a branch of Radial Basis Function Neural Network (RBFNN). It can directly modify the network by sampling or calculating the data without recalculating the parameters. GRNN model consists of four layers: input layer, pattern layer, summation layer and output layer. It is known for predicting a continuous output with high efficiency and accuracy. Group Method of Data Handling (GMDH, also known as Polynomial Neural Networks) is widely used in many non-linear systems. According to the characteristics of these architectures, we preliminarily choose BP, GRNN and GMDH architectures in our experiments.

3.2 Inputs/Output

We define three groups as the input to find the relations and the factors that affect the result most. We use the number to represent the factor as introduce in 2.2, the three groups are defined as follow:
Parameter Scheme 1: 1+2+3+4
Parameter Scheme 2: 1+2+3+4+5+6
Parameter Scheme 3: 1+2+3+4+5+6+7+8+9

We set the tomorrow Brent crude oil price as output. Through the different combinations of input candidates, we hope to find the best input group to predict the tomorrow price.

3. 3 Other Parameters

The other parameters vary according to the different architectures. In the three architectures, the mainly parameters are the percentage of test set and the number of the hidden neurons.

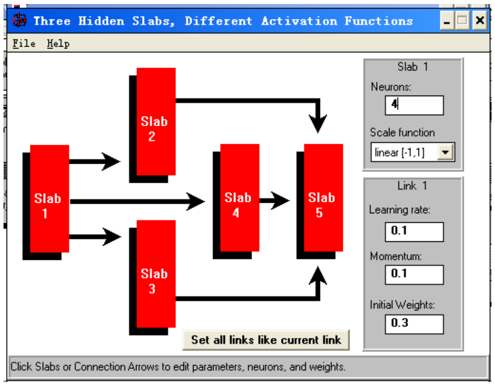
Meanwhile, we use R squared, Mean Squared Error (MSE) and Mean Absolute Error (MAE) to evaluate the architecture performance. The R square is a representation of fitting degree quality. The bigger R square is, the better the results are. MSE and MAE represent the difference between the predicted and actual results. On the contrary, the smaller MAE and MSE are, the better the results are.

4. Experiments and Results

We did 27 experiments using different architectures and different parameters. However, the 27 cannot be described in detail here. So we divide the 27 experiments into three parts according the different architectures. The experiments and their results will be further discussed in the following part. The Parameter Scheme we have mentioned in **3.2** will also be used in this part.

4.1 BP Model

For backpropagation neural network, there exist various BP architectures in NeuroShell2 as layers and circulations of data vary. The BP architecture chosen is as follow.



Three Hidden Slabs, Different Activation Functions

Slab 1

Neurons: 4

Scale function: linear [-1,1]

Link 1

Learning rate: 0.1

Momentum: 0.1

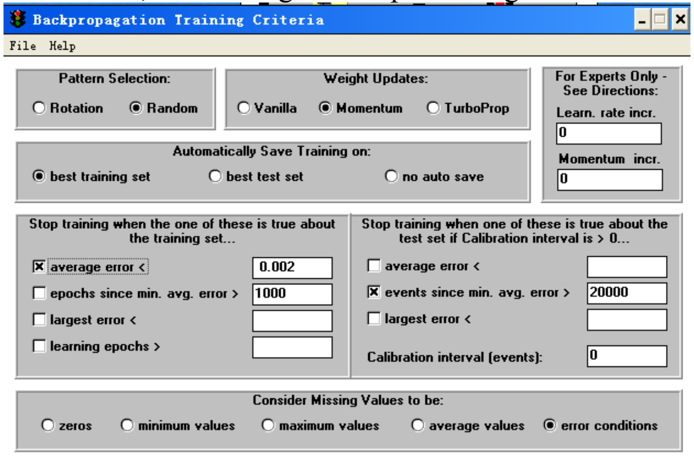
Initial Weights: 0.3

Click Slabs or Connection Arrows to edit parameters, neurons, and weights.

Figure 1. The Backpropagation Architecture

By default, the number of hidden neuron in NeuroShell2 is calculated by $J = (N+K)/2+\sqrt{P}$. In this formula, N represents the number of input dimensions, K denotes the number of output neuron while P is a representation of the training set patterns number. To verify the influence of J , we set the number of hidden neuron as a variable. We use two ways to calculate J : (1) $J = (N+K)/2+\sqrt{P}$; (2) $J = 2(N+K)$ .At the same time, the percentage of test may also influence the result so this can also be set as a variable.

In this architecture, the different number of hidden neurons are calculated as $J_1 = (N+K)/2+\sqrt{P}$ and $J_2 = 2(N+K)$. As there are three slabs, we define the number of hidden neuron as h_1 , h_2 and h_3 . And the h_1 , h_2 and h_3 satisfy $h_1 + h_2 + h_3 = J$. The Activation Function for Slab1, Slab2 and Slab3 are Gaussian, tanh and Gaussian comp respectively. The percentage of test set is 20% and 30%, this variable is defined as n . Moreover, the test set is chosen randomly from the original data. Meanwhile, the training will stop if average error < 0.002.



Backpropagation Training Criteria

Pattern Selection: ○ Rotation ● Random

Weight Updates: ○ Vanilla ● Momentum ○ TurboProp

For Experts Only - See Directions:

Learn. rate incr. 0

Momentum incr. 0

Automatically Save Training on: ● best training set ○ best test set ○ no auto save

Stop training when the one of these is true about the training set...

☒ average error < 0.002

☐ epochs since min. avg. error > 1000

☐ largest error <

☐ learning epochs >

Stop training when one of these is true about the test set if Calibration interval is > 0...

☐ average error <

☒ events since min. avg. error > 20000

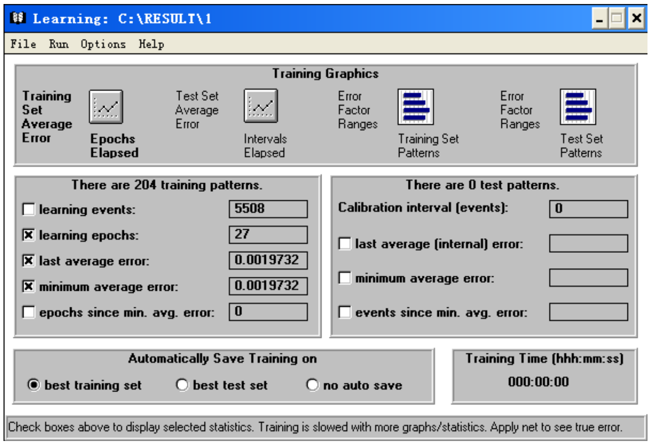
☐ largest error <

Calibration interval (events): 0

Consider Missing Values to be: ○ zeros ○ minimum values ○ maximum values ○ average values ● error conditions

Figure 2. Setting the training criteria

The learning result of training set is as follows.



There are 204 training patterns.

☐ learning events: 5508

☒ learning epochs: 27

☒ last average error: 0.0019732

☒ minimum average error: 0.0019732

☐ epochs since min. avg. error: 0

There are 0 test patterns.

Calibration interval (events): 0

☐ last average (internal) error:

☐ minimum average error:

☐ events since min. avg. error:

Automatically Save Training on

● best training set ○ best test set ○ no auto save

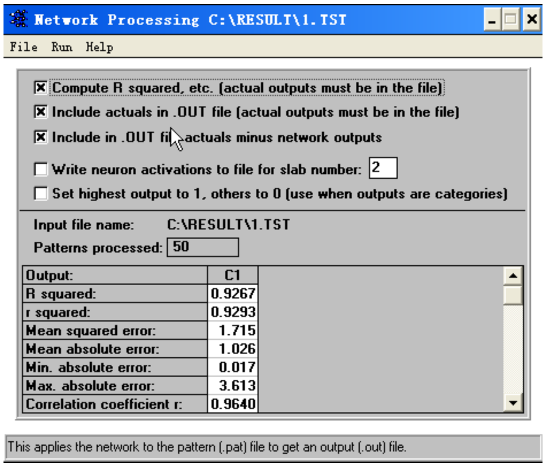
Training Time (hhh:mm:ss)

000:00:00

Check boxes above to display selected statistics. Training is slowed with more graphs/statistics. Apply net to see true error.

Figure 3. The learning result

After building neural network, we should apply the network on our test set. The R squared, MSE and MAE are calculated.



☒ Compute R squared, etc. (actual outputs must be in the file)

☒ Include actuals in .OUT file (actual outputs must be in the file)

☒ Include in .OUT file actuals minus network outputs

☐ Write neuron activations to file for slab number: 2

☐ Set highest output to 1, others to 0 (use when outputs are categories)

Input file name: C:\RESULT\1.TST

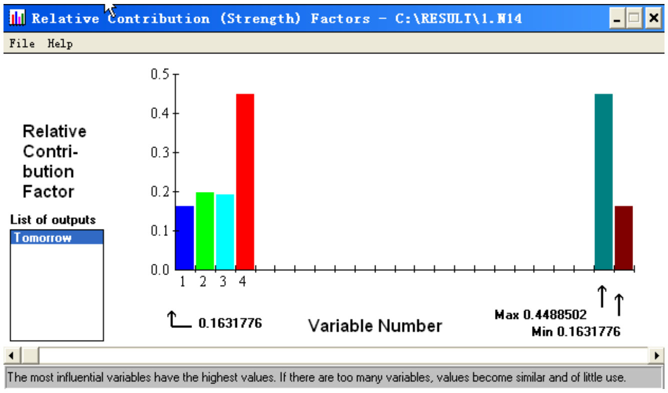
Patterns processed: 50

Output:	C1
R squared:	0.9267
r squared:	0.9293
Mean squared error:	1.715
Mean absolute error:	1.026
Min. absolute error:	0.017
Max. absolute error:	3.613
Correlation coefficient r:	0.9640

This applies the network to the pattern (.pat) file to get an output (.out) file.

Figure 4. R squared, MSE and MAE

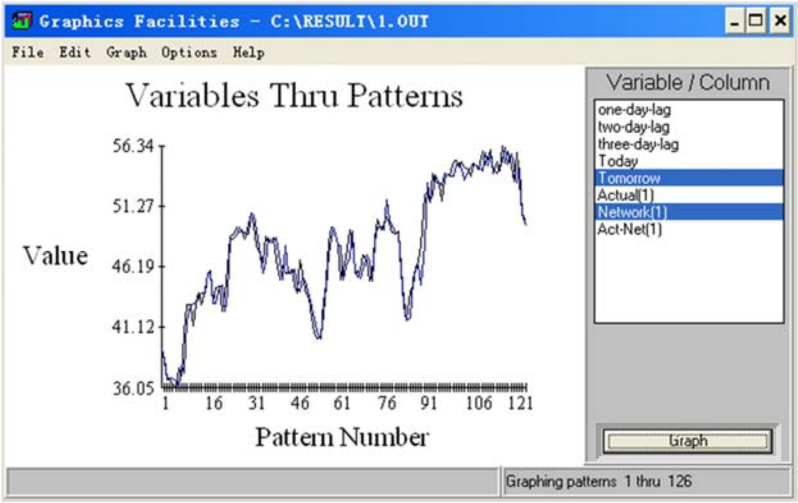
Utilizing NeuroShell2, we also can learn the influence of different factors.



The most influential variables have the highest values. If there are too many variables, values become similar and of little use.

Figure 5. Relative contribution factors

From above picture, we can find that the today crude oil price influences the tomorrow price most.
We can also observe the predict and actual data through NeuroShell2.



Graphing patterns 1 thru 126

Figure 6. The predict and actual result of experiment 1.9

The 12 experiments conducted through BP neural network are as follows.

Table 1. The R squared using BP architecture with different parameter scheme, test set percentage and number of hidden neuron

	$n = 20$, $h_{1,2,3} = 6$	$n = 30$, $h_{1,2,3} = 6$	$n = 20$, $h_{1,2,3} = 4$	$n = 30$, $h_{1,2,3} = 4$
Parameter Scheme 1	0.9267	0.9531	0.9267	0.9296
Parameter Scheme 2	0.9334	0.9224	0.9055	0.9157
Parameter Scheme 3	0.9080	0.8884	0.8994	0.8921

Table 2. The MSE using BP architecture with different parameter scheme, test set percentage and number of hidden neuron

	$n = 20$, $h_{1,2,3} = 6$	$n = 30$, $h_{1,2,3} = 6$	$n = 20$, $h_{1,2,3} = 4$	$n = 30$, $h_{1,2,3} = 4$
Parameter Scheme 1	1.715	0.985	1.715	1.980
Parameter Scheme 2	1.508	1.930	2.140	2.098
Parameter Scheme 3	1.974	2.649	2.158	2.561

Table 3. The MAE using BP architecture with different parameter scheme, test set percentage and number of hidden neuron

	$n = 20$, $h_{1,2,3} = 6$	$n = 30$, $h_{1,2,3} = 6$	$n = 20$, $h_{1,2,3} = 4$	$n = 30$, $h_{1,2,3} = 4$
Parameter Scheme 1	1.026	0.760	1.028	1.083
Parameter Scheme 2	0.983	1.052	1.176	1.090
Parameter Scheme 3	1.130	1.282	1.249	1.302

4.2 GRNN Model

In this part, the architecture is as follow.

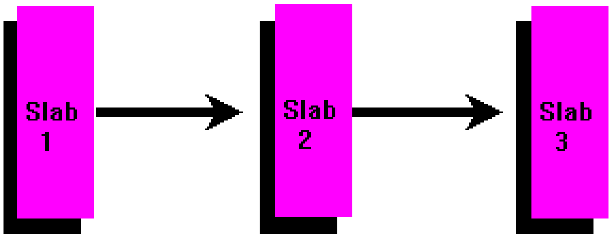


Figure 7. The GRNN architecture

In this architecture, the hidden neuron number should be the same with training set patterns number. Meanwhile, the performance is not as good as the other two architectures, we just choose the n as variable. The J will also change along with n. The following is the result of this architecture.

Table 4. The R squared using GRNN architecture with different parameter scheme, test set percentage and number of hidden neuron

	$n = 20$, $J = 201$	$n = 30$, $J = 173$
Parameter Scheme 1	0.8400	0.8405
Parameter Scheme 2	0.8855	0.8769
Parameter Scheme 3	0.8964	0.9034

Table 5. The MSE using GRNN architecture with different parameter scheme, test set percentage and number of hidden neurons

	$n = 20$, $J = 201$	$n = 30$, $J = 173$
Parameter Scheme 1	3.743	4.482
Parameter Scheme 2	2.593	3.063
Parameter Scheme 3	2.222	2.292

Table 6. The MAE using GRNN architecture with different parameter scheme, test set percentage and number of hidden neurons

	$n = 20$, $J = 201$	$n = 30$, $J = 173$
Parameter Scheme 1	1.606	1.727
Parameter Scheme 2	1.310	1.403
Parameter Scheme 3	1.195	1.165

4.3 GMDH Model

In this part, the architecture is as follow.

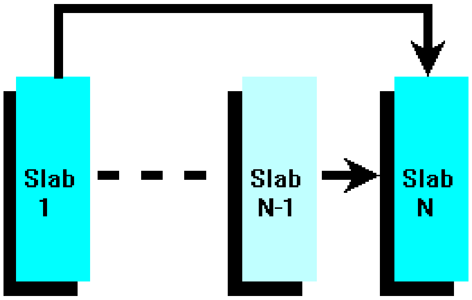


Figure 8. The GMDH architecture

In this architecture, the hidden neuron number cannot be set. So we just choose n as the variable. The n can be set as 20, 30, 50. The following is the result of this architecture.

Table 7. The R squared using GMDH architecture with different parameter scheme, test set percentage

	$n = 20$	$n = 30$	$n = 50$
Parameter Scheme 1	0.9499	0.9563	0.9482
Parameter Scheme 2	0.9411	0.9419	0.9432
Parameter Scheme 3	0.9370	0.9420	0.9444

Table 8. The MSE using GMDH architecture with different parameter scheme, test set percentage

	$n = 20$	$n = 30$	$n = 50$
Parameter Scheme 1	1.1725	1.2290	1.2977
Parameter Scheme 2	1.3334	1.4462	1.3085
Parameter Scheme 3	1.4968	1.2436	1.2674

Table 9. The MAE using GMDH architecture with different parameter scheme, test set percentage

	$n = 20$	$n = 30$	$n = 50$
Parameter Scheme 1	0.8229	0.8081	0.8227
Parameter Scheme 2	0.9216	0.9283	0.8768
Parameter Scheme 3	0.9139	0.8648	0.8643

4.4 Analysis and Comparison

The experiment that produced biggest R squared is conducted by GMDH architecture with $n = 30$. The results of R Squared, MSE, MAE are 0.9563, 1.2290 and 0.8081 respectively. The smallest MAE and MSE is conducted by BP, n = 30 and $h_{1,2,3}$= 6. From above results of various architectures, we can find that the BP and GMDH perform better while GRNN performs worst. The BP performs almost similar to GMDH. In this case, it seems that GRNN is not suitable for predicting problem. Meanwhile, both the percentage of test set and hidden neuron number will affect the predicting result in some degree. Moreover, the performance on R squared is usually proportional to the performance on MSE and MAE.

5. Limitations

In our research, we have considered the percentage of test set and hidden neuron number. Meanwhile, we produced good result for predicting. However, there are also many factors and architecture we have not taken into account such as learning rate and other combination of input. Furthermore, we just use the architectures that embedded in Neuroshell2, we can utilize other architectures or mix architectures in the future research to get better result.

6. Conclusion

In our research, we use more than 200 patterns and three different architectures to predict the tomorrow crude oil price. We utilize the BP, GRNN and GMDH architectures. In general, the BP and GMDH perform better than GRNN. We also devote ourselves to find whether the percentage of test set or number of hidden neurons will influence the result. Meanwhile, we have got a good result that R squared is more than 0.95. This means that we can take advantage of GMDH to get effective prediction of tomorrow crude oil price.

References

Chang, H. F., Huang, L. C., & Chin, M. C. (2013). Interactive relationships between crude oil prices, gold prices, and the NT–US dollar exchange rate—a Taiwan study. *Energy Policy, 63*(63), 441-448.

Charles, A., & Darné, O. (2014). Volatility persistence in crude oil markets. *Energy Policy,65*(65), 729-742.

Kohonen, T. (1988). Self-organized formation of topologically correct feature maps. *Neurocomputing: foundations of research*. MIT Press.

Su, C. W., Li, Z. Z., Chang, H. L., & Lobonţ, O. R. (2017). When will occur the crude oil bubbles?. *Energy Policy,102*, 1-6.