BAB 2
LANDASAN TEORI

2.1 Emas

Emas adalah logam mulia yang responsif terhadap pergerakan harga. Emas memiliki keunikan dalam hal kemampuannya untuk menyimpan nilai dan melindungi risiko [2]. Emas memiliki banyak fungsi seperti sebagai alat tukar dalam melakukan perdagangan antar negara, sebagai perhiasan, sebagai nilai tukar sebuah negara, dan sebagai aset investasi. Emas sebagai aset investasi memiliki berbagai macam jenis yakni [18].

1. Emas Perhiasan.
   Berinvestasi dalam bentuk perhiasan emas memiliki kelebihan ganda, karena selain berfungsi sebagai alat investasi, perhiasan emas juga bisa dipakai sebagai aksesoris sehari-hari.

2. Emas Batangan.
   Dengan kandungan emas hingga 24 karat dan kemurnian emas hingga 99%, emas batangan merupakan investasi yang layak. Keunggulan emas batangan lainnya adalah saat inflasi meningkat, maka harga emas batangan pun ikut naik.

3. Sertifikat Emas.
   Sertifikat emas ini merupakan selembar kertas yang membuktikan kepemilikan emas yang disimpan di bank-bank suatu negara.

2.2 Time Series Data

Time series merupakan rangkaian pengamatan suatu data yang disusun dalam waktu dengan jarak yang sama. Data *time series* dikumpulkan secara rutin setiap hari, setiap minggu, atau setiap bulan dan dari data yang dikumpulkan tersebut terlihat adanya tren. Secara umum, *time series* memiliki tiga model, yaitu model tren, model siklus, dan model musiman. [19].

Teknik prediksi *time series* terbagi menjadi dua kategori. Pertama, model prediksi berbasis matematika dan statistik seperti *exponential smoothing*, *moving average*, ARIMA, dan regresi. Kedua, model prediksi berbasis kecerdasan buatan

seperti *simulated annealing*, klasifikasi, *neural network*, algoritma genetika, model hybrid, dan *genetic programming* [20].

2.3 Machine Learning

Machine learning adalah teknik pendekatan kecerdasan buatan (AI) yang dirancang untuk meniru atau menggantikan peran manusia dalam melakukan aktivitas dan memecahkan masalah. Dengan kata lain, pembelajaran mesin adalah mesin yang dibuat untuk belajar dan beroperasi tanpa instruksi apa pun dari pengguna. Secara umum, *machine learning* dibagi menjadi tiga jenis, yaitu *reinforcement learning*, *unsupervised learning*, dan *supervised learning* [21].

2.4 ARIMA

Auto Regressive Integrated Moving Average (ARIMA) adalah model yang menggunakan nilai masa lalu dan sekarang dari variabel dependen untuk menghasilkan prediksi jangka pendek yang akurat [22]. Metode ARIMA melibatkan tiga proses utama yaitu pemeriksaan diagnostik, identifikasi, dan prediksi [13, 23].

1. Proses pemeriksaan diagnostik melibatkan kontrol stasioneritas pada data *time series*, yang sangat penting untuk membuat model ARIMA yang praktis. Untuk membuat *time series* yang tidak stasioner menjadi stasioner, *differencing* (d) pada derajat yang sesuai dilakukan dan stabilitasnya diuji lagi. Proses ini terus berlanjut hingga diperoleh deret yang stasioner. Jika operasi *differencing* dilakukan beberapa kali, maka parameter integrasi dari model ARIMA ditetapkan menjadi (d).

2. Proses identifikasi dilakukan pada data yang sudah stasioner, menentukan parameter dari operasi *Auto Regressive* (AR) dan *Moving Average* (MA).

3. Prediksi dilakukan dengan menggunakan model ARIMA yang didefinisikan sebagai ARIMA (p, d, q) dengan p adalah derajat model *Auto Regressive* (AR), d adalah derajat *differencing*, dan q adalah derajat *Moving Average* (MA) [13]. Persamaan 2.1 merupakan formula matematis dari model ARIMA.

$$y_t = \alpha_1 w_{t-1} + \alpha_2 w_{t-2} + \cdots + \alpha_p w_{t-p} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q}. \quad (2.1)$$

(a) y_t : data riil yang dilinierisasi pada waktu ke-t.

(b) t : waktu.

(c) $\alpha_1, \alpha_2, ...\alpha_p$: parameter *Auto Regressive* (AR).

(d) $w_{t-1}, w_{t-2}, ...w_{t-p}$: data yang diobservasi.

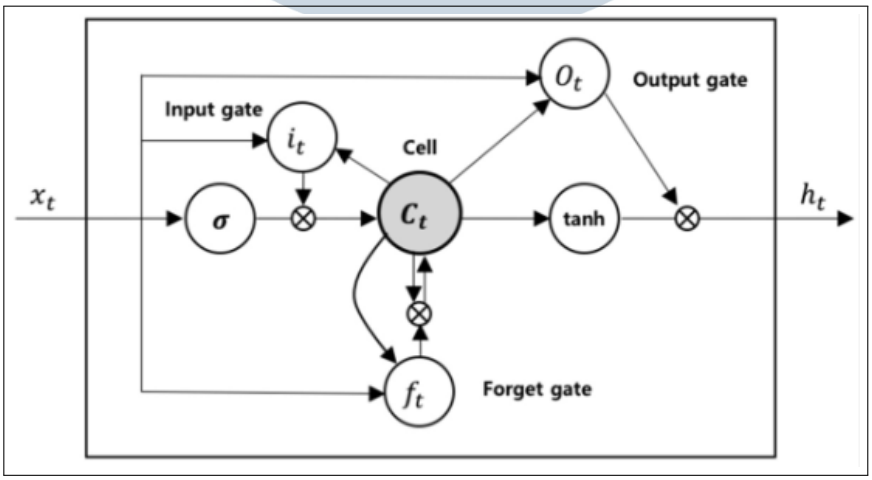
(e) ε_t : residual acak yang tidak diketahui (error).

(f) $\theta_1, \theta_2, ...\theta_q$: rata-rata bergerak (*moving average*).

(g) $\varepsilon_{t-1}, \varepsilon_{t-2}, ...\varepsilon_{t-q}$: data error.

2.5 Long Short Term Memory

Long Short Term Memory (LSTM) diusulkan oleh Hochreiter dan Schmidhuber pada tahun 1997 yang merupakan pengembangan dari *Recurrent Neural Network* (RNN). LSTM merupakan metode yang mampu menyimpan informasi dalam jangka waktu yang panjang dan secara selektif menghapus informasi yang sudah tidak relevan. Arsitektur model LSTM dapat dilihat pada Gambar 2.1 [24].



Gambar 2.1. LSTM Diagram

Struktur LSTM mencakup *cell gates* dan *memory cells*. Sebuah *cell gate* terdiri dari tiga gates yaitu *forget gate*, *input gate*, dan *output gate* [24]. *Input gate* mengatur informasi yang akan ditambahkan atau diubah dalam sel. Kemudian, *output gate* mengatur informasi yang akan dijadikan keluaran. Lalu, *forget gate* memutuskan apakah informasi dari sel sebelumnya perlu dihapus atau disimpan. Dalam memproses masukkannya, berikut langkah-langkah yang digunakan dalam LSTM [25].

1. Langkah pertama adalah menggunakan fungsi sigmoid yang dikenal sebagai *forget gate* f_t untuk mengidentifikasi data yang akan dihapus atau disimpan dalam *memory cell*. Pada langkah awal, dapat menggunakan Persamaan (2.2).

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.2}$$

Dimana :

(a) f_t : nilai dari *forget gate*.

(b) σ : fungsi sigmoid.

(c) W_f : bobot untuk nilai *input* pada waktu ke t.

(d) h_{t-1} : nilai *output* dari waktu ke $t-1$.

(e) x_t : nilai *input* pada waktu ke t.

(f) b_f : bias pada *forget gate*.

2. Langkah kedua adalah memilih data yang akan disimpan ke *cell state*. *Input gate* terdiri dari dua lapisan yaitu *input gate* dan tanh. Lapisan *input gate* i_t dapat menentukan nilai yang dapat diperbarui menggunakan fungsi aktivasi sigmoid, dan lapisan tanh C_t dapat membuat kandidat dengan nilai baru menggunakan fungsi aktivasi tanh. Pada langkah ini menggunakan Persamaan (2.3) dan (2.4).

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.3}$$

Dimana :

(a) i_t : nilai dari *input gate*.

(b) σ : fungsi sigmoid.

(c) W_i : bobot untuk nilai *input* pada waktu ke t.

(d) h_{t-1} : nilai *output* dari waktu ke $t-1$.

(e) x_t : nilai *input* pada waktu ke t.

(f) b_i : bias pada *input gate*.

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C) \tag{2.4}$$

Dimana :

(a) $\tilde{C}_t$: nilai kandidat *cell state*.

(b) tanh : fungsi hiperbolik tangent.

(c) W_c : bobot untuk nilai *input cell* ke c.

(d) h_{t-1} : nilai *output* dari waktu ke $t-1$.

(e) x_t : nilai *input* pada waktu ke t.

(f) b_i : bias pada *input gate*.

(g) b_c : bias pada bias pada cell ke c.

3. Pada langkah berikutnya, *cell state* baru dibuat dengan mengalikan persamaan (2.2) dengan *cell state* sebelumnya, yang kemudian ditambahkan dengan persamaan (2.3) dan (2.4) di langkah kedua. Pada langkah ini menggunakan Persamaan (2.5)

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.5}$$

Dimana :

(a) C_t : nilai *memory cell state*.

(b) f_t : nilai *forget gate*.

(c) C_{t-1} : nilai *memory cell state* pada *cell* sebelumnya.

(d) i_t : nilai dari *input gate*.

(e) $\tilde{C}_t$: nilai kandidat *memory cell state*.

4. Langkah terakhir adalah menentukan hasil *output*. Pertama, lapisan sigmoid memutuskan bagian dari *cell state* yang akan dijadikan *output*. Kemudian, *output* dari *cell state* tersebut dilewatkan melalui lapisan tanh dan dikalikan dengan hasil dari lapisan sigmoid untuk menghasilkan *output* akhir. Pada langkah ini dapat menggunakan Persamaan (2.6) dan (2.7) secara berurutan.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.6}$$

Dimana :

(a) o_t : nilai dari *output gate*.

(b) σ : fungsi sigmoid.

(c) W_o : bobot untuk nilai *input* pada waktu ke t.

(d) h_{t-1} : nilai *output* dari waktu ke $t-1$.

(e) x_t : nilai *input* pada waktu ke t.

(f) b_o : bias pada *output gate*.

$$h_t = o_t * \tanh(C_t) \tag{2.7}$$

Dimana :

(a) h_t : output final

(b) o_t : nilai output gate

(c) C_t : nilai memory cell state yang baru

(d) tanh : fungsi hyperbolic tangent

2.6 Hybrid ARIMA-LSTM

Hybrid ARIMA-LSTM adalah model yang mempertimbangkan keunggulan linier dan non linier. Secara khusus, model ARIMA digunakan untuk menganalisis aspek linier dari *time series*, sedangkan model LSTM menangani komponen non linier. Model hybrid sangat menguntungkan ketika model individual tidak dapat menangkap seluruh pola yang ada dalam data *time series* [26]. Model hybrid ARIMA-LSTM dibuat untuk melakukan prediksi tentang masa depan dengan menggunakan data historis dari *time series*.

Rumus prediksi *time series* dari model yang dihasilkan secara umum dapat dinyatakan sebagai jumlah dari komponen linier dan non linier, dengan Persamaan (2.8).

$$y_t = L_t + N_t \tag{2.8}$$

L_t menunjukkan komponen linier dari *time series* sedangkan N_t menunjukkan komponen non-linier. Pada model hybrid, komponen linier dari *time series* diprediksi terlebih dahulu menggunakan model ARIMA (L_t) kemudian dilanjutkan menggunakan model LSTM (N_t) [13, 23]. Model hybrid ini menggunakan model ARIMA untuk memprediksi dan mendapatkan kesalahan atau residual dari ARIMA, kemudian menggunakan model LSTM untuk memprediksi kesalahan dan setelah memprediksi kesalahan dari LSTM, dilakukan penambahan nilai prediksi kesalahan LSTM dengan prediksi ARIMA untuk mendapatkan nilai prediksi model hybrid [27].

2.7 Root Mean Square Error

Root Mean Square Error (RMSE) adalah metrik matematis yang berasal dari *Mean Square Error* (MSE) yang diperoleh selama evaluasi metode tertentu. RMSE adalah metrik yang banyak digunakan dalam pemodelan prediktif. Metrik ini berfungsi sebagai sarana untuk menilai keakuratan prediksi model dengan membandingkannya dengan nilai observasi yang sesuai. RMSE memberikan nilai numerik tunggal yang mengukur perbedaan keseluruhan antara prediksi model dan hasil aktual dengan menghitung akar kuadrat dari rata-rata perbedaan kuadrat antara nilai yang diprediksi dan nilai yang diamati. Persamaan matematis RMSE dapat menggunakan persamaan (2.9) [15].

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(Actual_i - Predicted_i)^2} \tag{2.9}$$

2.8 Mean Absolute Percentage Error

Mean Absolute Percentage Error (MAPE) merupakan nilai rata-rata selisih mutlak yang ada antara nilai prediksi dan nilai sebenarnya, yang dinyatakan sebagai persentase dari nilai sebenarnya. Menggunakan MAPE saat mengevaluasi hasil prediksi memungkinkan melihat seberapa akurat angka prediksi dan angka sebenarnya [28]. Nilai MAPE dapat dihitung dengan menggunakan persamaan (2.10) [29].

$$MAPE = \frac{100\%}{n}\Sigma_{i=1}^{n}\frac{|Actual_i - Predicted_i|}{Actual_i} \tag{2.10}$$

Penilaian MAPE dilakukan dengan melihat besarnya nilai MAPE. Nilai MAPE yang kecil menyatakan akurasi model prediksi sangat baik, sedangkan nilai MAPE yang besar menyatakan akurasi model prediksi yang buruk. Range nilai untuk MAPE dapat dilihat pada Tabel 2.1 [30].

Tabel 2.1. Akurasi nilai MAPE

MAPE	Akurasi
$<$10%	akurasi prediksi sangat baik
10-20%	akurasi prediksi baik
20-50%	akurasi prediksi memadai
$>$50%	akurasi prediksi buruk