Prediksi Harga Beras di Kota Bandung Menggunakan *LSTM* dengan Optimasi *Adam*

Kinanti Putri Kirana *, Reny Rian Marliana

Prodi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Bandung, Indonesia.

kinantiputrikirana@gmail.com, renyrianmarliana@unisba.ac.id

Abstract. Fluctuation in rice prices in Bandung City affect economic stability, particularly for low-income communities. This study aims to predict rice prices in 2025 using teh Long Short-Term Memory (LSTM) method, a Recurrent Neural Network (RNN) algorithm designed to handle time series data with long-term dependencies. The data used consists of weekly rice prices from six quality categories, spanning from September 2017 to December 2024, obtained from the National PIHPS. The model was trained for 100 epochs using Adam optimization and evaluated using R^2, MSE, MAE, and RMSE metrics. The results show that the LSTM model performed well in capturing historical patterns, with the highest R-squared value of 0.9256 achieved in the Medium I category. The predictions indicate a downward trens for lower-quality and Medium I rice, and an upqard trend for Medium II and Super I. This model has proven to be sensitie to the characteristics of each segment and is suitable for supporting food price control police.

Keywords: *LSTM, Rice Prices, Time Series, Adam Optimizer, Prediction.*

Abstrak. Fluktuasi harga beras di Kota Bandung memengaruhi stabilitas ekonomi, khususnya bagi Masyarakat berpenghasilan rendah. Penelitian ini bertujuan memprediksi harga beras pada tahun 2025 menggunakan metode *Long Short-Term Memory* (LSTM), algoritma *Recurrent Neural Network* (RNN) yang dirancang untuk menangani data *time series* dengan ketergantungan jangka panjang. Data yang digunakan adalah harga mingguan enam kategori kualitas beras dari September 2017 hingga Desember 2024 yang diperoleh dari PIHPS Nasional. Model dilatih selama 100 *epoch* menggunakan optimasi Addam, dengan evaluasi menggunakan metrik R^2, MSE, MAE, dan RMSE. Hasil menunjukkan bahwa model LSTM memiliki performa tinggi dalam menangkap pola historis, dengan nilai *R-squared* tertinggi pada kategori Medium I sebesar 0.9256. Prediksi menunjukkan tren penurunan pada beras kualitas rendah dan Medium I, serta tren naik pada Medium II dan Super I. Model ini terbukti sensitive terhadap karakteristik tiap segmen dan layak digunakan untuk mendukung kebijakan pengendalian harga pangan.

Kata Kunci: *LSTM, Harga Beras, Deret Waktu, Optimasi Adam, Prediksi.*

*kinantiputrikirana@gmail.com

A. Pendahuluan

Indonesia merupakan negara agraris dengan kekayaan sumber daya alam yang melimpah dan peran penting sektor pertanian dalam perekonomian nasional. Pada masa lampau, masyarakat di berbagai daerah memiliki beragam makanan pokok seperti jagung di Jawa Tengah dan Jawa Timur, serta sagu di wilayah timur Indonesia. Namun, seiring modernisasi, pola konsumsi masyarakat bergeser dan menjadikan beras sebagai makanan utama di hampir seluruh wilayah Indonesia (Arifin & Saliemo, 1992). Peningkatan konsumsi ini tidak selalu diimbangi oleh peningkatan produksi, sehingga menyebabkan ketergantungan pada distribusi dan menjadikan harga beras rentan terhadap fluktuasi.

Flutuasi harga beras dapat disebabkan oleh banyak faktor, seperti faktor cuaca, produksi pertanian, kebijakan impor, serta distribusi dan permintaan pasar (Malian et al., 2004). Ketidakstabilan harga ini berdampak langsung pada daya beli masyarakat, khususnya kelompok berpenghasilan rendah. Kota Bandung menjadi wilayah yang sangat relevan untuk dikaji karena merupakan pusat ekonomi Jawa Barat dengan tingkat konsumsi beras yang tinggi. Data dari BPS Kota Bandung (2023) menunjukkan bahwa beras termasuk dalam komoditas utama dalam struktur pengeluaran rumah tangga. Harga beras di Kota Bandung sering mengalami fluktuasi akibat pasokan dari daerah produsen, biaya distribusi, serta kebijakan dengan pemerintah (Kurniawan, 2024).

Dalam menghadapi ketidakpastian harga beras, peramalan menjadi salah satu langkah strategis yang dapat membantu pengambilan keputusan, baik di tingkat pemerintah, pelaku usaha, maupun konsumen. Salah satu metode yang mampu menangani data *time series* secara efektif adalah *Long Short-Term Memory* (LSTM), yang merupakan pengembangan dari *Recurrent Neural Network* (RNN). LSTM dirancang untuk mengatasi masalah *vanishing gradient* dan mampu menangkap ketergantungan jangka panjang pada data deret waktu (Julian & Pribadi, 2021). Untuk mempercepat proses pelatihan dan meningkatkan konvergensi model, digunakan algoritma optimasi Adam yang telah terbukti efisien dalam berbagai studi peramalan (Kingma & Ba, 2014).

Tujuan dari penelitian ini adalah untuk menerapkan metode LSTM dengan optimasi Adam dalam memprediksi harga beras di Kota Bandung pada tahun 2025 serta mengevaluasi tingkat akurasi model berdasarkan metrik *R-squared,* MSE, MAE, dan RMSE.

B. Metode

Penelitian ini menggunakan metode *Long Short-Term Memory* mencakup beberapa tahapan, yaitu pengumpulan data harga beras mingguan dari enam kategoru kualitas di Kota Bandung periode 2017-2024, preprocessing data untuk menyelaraskan frekuensi dan melakukan normalisasi, pembuatan model LSTM, pelatihan model selama 100 epoch menggunakan algoritma Adam, serta pengujian dan evaluasi model menggunakan metrik R^2, MSE, MAE, dan RMSE. Seluruh proses dilakukan menggunakan Python dengan bantuan pustaka *TensorFlow* dan *Keras.* Detail alur penelitian yang digunakan dapat dilihat pada Gambar 1.

Gambar 1. Plot

Data yang digunakan merupakan data sekunder yang bersumber dari Website Pusat Informasi Harga Pangan Strategis Nasional (PIHPS). Data tersebut merupakan data harga beras mingguan di Kota Bandung dari September 2017 hingga Desember 2024 dengan jumlah data 375 minggu (dalam satuan rupiah) yang dapat dilihat pada Tabel 1.

Tabel 1. Variabel penelitian

Tanggal	Beras Kualitas Bawah I	Beras Kualitas Bawah II	Beras Kualitas Medium I	Beras Kualitas Medium II	Beras Kualitas Super I	Beras Kualitas Super II
04/09/2017	10950	10400	11850	11200	12650	12000
15/09/2017	11000	10000	12650	11750	12750	11500
22/09/2017	11000	10000	12650	11750	12750	11500
⋮	⋮	⋮	⋮	⋮	⋮	⋮
18/12/2024	13750	13150	15750	14900	15900	15400
25/12/2024	13750	13150	15750	14900	15900	15400
30/12/2024	13750	13150	15750	14900	15900	15400

Long Short-Term Memory

LSTM memiliki struktur sel memori yang dilengkapi dengan tiga jenis gerbang yaitu *forget gate, input gate,* dan *output gate* yang mengatur aliran informasi melalui lapisan sigmoid dan tanh untuk menentukan informasi yang disimpan atau dilupakan (Chung & Shin, 2018). Dengan kemampuan ini, LSTM lebih efektif dalam menangani data sekuensial dibandingkan RNN konvensional (Tian et al., 2018). LSTM dirancang untuk mempertahankan gradien selama proses pelatihan, memungkinkan jaringan mempelajari pola yang lebih baik (Lasijan et al., 2023). Proses pelatihan LSTM mencakup *forward pass* untuk menghasilkan *output* prediksi secara berurutan dan *backward pass* melalui *Backpropagation Through Time* (BPTT) untuk memperbarui bobot berdasarkan *error* yang terjadi. Algoritma dalam unit LSTM dijelaskan sebagai berikut:

1. *Forget Gate* pada unit LSTM berfungsi menyaring informasi yang perlu dibuang dari memori. Dengan menerima *input* saat ini ($\boldsymbol{x}_t$) dan *hidden state* sebelumnya ($\boldsymbol{h}_{t-1}$), *gate* ini menggunakan fungsi sigmoid untuk menghasilkan *forget mask* yaitu nilai antara 0 dan 1 yang menentukan informasi mana yang dipertahankan atau di hapus dari *cell state. Output* dari *forget gate* dapat dilihat pada Persamaan (1).
$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}^{(f)} \cdot \boldsymbol{h}\boldsymbol{x}_t) \quad (1)$$
Output dari *forget gate* kemudian digunakan untuk menyaring *cell state* sebelumnya ($\boldsymbol{c}_{t-1}$) melalui perkalian elemen demi elemen, guna menentukan bagian memori yang akan dipertahankan. *Output* dari *cell state* dapat dilihat pada Persamaan (2).
$$\boldsymbol{c}_{\ddagger} = \boldsymbol{c}_{t-1} \cdot \boldsymbol{f}_t \quad (2)$$
2. *Input Gate* berfungsi menambahkan informasi baru ke dalam *cell state* setelah informasi lama disaring oleh *forget gate.* Dengan menerima *input* saat ini ($\boldsymbol{x}_t$) dan *hidden state* sebelumnya ($\boldsymbol{h}_{t-1}$), *input gate* menghasilkan dua komponen utama yaitu aktivasi sigmoid yang menentukan seberapa banyak informasi baru ditambahkan, dan aktivasi tanh yang menghasilkan kandidat nilai untuk ditambahkan ke memori. *Output* dari *input gate* dengan menggunakan fungsi sigmoid dapat dilihat pada Persamaan (3).
$$\boldsymbol{i}_t^{\dagger} = \sigma\left(\boldsymbol{W}^{(i\dagger)} \cdot \boldsymbol{h}\boldsymbol{x}_t\right) \quad (3)$$
Komponen kedua adalah aktivasi tanh yang disebut dengan *candidate cell state* untuk menentukan informasi apa yang akan ditambahkan. *Output* dari *input gate* dengan menggunakan fungsi tanh dapat dilihat pada Persamaan (4).
$$\boldsymbol{i}_t^{\ddagger} = tanh\left(\boldsymbol{W}^{(i\ddagger)} \cdot \boldsymbol{h}\boldsymbol{x}_t\right) \quad (4)$$
Setelah mendapatkan output dari fungsi sigmoid dan tanh, *input* gate akhir dihitung dengan mengalikan keduanya secara elemen demi elemen untuk menentukan informasi baru yang akan ditambahkan ke cell state. *Output* dari *input gate* dapat dilihat pada Persamaan (5).
$$\boldsymbol{i}_t = \boldsymbol{i}_t^{\dagger} \odot \boldsymbol{i}_t^{\ddagger} \quad (5)$$

3. *Update Cell State* berfungsi menyimpan dan meneruskan informasi penting antar *timestep* dengan memperbarui *cell state* sebelumnya menggunakan $(\boldsymbol{c}_{t-1})$ *output* dari *forget gate* $(\boldsymbol{f}_t)$ dan hasil perkalian *input gate* $(\boldsymbol{i}_t)$ dengan kandidat nilai baru. *Output* dari *update cell state* dapat dilihat pada Persamaan (6).

$$\boldsymbol{C}_t = \boldsymbol{c}_{\ddagger} + \boldsymbol{i}_t \tag{6}$$

4. *Output Gate* mengatur seberapa banyak informasi dari *cell state* yang ditampilkan sebagai *output,* dengan dua komponen utama: fungsi sigmoid untuk menentukan proporsi informasi yang ditampilkan dan fungsi tanh untuk mengubah *cell state* menjadi *output* yang sesuai. *Output* dari *output gate* dengan menggunakan fungsi sigmoid dapat dilihat pada Persamaan (7).

$$\boldsymbol{o}_t^{\dagger} = \sigma\left(\boldsymbol{W}^{(o\dagger)} \cdot \boldsymbol{hx}_t\right) \tag{7}$$

Fungsi aktivasi tanh digunakan untuk merepresentasikan isi *cell state* yang akan ditampilkan sebagai *output* akhir. *Output* dari *output gate* dengan menggunakan fungsi tanh dapat dilihat pada Persamaan (8).

$$\boldsymbol{o}_t^{\ddagger} = tanh\left(\boldsymbol{W}^{(o\ddagger)} \cdot \boldsymbol{hx}_t\right) \tag{8}$$

Setelah diperoleh hasil dari sigmoid dan tanh, *output gate* $(\boldsymbol{o}_t)$ dihitung dengan mengalikan keduanya secara elemen demi elemen untuk menghasilkan output akhir. *Output* dari *output gate* dapat dilihat pada Persamaan (9).

$$\boldsymbol{o}_t = \boldsymbol{o}_t^{\dagger} \odot \boldsymbol{o}_t^{\ddagger} \tag{9}$$

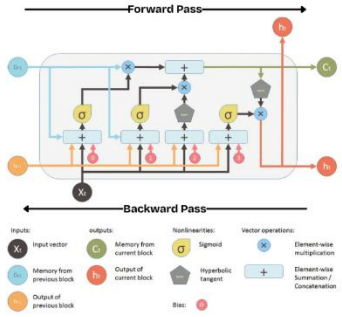
5. *Hidden State* baru $(\boldsymbol{h}_t)$ adalah *output* utama unit LSTM pada setiap *timestep*, yang diperoleh dari hasil *output gate* $(\boldsymbol{o}_t)$ dan merepresentasikan informasi terkini yang diteruskan ke *timestep* berikutnya. *Output* dari *hidden state* baru dapat dilihat pada Persamaan (10).

$$\boldsymbol{h}_{t+1} = \boldsymbol{o}_t \tag{10}$$

Setelah *hidden state* baru $(\boldsymbol{h}_t)$ dihitung, nilainya digunakan sebagai *input* ke *lapisan output (dense layer)* untuk menghasilkan prediksi pada waktu t, yaitu $\hat{\boldsymbol{x}}_t$. *Output* dari hasil prediksi dapat dilihat pada Persamaan (11).

$$\hat{\boldsymbol{x}}_t = \boldsymbol{W}_x \cdot \boldsymbol{h}_t + b_x \tag{11}$$

Arsitektur LSTM dapat dilihat pada Gambar 2.



Gambar 2. Arsitektur LSTM

Backpropagation Through an LSTM

Backpropagation adalah proses menghitung dan meneruskan *error* prediksi ke layer sebelumnya untuk memperbarui bobot jaringan (Tom Michael Mitchell, 1997). Dalam LSTM, proses ini terjadi saat *backward pass*, di mana *error* dihitung mundur ke seluruh *timestep* untuk memperbarui parameter, termasuk *cell state* dan *gate-gate* seperti *input, forget, dan output gate.* Pada *forward pass,* unit LSTM melakukan percabangan data, perkalian, dan penjumlahan elemen vektor. Agar *error* dapat dipropagasikan dengan akurat saat *backward pass,* setiap operasi ini harus ditangani secara spesifik melalui perhitunan turunan *loss function* terhadap parameter LSTM untuk memperbarui bobot (Kelleher et al., 2020)Proses *backpropagation* pada LSTM dijelaskan sebagai berikut:
Dimulai dengan menggabungkan *error* yang masuk ke dalam *output gate* menggunakan Persamaan (12).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{o}_t} = \frac{\partial \varepsilon_t}{\partial \boldsymbol{o}_t} + \frac{\partial \varepsilon_{t+1}}{\partial \boldsymbol{h}_t} \tag{12}$$

Error disebarkan melalui perkalian di *output gate* melaui Persamaan (13) dan Persamaan (14).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{o}_t^{\ddagger}} = \frac{\partial \varepsilon_t}{\partial \boldsymbol{o}_t} \odot \boldsymbol{o}_t^{\dagger} \tag{13}$$

$$\frac{\partial \varepsilon}{\partial \boldsymbol{o}_t^{\dagger}} = \frac{\partial \varepsilon_t}{\partial \boldsymbol{o}_t} \odot \boldsymbol{o}_t^{\ddagger} \tag{14}$$

Backpropagation dilakukan melalui fungsi tanh di *cell state* melalui Persamaan (15).

$$\delta_{o^{\ddagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{o}_t^{\ddagger}} \odot (\mathbf{1} - tanh^2(\boldsymbol{c}_t^{\ddagger})) \tag{15}$$

Error menuju *cell state* dihitung melalui Persamaan (16).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{c}_t} = \delta_{o\ddagger} + \frac{\partial \varepsilon_{t+1}}{\partial \boldsymbol{c}_t} \tag{16}$$

Error kemudian disebarkan ke *input gate* melalui Persamaan (17) dan Persamaan (18).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{i}_t^{\ddagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{c}_t} \odot \boldsymbol{i}_t^{\dagger} \tag{17}$$

$$\frac{\partial \varepsilon}{\partial \boldsymbol{i}_t^{\dagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{c}_t} \odot \boldsymbol{i}_t^{\ddagger} \tag{18}$$

Error disebarkan ke *cell state* sebelumnya melalui Persamaan (19).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{c}_{t-1}} = \frac{\partial \varepsilon}{\partial \boldsymbol{c}_t} \odot \boldsymbol{f}_t \tag{19}$$

Error kemudian diteruskan ke *forget gate* melalui Persamaan (20).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{f}_t} = \frac{\partial \varepsilon}{\partial \boldsymbol{c}_t} \odot \boldsymbol{c}_{t-1} \tag{20}$$

Dilakukan *backpropagation* melalui fungsi aktivasi *gate* melalui Persamaan berikut:

$$\delta_f = \frac{\partial \varepsilon}{\partial \boldsymbol{f}} \odot (\boldsymbol{f}_t \odot (\mathbf{1} - \boldsymbol{f}_t)) \tag{21}$$

$$\delta_{i^{\dagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{i}^{\dagger}} \odot (\boldsymbol{i}_t^{\dagger} \odot (\mathbf{1} - \boldsymbol{i}_t^{\dagger})) \tag{22}$$

$$\delta_{i^{\ddagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{i}^{\ddagger}} \odot (\mathbf{1} - tanh^2(\boldsymbol{i}_t^{\ddagger})) \tag{23}$$

$$\delta_{o^{\dagger}} = \frac{\partial \varepsilon}{\partial \boldsymbol{o}^{\dagger}} \odot (\boldsymbol{o}_t^{\dagger} \odot (\mathbf{1} - \boldsymbol{o}_t^{\dagger}) \tag{24}$$

Setelah semua *error* dihitung, bobot diperbaui menggunakan gradien yang dihitung melalui Persamaan (25).

$$\Delta \boldsymbol{W} = \delta \cdot \boldsymbol{h}\boldsymbol{x}^T \tag{25}$$

Pembaruan bobot dilakukan dengan optimasi Adam, gradien total terhadap hx digunakan untuk meneruskan informasi error ke *timestep* sebelumnya melalui Persamaan (26).

$$\frac{\partial \varepsilon}{\partial \boldsymbol{h}\boldsymbol{x}} = \left(\boldsymbol{W}^{(f)T} \cdot \delta_f\right) + \left(\boldsymbol{W}^{(i\dagger)T} \cdot \delta_{i\dagger}\right) + \left(\boldsymbol{W}^{(i\ddagger)T} \cdot \delta_{i\ddagger}\right) + (\boldsymbol{W}^{(o\dagger)T} \cdot \delta_{o\dagger}) \tag{26}$$

Optimasi Adam

Adam (*Adaptive Moment Estimation*) adalah algoritma optimasi yang adaptif, menyesuaikakan bobot dan *learning rate* selama pelatihan, membuat model lebih cepat dan efisien. Adam sangat cocok membatu model LSTM yang kompleks dan rawan masalah *vanishing gradient*, membantu mempercepat konvergensi dan menjaga kestabilan gradien selama *backpropagation*. Proses optimasi Adam melibatkan 4 tahapan:

1. Estimasi momen pertama (rata-rata gradien) melalui Persamaan (27).
$$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\Delta \boldsymbol{W} \tag{27}$$
2. Estimasi momen kedua (rata-rata kuadrat gradien) melalui Persamaan (28).
$$\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\Delta \boldsymbol{W}^2 \tag{28}$$
3. Koreksi bias untuk momen pertama dan momen kedua melalui Persamaan (29) dan Persamaan (30).
$$\widehat{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{1 - \beta_1^t} \tag{29}$$

$$\hat{\boldsymbol{v}}_t = \frac{\boldsymbol{v}_t}{1-\beta_2^t} \tag{30}$$

4. *Update* parameter melalui Persamaan (31).

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \alpha \frac{\hat{\boldsymbol{m}}_t}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon} \tag{31}$$

C. Hasil Penelitian dan Pembahasan

Pembagian Data *Training* dan Data *Testing*

Data observasi sebayak 375 dibagi menjadi dua bagian yaitu data *training* dan data *testing.* Data training digunakan untuk melatih model dalam mengenali pola historis harga beras, sedangkan data *testing* digunakan untuk mengevaluasi model. Rasio pembagian data yang digunakan dalam penelitian ini 80:20 seperti yang tercantum pada Tabel 2.

Tabel 2. Hasil Standardisasi Data

Keterangan	Data *Training*	Data *Testing*	Total
Persentase	80%	20%	100%
Jumlah	300	75	375

Pelatihan Model

Pelatihan model dilakukan selama 100 *epoch* untuk setiap variabel harga beras berdasarkan kualitas, dengan menggunakan fungsi *early stopping* untuk menghentikan pelatihan saat performa validasi tidak membaik, guna mencegah *overfitting*. Tujuannya adalah meminimalkan selisih antara nilai aktual dan prediksi agar model dapat mempelajari pola historis harga beras secara akurat.

Tabel 3. Hasil Pelatihan Model per Kualitas Beras

Kualitas Beras	Nilai *Loss* Akhir	Jumlah *Epoch* Pelatihan
Beras Kualitas Bawah I	0.0025	41
Beras Kualitas Bawah II	0.0028	35
Beras Kualitas Medium I	0.0026	42
Beras Kualitas Medium II	0.0019	19
Beras Kualitas Super I	0.0033	30
Beras Kualitas Super II	0.0025	38

Evaluasi Model

Setelah model dilatih dengan data training, tahap selanjutnya adalah evaluasi untuk mengukur kemampuan prediksi model terhadap harga beras, dengan membandingkan hasil prediksi dan data aktual menggunakan metrik evaluasi R^2, MSE, MAE, dan RMSE. Evaluasi dilakukan pada data yang telah didenormalisasi agar hasilnya mencerminkan nilai harga sebenarnya dalam satuan rupiah per kilogram.

Tabel 4. Hasil Evaluasi Model per Kualitas Beras

Jenis Kualitas Beras	R^2	MSE	MAE	RMSE
Bawah I	0.8557	52649.98	152.15	229.46
Bawah II	0.8057	87479.23	221.01	295.77
Medium I	0.9256	37506.56	138.87	193.67

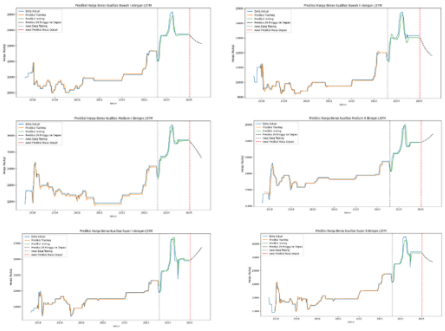
Medium II	0.8287	61227.75	139.35	247.44
Super I	0.7909	113517.70	183.20	336.92
Super II	0.8342	94875.84	190.02	308.02

Hasil Prediksi Harga Beras

Setelah melalui tahap pelatihan dan evaluasi model, dilakukan proses prediksi terhadap harga beras menggunakan metode LSTM. Model yang telah dilatih dilatih ini kemudian digunakan untuk memprediksi harga beras selama 29 periode ke depan, mulai dari Januari hingga Juli 2025. Tabel berikut menyajikan hasil prediksi harga beras untuk masing-masing jenis kualitas.

Tabel 5. Prediksi Harga Beras untuk 29 Periode Selanjutnya (dalam satuan rupiah)

No	Periode	Bawah I	Bawah II	Medium I	Medium II	Super I	Super II
1	12/01/2025	13688	13022	15694	14919	15940	15317
2	19/01/2025	13653	12974	15654	14926	15960	15287
3	26/01/2025	13621	12916	15603	14936	15981	15250
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
27	13/07/2025	13148	11845	14722	15362	16687	14720
28	20/07/2025	13140	11817	14680	15390	16725	14709
29	27/07/2025	13132	11789	14637	15419	16764	14699



Gambar 3. Grafik Prediksi Harga Beras Kualitas Bawah – Kualitas Super

Selama periode prediksi, harga beras menunjukkan tren penurunan bertahap pada sebagian besar kategori, terutama pada Super II, Medium I, Bawah I, dan Bawah II. Medium I turun dari Rp15.694 menjadi Rp14.637, sementara penurunan paling tajam terjadi di Bawah II dari Rp13.022 menjadi Rp11.789. Sebaliknya, Medium II dan Super I mengalami kenaikan, masing-masing dari Rp14.919 menjadi Rp15.419 dan Rp15.940 menjadi Rp16.764 yang mengindikasikan potensi peningkatan permintaan atau tekanan pasokan di segmen tersebut. Super II justru menurun secara bertahap. Pola ini menunjukkan dinamika pasar yang bervariasi antar segmen kualitas beras.

D. Kesimpulan

Berdasarkan analisis yang dilakukan dapat disimpulakan bahwa model *Long Short-Term Memory* (LSTM) dengan algoritma optimasi Adam mampu memprediksi harga beras di Kota Bandung secara akurat. Evaluasi performa model menunjukkan nilai R^2 tertinggi sebesar 0.9256 pada kategori Medium I. Prediksi harga awal tahun 2025 menunjukkan tren penurunan pada beras kualitas Bawah dan Medium I, serta tren kenaikan pada kategori Medium II dan Super I. Hal ini menunjukkan bahwa model LSTM mampu menangkap pola historis yang kompleks dan sensitif terhadap karakteristik tiap

kategori beras.

Daftar Pustaka

Arifin, M., & Saliemo, H. P. (1992). *POLA KONSUMSI PANGAN POKOK DI BEBERAPA PROPINSI DI INDONESIA*.

Chung, H., & Shin, K. (2018). Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction. *Sustainability*, *10*(10), 3765. https://doi.org/10.3390/su10103765

Dyar Al Falah Hilman, & Aceng Komarudin Mutaqin. (2023). Penerapan Regresi Double Poisson untuk Memprediksi Pertandingan dan Klasemen Liga 1 Indonesia. *Jurnal Riset Statistika*, 97–106. https://doi.org/10.29313/jrs.v3i2.2784

Julian, R., & Pribadi, M. R. (2021). Peramalan Harga Saham Pertambangan Pada Bursa Efek Indonesia (BEI) Menggunakan Long Short Term Memory (LSTM). *Jurnal Teknik Informatika Dan Sistem Informasi*, *8*(3). http://jurnal.mdp.ac.id

Kelleher, J. D., Mac Namee, B., & D'Arcy, A. (2020). *Fundamentals of Machine Learning for Predictive Data Analytics*.

Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. http://arxiv.org/abs/1412.6980

Kurniawan, R. (2024). *Ini Penyebab Beras Mahal di Kota Bandung*.

Lasijan, T. G., Santoso, R., & Hakim, A. R. (2023). PREDIKSI HARGA EMAS DUNIA MENGGUNAKAN METODE LONG-SHORT TERM MEMORY. *Jurnal Gaussian*, *12*(2), 287–295. https://doi.org/10.14710/j.gauss.12.2.287-295

Malian, A. H., Mardianto, S., & Ariani, M. (2004). *FAKTOR-FAKTOR YANG MEMPENGARUHI PRODUKSI, KONSUMSI DAN HARGA BERAS SERTA INFLASI BAHAN MAKANAN*.

Marasabessy, M. D., & Darwis, S. (2023). Konfidensi Premi Asuransi Gempa Menggunakan Simulasi Monte Carlo. *DataMath: Journal of Statistics and Mathematics*, *1*(1), 1–10.

Muhammad Rizq Nafisyah Alam, & Aceng Komarudin Mutaqin. (2023). Pemodelan Distribusi Poisson-Sujatha pada Data Frekuensi Klaim Asuransi Kendaraan Bermotor di Indonesia. *Jurnal Riset Statistika*, 71–78. https://doi.org/10.29313/jrs.v3i1.1944

Tian, C., Ma, J., Zhang, C., & Zhan, P. (2018). A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network. *Energies*, *11*(12). https://doi.org/10.3390/en11123493

Tom Michael Mitchell. (1997). Machine Learning textbook. In *McGraw Hill*.