دانشگاه تهران

دانشکده علوم و فنون نوین

گزارش پروژه شبکه های عصبی

پیش بینی قیمت طلا با استفاده از LSTM

نگارنده
محمد متقی

استاد
دکتر ویسی

بهمن ۱۴۰۳

چکیده

پیش‌بینی قیمت طلا به دلیل نوسانات شدید و تأثیرپذیری از عوامل اقتصادی و سیاسی، یکی از چالش‌های مهم در حوزه مالی محسوب می‌شود. در سال‌های اخیر، استفاده از شبکه‌های عصبی مصنوعی (ANN) و مدل‌های پیشرفته مانند LSTM (شبکه‌های حافظه کوتاه‌مدت بلند) به دلیل توانایی آن‌ها در یادگیری الگوهای پیچیده و وابستگی‌های زمانی، به طور گسترده‌ای در این زمینه به کار گرفته شده است. این روش‌ها با استفاده از داده‌های تاریخی و شاخص‌های اقتصادی، می‌توانند پیش‌بینی‌های قابل اعتمادی ارائه دهند. بهینه‌سازی مدل‌های عصبی با تنظیم پارامترها، افزایش داده‌های آموزشی، و ترکیب آن‌ها با روش‌های ترکیبی دیگر می‌تواند دقت و قابلیت اطمینان این پیش‌بینی‌ها را بهبود بخشد. در این مطالعه، کاربرد شبکه‌های عصبی در پیش‌بینی قیمت طلا مورد بررسی قرار گرفته و عملکرد آن‌ها تحلیل شده است.

فهرست مطالب

عنوان **صفحه**

فهرست اشکال

عنوان **صفحه**

فصل اول
مقدمه

در این پروژه، از روش‌های یادگیری عمیق و به طور خاص شـــبکه‌های حافظه کوتاه‌مدت بلند (LSTM) برای پیش‌بینی قیمت طلا اسـتفاده شـده اسـت. داده‌های مورد استفاده شامل قیمت روزانه طلا از سال ۲۰۱۳ تا ۲۰۲۳ است که از یک فایل CSV اسـتخراج شـده اسـت. هدف این مطالعه بررسـی روند تغییرات قیمت و ارائه یک مدل قابل اعتماد برای پیش‌بینی قیمت آینده بوده است.

فصل دوم
پیش پردازش و تحلیل تصویری داده ها

۲-۱پیش پردازش داده ها

در ابتدا، مجموعه داده‌ای که شامل قیمت‌های روزانه طلا از سال ۲۰۱۳ تا ۲۰۲۳ است، بارگذاری شده است. این داده‌ها شامل تاریخ، قیمت باز شدن، بیشترین و کمترین قیمت، قیمت پایانی و حجم معاملات هستند.

	Date	Price	Open	High	Low	Vol.	Change %
0	12/30/2022	1,826.20	1,821.80	1,832.40	1,819.80	107.50K	0.01%
1	12/29/2022	1,826.00	1,812.30	1,827.30	1,811.20	105.99K	0.56%
2	12/28/2022	1,815.80	1,822.40	1,822.80	1,804.20	118.08K	-0.40%
3	12/27/2022	1,823.10	1,808.20	1,841.90	1,808.00	159.62K	0.74%
4	12/26/2022	1,809.70	1,805.80	1,811.95	1,805.55	NaN	0.30%
...	...	...	...	...	...	...	...
2578	01/08/2013	1,663.20	1,651.50	1,662.60	1,648.80	0.13K	0.97%
2579	01/07/2013	1,647.20	1,657.30	1,663.80	1,645.30	0.09K	-0.16%
2580	01/04/2013	1,649.90	1,664.40	1,664.40	1,630.00	0.31K	-1.53%
2581	01/03/2013	1,675.60	1,688.00	1,689.30	1,664.30	0.19K	-0.85%
2582	01/02/2013	1,689.90	1,675.80	1,695.00	1,672.10	0.06K	0.78%

2583 rows × 7 columns

شکل ۱ فرمت داده ها

۲-۱-۱ پاک سازی داده ها

تاریخ‌ها به فرمت datetime تبدیل شده‌اند.

داده‌ها براساس تاریخ مرتب‌سازی شده‌اند.

مقادیر متنی (مثل اعداد دارای ویرگول) به فرمت عددی تبدیل شده‌اند.

مقادیر خالی (NaN) حذف یا جایگزین شده‌اند.

۲-۲ تقسیم داده ها به داده های آموزشی و آزمون

از آنجایی که در داده‌های سری زمانی نمی‌توانیم روی داده‌های آینده آموزش ببینیم، نباید داده‌های سری زمانی را به‌صورت تصادفی تقسیم کنیم. در روش تقسیم سری زمانی، مجموعه تست همیشه از داده‌های بعد از مجموعه آموزش تشکیل می‌شود. ما آخرین سال داده‌ها را برای تست در نظر می‌گیریم و بقیه داده‌ها را برای آموزش استفاده می‌کنیم.

```
plt.figure(figsize=(15, 6), dpi=150)
plt.rcParams['axes.facecolor'] = 'white'
plt.rc('axes',edgecolor='white')
plt.plot(df.Date[:-test_size], df.Price[:-test_size], color='black', lw=2)
plt.plot(df.Date[-test_size:], df.Price[-test_size:], color='blue', lw=2)
plt.title('Gold Price Training and Test Sets', fontsize=15)
plt.xlabel('Date', fontsize=12)
plt.ylabel('Price', fontsize=12)
plt.legend(['Training set', 'Test set'], loc='upper left', prop={'size': 15})
plt.grid(color='white')
plt.show()
```

شکل ۲ تقسیم داده ها به دو گروه آموزش و آزمون

.

فصل سوم
بازسازی داده‌ها

۳-۱ باز سازی داده ها و *ایجاد پنجره‌ی لغزان*

استفاده از مقادیر زمانی قبلی برای پیش‌بینی مقدار زمانی بعدی پنجره‌ی لغزان (Sliding Window) نامیده می‌شود. به این روش، داده‌های سری زمانی را می‌توان به یادگیری نظارت‌شده تبدیل کرد.

برای انجام این کار، از مقادیر زمانی قبلی به‌عنوان متغیرهای ورودی (X) و از مقدار زمانی بعدی به‌عنوان متغیر خروجی (y) استفاده می‌کنیم.

تعداد مقادیر زمانی قبلی که به‌عنوان ورودی در نظر می‌گیریم، عرض پنجره (Window Width) نام دارد. در اینجا، مقدار عرض پنجره را ۶۰ تنظیم می‌کنیم. بنابراین:

X_train و X_test شامل لیست‌هایی تو در تو هستند که هر کدام ۶۰ قیمت متوالی طلا را در خود دارند.

y_train و y_test نیز شامل قیمت طلای روز بعد هستند که به هر لیست در X_train و X_test مربوط می‌شود.

```
In [27]: window_size = 60
```

Training Set:

```
In [28]: train_data = df.Price[:-test_size]
         train_data = scaler.transform(train_data.values.reshape(-1,1))
```

```
In [29]: X_train = []
         y_train = []

         for i in range(window_size, len(train_data)):
             X_train.append(train_data[i-60:i, 0])
             y_train.append(train_data[i, 0])
```

Test Set:

```
In [30]: test_data = df.Price[-test_size-60:]
         test_data = scaler.transform(test_data.values.reshape(-1,1))
```

```
In [31]: X_test = []
         y_test = []

         for i in range(window_size, len(test_data)):
             X_test.append(test_data[i-60:i, 0])
             y_test.append(test_data[i, 0])
```

شکل ۳ باز سازی داده ها و ایجاد پنجره‌ی لغزان

۳-۲ تبدیل داده به آرایه های numpy

اکنون X_train و X_test لیست‌های تو در تو (لیست‌های دو بعدی) هستند و y_train یک لیست یک‌بعدی است. برای آموزش شبکه عصبی در TensorFlow، باید آن‌ها را به آرایه‌های Numpy با ابعاد بالاتر تبدیل کنیم، زیرا این قالب داده توسط TensorFlow پذیرفته می‌شود.

```
In [32]: X_train = np.array(X_train)
         X_test  = np.array(X_test)
         y_train = np.array(y_train)
         y_test  = np.array(y_test)
```

```
In [33]: X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
         X_test  = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
         y_train = np.reshape(y_train, (-1,1))
         y_test  = np.reshape(y_test, (-1,1))
```

```
In [34]: print('X_train Shape: ', X_train.shape)
         print('y_train Shape: ', y_train.shape)
         print('X_test Shape:  ', X_test.shape)
         print('y_test Shape:  ', y_test.shape)

         X_train Shape:  (2263, 60, 1)
         y_train Shape:  (2263, 1)
         X_test Shape:   (260, 60, 1)
         y_test Shape:   (260, 1)
```

شکل ۳ **numpy**

.

فصل چهارم
ساخت شبکه ی LSTM

۱-۴ ساخت شبکه

ما یک شبکه LSTM می‌سازیم که نوعی از شبکه‌های عصبی بازگشتی (RNN) است و به‌طور خاص برای حل مشکل کم‌زرگ‌شدن گرادیان (Vanishing Gradient) طراحی شده است.

```
In [35]: def define_model():
             input1 = Input(shape=(window_size,1))
             x = LSTM(units = 64, return_sequences=True)(input1)
             x = Dropout(0.2)(x)
             x = LSTM(units = 64, return_sequences=True)(x)
             x = Dropout(0.2)(x)
             x = LSTM(units = 64)(x)
             x = Dropout(0.2)(x)
             x = Dense(32, activation='softmax')(x)
             dnn_output = Dense(1)(x)

             model = Model(inputs=input1, outputs=[dnn_output])
             model.compile(loss='mean_squared_error', optimizer='Nadam')
             model.summary()

             return model
```

مدل تعریف‌شده یک شبکه عصبی LSTM است که برای پیش‌بینی قیمت طلا از داده‌های سری زمانی طراحی شده است. این مدل شامل سه لایه LSTM است که هرکدام به‌طور مؤثر وابستگی‌های زمانی در داده‌ها را یاد می‌گیرند. برای جلوگیری از بیش‌برازش (Overfitting)، از لایه‌های Dropout با نرخ ۲۰٪ در میان لایه‌های LSTM استفاده شده است. در انتهای مدل، یک لایه Dense با ۳۲ نورون و از تابع فعال‌سازی Softmax استفاده می‌کند تا ابعاد داده‌ها کاهش یابد، و سپس یک لایه Dense با یک نورون خروجی، پیش‌بینی قیمت طلا را برای روز بعد انجام می‌دهد. این مدل با استفاده از تابع هزینه میانگین مربع خطا (MSE) و بهینه‌ساز Nadam آموزش داده می‌شود، که برای مسائل رگرسیونی مناسب است و به بهبود دقت پیش‌بینی کمک می‌کند.

۲-۴ آموزش مدل

```
model = define_model()
history = model.fit(X_train, y_train, epochs=150, batch_size=32, validation_split=0.1, verbose=1)
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 60, 1)]           0

 lstm (LSTM)                 (None, 60, 64)            16896

 dropout (Dropout)           (None, 60, 64)            0

 lstm_1 (LSTM)               (None, 60, 64)            33024

 dropout_1 (Dropout)         (None, 60, 64)            0

 lstm_2 (LSTM)               (None, 64)                33024

 dropout_2 (Dropout)         (None, 64)                0

 dense (Dense)               (None, 32)                2080

 dense_1 (Dense)             (None, 1)                 33

=================================================================
Total params: 85,057
Trainable params: 85,057
Non-trainable params: 0
```

مدل LSTM برای پیش‌بینی قیمت طلا با استفاده از داده‌های سری زمانی آموزش داده می‌شود. با استفاده از ۱۵۰ اپوک، داده‌ها آموزش داده می‌شوند و مدل پس از هر اپوک ارزیابی می‌شود تا اطمینان حاصل شود که به‌درستی در حال یادگیری است.

فصل پنجم
ارزیابی و نتایج

۵-۱ ارزیابی مدل

حال پیش‌بینی سری زمانی خود را با استفاده از متریک MAPE (میانگین درصد مطلق خطا) ارزیابی می‌کنیم.

```
In [38]: MAPE = mean_absolute_percentage_error(y_test, y_pred)
         Accuracy = 1 - MAPE
```

```
In [39]: print("Test Loss:", result)
         print("Test MAPE:", MAPE)
         print("Test Accuracy:", Accuracy)
```

```
Test Loss: 0.0009137656888924539
Test MAPE: 0.03321789869302878
Test Accuracy: 0.9667821013069712
```

شکل ٤ ارزیابی با استفاده از **MAPE**

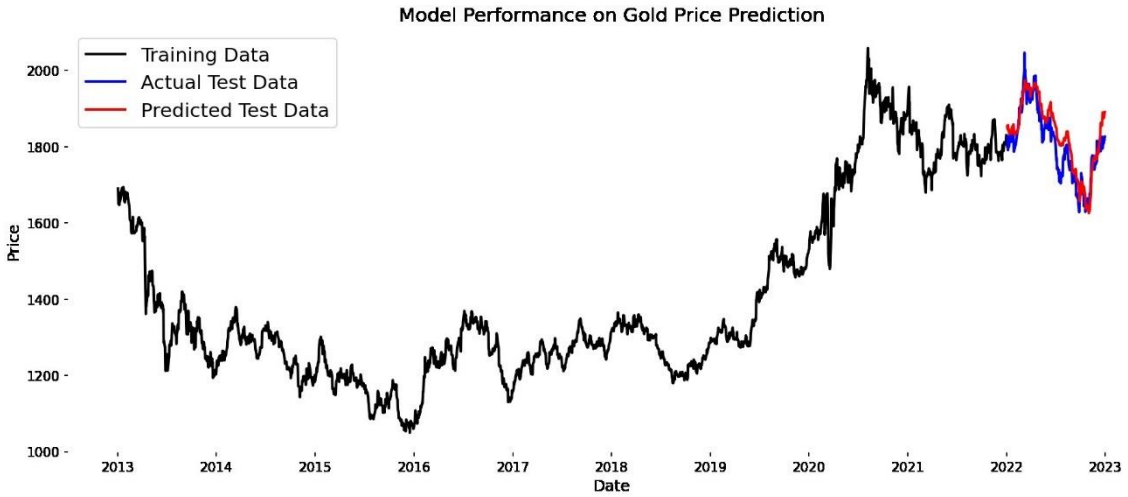
۵-۲ مصورسازی نتایج

ابتدا مقادیر قیمت واقعی و پیش بینی شده به مقیاس اولیه آنها بر می گردانیم و سپس بررسی نزدیکی قیمت های پیش بینی شده توسط مدل به قیمت های واقعی را در شکل نشان می دهیم.

```
y_test_true = scaler.inverse_transform(y_test)
y_test_pred = scaler.inverse_transform(y_pred)
```

Investigating the closeness of the prices predicted by the model to the real prices:

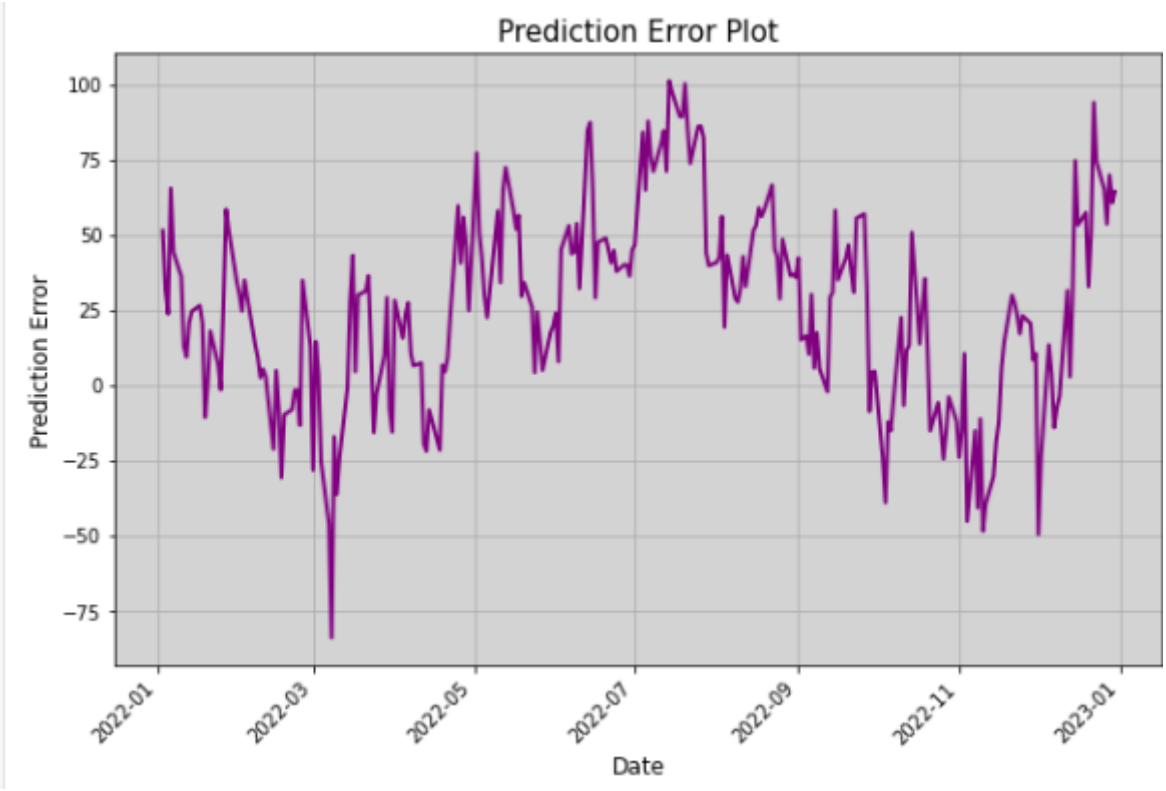
```
plt.figure(figsize=(15, 6), dpi=150)
plt.rcParams['axes.facecolor'] = 'white'
plt.rc('axes',edgecolor='white')
plt.plot(df['Date'].iloc[:-test_size], scaler.inverse_transform(train_data), color='black', lw=2)
plt.plot(df['Date'].iloc[-test_size:], y_test_true, color='blue', lw=2)
plt.plot(df['Date'].iloc[-test_size:], y_test_pred, color='red', lw=2)
plt.title('Model Performance on Gold Price Prediction', fontsize=15)
plt.xlabel('Date', fontsize=12)
plt.ylabel('Price', fontsize=12)
plt.legend(['Training Data', 'Actual Test Data', 'Predicted Test Data'], loc='upper left', prop={'size': 15})
plt.grid(color='white')
plt.show()
```



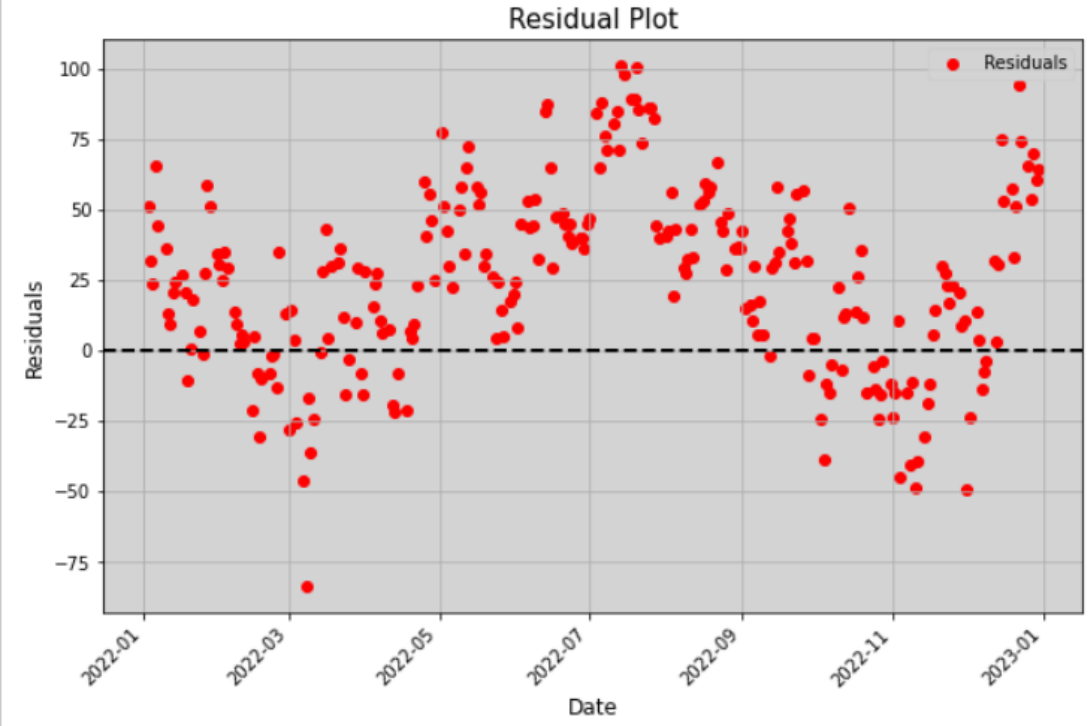
شکل ۵ نتایج پیش بینی قیمت طلا (واحد دلار)

برای ارزیابی دقت مدل پیش‌بینی، می‌توان از چندین نمودار و معیار مختلف استفاده کرد که هرکدام اطلاعات خاصی درباره عملکرد مدل به ما می‌دهند. اولین نمودار خطای پیش‌بینی است که تفاوت میان قیمت‌های پیش‌بینی‌شده و واقعی را برای هر روز نشان می‌دهد. این نمودار به ما کمک می‌کند تا متوجه شویم مدل در کجا دقیق‌تر عمل کرده و کجا دچار خطا شده است. در کنار این، نمودار باقی‌مانده (Residual Plot) خطاهای مدل را در مقابل تاریخ‌ها به‌صورت پراکنده نمایش می‌دهد. این نمودار نشان می‌دهد که آیا مدل برای همه داده‌ها به‌طور یکسان عمل می‌کند یا خیر، و اینکه آیا الگوهای خاصی در خطاها وجود دارد یا نه.

علاوه بر این، برای اندازه‌گیری دقت مدل به‌صورت عددی، از معیارهایی مانند میانگین خطای مطلق (MAE)، میانگین مربعات خطا (MSE) و ریشه میانگین مربعات خطا (RMSE) استفاده می‌شود. MAE میزان میانگین تفاوت‌های مطلق بین پیش‌بینی‌ها و مقادیر واقعی را نشان می‌دهد که هرچه کمتر باشد، مدل دقت بیشتری دارد. MSE نیز میانگین مربعات خطاها را محاسبه می‌کند که حساسیت بیشتری به خطاهای بزرگ دارد. RMSE که ریشه MSE است، به دلیل داشتن واحد مشابه با داده‌ها، به‌طور خاصی برای اندازه‌گیری خطای مدل در داده‌های عددی مفید است. در نهایت، نمودار مقایسه‌ای که پیش‌بینی‌های مدل را با داده‌های واقعی مقایسه می‌کند، به‌طور بصری نشان می‌دهد که مدل چقدر به درستی پیش‌بینی کرده و تفاوت‌های آن با واقعیت را به‌وضوح نمایش می‌دهد.

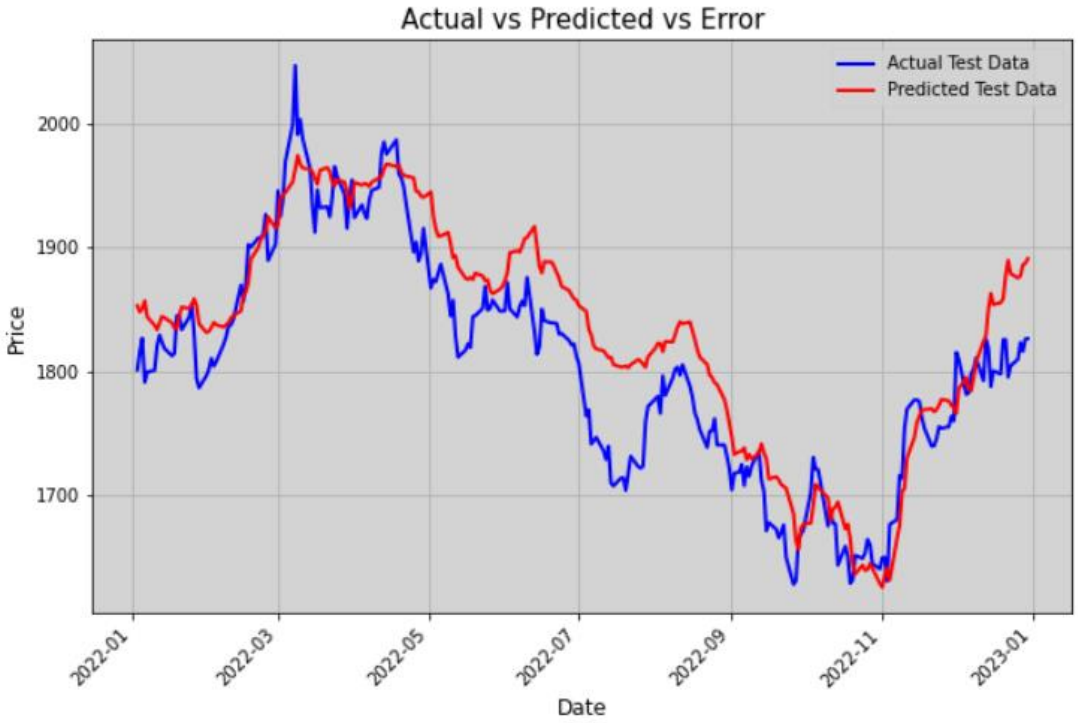


شکل ٦ نمودار خطای پیش بینی




```
Mean Absolute Error (MAE): 33.8225
Mean Squared Error (MSE): 1706.3695
Root Mean Squared Error (RMSE): 41.3082
```

شکل ۷ نمودار باقی مانده



شکل ۸ سال ۲۰۲۲ پیش بینی شده و واقعی

۳-۵ نتیجه گیری

همانطور که مشاهده می شود، قیمت پیش بینی شده توسط مدل LSTM تا حد زیادی از قیمت های واقعی پیروی می کند. مقدار اتلاف و دقت به دست آمده در داده های تست نیز عملکرد عالی مدل را تایید می کند.

Loss: 0.001 🏆

Accuracy: 96% 🏆

فصل ششم
مراجع و منابع

1. **Gold Price Prediction using LSTM**. (2023). GitHub Repository. Retrieved from https://github.com/MYoussef885/Gold_Price_Prediction
   This resource contains the complete code for predicting gold prices using LSTM networks. Historical gold price data from 2013 to 2023 was used in this project.
2. **Gold Price Prediction using Machine Learning Techniques**. (2023). Academia.edu. Retrieved from https://www.academia.edu/79395368/Gold_Price_Prediction_System
   This paper uses machine learning methods such as LSTM to predict gold prices and thoroughly explains the data preprocessing and model-building steps.
3. **Gold Price Prediction with LSTM**. (2023). Kaggle Notebook. Retrieved from https://www.kaggle.com/code/exampleuser/gold-price-prediction-lstm
   This project includes Python code for predicting gold prices using LSTM and uses historical gold price data as input.
4. **A CNN–LSTM model for gold price time-series forecasting**. (2023). Papers with Code. Retrieved from https://paperswithcode.com/paper/a-cnn-lstm-model-for-gold-price-time-series
   This paper uses a hybrid CNN-LSTM model to predict gold prices, and the results show that this model achieves high accuracy in gold price forecasting.
5. **Gold Price Prediction using Deep Learning**. (2023). GitHub Repository. Retrieved from https://github.com/Ganeshkumar2028/Gold-Price-Prediction
   This project uses deep learning methods such as LSTM and XGBoost to predict gold prices and includes steps for data collection, preprocessing, modeling, and evaluation.