BAB II
KAJIAN LITERATUR

2.1 Harga Pangan

Harga pangan merupakan nilai atau harga suatu komoditas pangan yang tidak hanya mempengaruhi keputusan konsumen, tetapi juga menjadi salah satu pendorong utama perubahan indeks harga konsumen (IHK), baik di tingkat lokal, nasional, maupun global [24], [25], [26]. Secara alami, harga pangan dapat mengalami fluktuasi dari waktu ke waktum yang sering kali disebabkan oleh faktor faktor seperti kondisi cuaca, musim panen, serta variasi permintaan masyarakat [7], [8], [9]. Sehingga efek dari fluktuasi harga pangan dari waktu ke waktu secara signifikan mempengaruhi perilaku konsumsi pangan masyarakat [27]. Dengan demikian, penetapan harga pangan sangat penting karena bertindak sebagai penghubung utama antara jaminan ketahanan pangan dan perilaku konsumsi yang ditunjukkan oleh masyarakat [28].

Dalam konteks kebijakan harga pangan, pemerintah memiliki peran strategis untuk memastikan stabilitas harga dan ketersediaan barang kebutuhan pokok. Berdasarkan Peraturan Presiden Nomor 59 Tahun 2020 Pasal 2 Ayat 6, pemerintah menetapkan kebijakan terkait Harga Acuan Pembelian di Tingkat Produsen dan Harga Acuan Penjualan di Tingkat Konsumen untuk komoditas strategis seperti kedelai, bawang merah, bawang putih, cabai rawit merah, cabai merah keriting, gula konsumsi, serta daging sapi/kerbau. Kebijakan ini bertujuan untuk menjaga keseimbangan harga di pasar, sehingga dapat melindungi produsen dari kerugian dan memastikan keterjangkauan harga bagi konsumen[29].

Selain itu, pemerintah pusat juga telah menetapkan jenis barang kebutuhan pokok yang meliputi hasil-hasil pertanian, industri, peternakan dan perikanan seperti beras, kedelai, bawang merah, gula, minyak goreng, tepung terigu, daging sapi, daging ayam ras, telur ayam ras dan ikan segar. Selanjutnya, Peraturan Menteri Perdagangan Tentang Harga Acuan Pembelian di Tingkat Produsen dan Harga Acuan Penjualan di Tingkat Konsumen, Sebagaimana Tertuang dalam Pasal 1, mendefinisikan berbagai ketentuan yang terkait dengan harga acuan sebagai berikut[30]:

1. Harga Acuan Pembelian di Tingkat Produsen adalah harga pembelian di tingkat produsen yang ditetapkan oleh Kepala Badan.
2. Harga Acuan Penjualan di Tingkat Konsumen adalah harga penjulan di tingkat konsumen yang ditetapkan oleh Kepala Badan.

3. Pelaku Usaha Pangan adalah setiap orang yang bergerak pada satu atau lebih subsistem agribisnis pangan, yaitu penyedia masukan produksi, proses produksi, pengolahan, pemasaran, perdagangan, dan penunjang.

4. Perusahaan Umum (Perum) BULOG yang selanjutnya disebut Perum BULOG adalah Badan Usaha Milik Negara, yang seluruh modalnya dimiliki negara berupa kekayaan negara yang dipisahkan dan tidak terbagi atas saham, yang menyelenggarakan usaha logistik Pangan serta usaha lainnya yang dapat menunjang tercapainya maksud dan tujuan perusahaan.

5. Badan Usaha Milik Negara di Bidang Pangan yang selanjutnya disebut BUMN Pangan adalah Badan Usaha Milik Negara yang bergerak atau berusaha di bidang Pangan baik produksi, distribusi, pemasaran, atau lainnya.

6. Badan Pangan Nasional adalah lembaga pemerintah yang berada di bawah dan bertanggung jawab kepada Presidenyang mempunyai tugas melaksanakan tugas pemerintahan di bidang pangan

7. Kepala Badan Pangan Nasional yang selanjutnya disebut Kepala Badan adalah kepala lembaga yang mempunyai tugas melaksanakan tugas pemerintahan di bidang pangan.

2.2 Forecasting

Peramalan (*Forcasting)* adalah proses memperkirakan suatu kejadian masa depan yang mungkin terjadi dengan menggunakan informasi dari data masa lalu dan data masa sekarang. Peramalan bertujuan untuk menghasilkan prediksi yang sekecil mungkin kesalahannya di masa depan demi mencapai hasil yang optimal. Peramalan yang baik adalah peramalan yang dilakukan dengan mengikuti langkah-langkah atau prosedur yang baik dan akan menentukan kualitas atau mutu dari hasil peramalan yang disusun. Pada dasarnya ada 3 langkah peramalan yang penting, yaitu [31] :

1. Menganalisa data masa lalu, tahap ini berguna untuk pola yang terjadi pada masa lalu.
2. Menentukan data yang digunakan. Metode yang baik adalah metode yang memberikan hasil ramalan yang tidak jauh berbeda dengan kenyataan yang terjadi.
3. Memproyeksikan data yang lalu dengan menggunakan metode yang digunakan, dan mempertimbangkan adanaya beberapa faktor perubahan.

Meskipun peramalan tidak selalu memberikan hasil yang pasti tetapi tujuannya adalah untuk memberikan hasil yang mendekati kenyataan seakurat mungkin [32], [33], [34], [35]. Dari segi jangka waktu peramalan biasanya dibagi menjadi tiga kelompok bagian yaitu [31], [36], [37]:

1. Peramalan jangka pendek (*Short-Term Forcast)*: biasanya digunakan untuk memprediksi peristiwa dalam rentang waktu harian, mingguan, hingga bulanan ke depan.
2. Peramalan jangka menengah (*Medium-Term Forcast)*: biasanya digunakan untuk jangka waktu dari 1 hingga 2 tahun kedepan.
3. Peramalan jangka panjang (*Long-Term Forcast)*: biasanya digunakan untuk meramalkan peristiwa yang terjadi lebih dari 2 tahun kedepan.

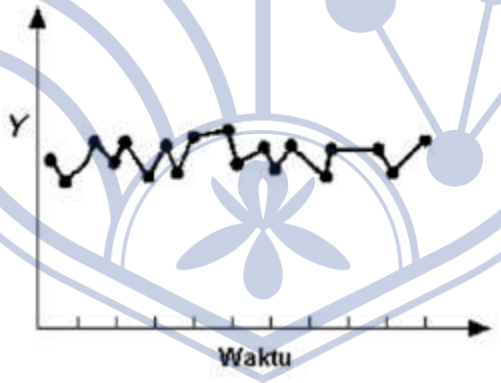
Forecasting memiliki berbagai teknik pendekatan yang dapat digunakan, tergantung pada jenis data dan tujuan dilakukannya peramalan. Salah satu pendekatan utama adalah peramalan kuantitatif. Dalam peramalan kuantitatif terdapat 2 kriteria yang dapat digunakan, yaitu pendekatan deret waktu (*time series*) dan pendekatan kausal [36].

2.2.1 Time Series

Time series (deret waktu), juga dikenal dengan data deret waktu adalah sekumpulan data yang dikumpulkan dari pengamatan atau tindakan yang dilakukan selama periode waktu tertentu. Data direkam secara dalam waktu yang konstan seperti harian, mingguan, bulanan, tahunan dan lain-lain. Secara umum terdapat empat pola dalam pola deret waktu yaitu horizontal, trend, musiman dan siklus [32], [38], [39], [40].

1. Horizontal

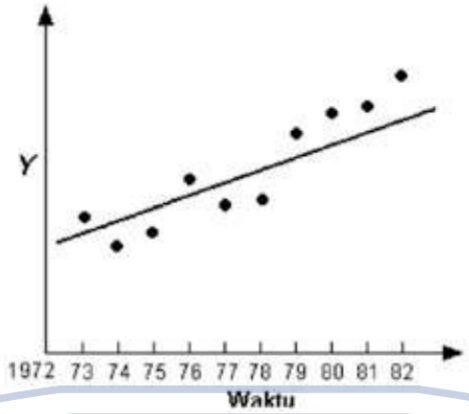
Pola data horizontal terjadi ketika data mengalami kenaikan atau penurunan disekitar rata-ratanya.



Gambar 2.1 Pola Data Horizontal [41]

2. Tren (*Trend)*

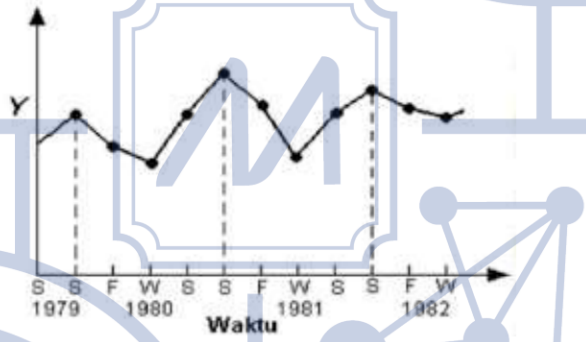
Pola data tren terjadi ketika data mengalami kenaikan atau penurunan jangan panjang dalam data.



Gambar 2.2 Pola Data Tren [41]

3. Musiman

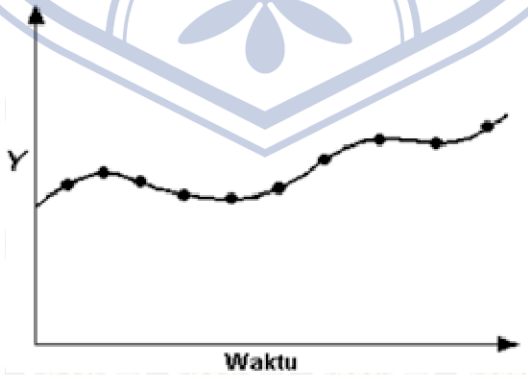
Pola data musiman terjadi ketika suatu data dipengaruhi oleh faktor musiman (misalnya kuartal tahun tertentu, bulanan, atau harian) dan berulang setiap tahun.



Gambar 2.3 Pola Data Musiman [41]

4. Siklis (*Cyclic)*

Pola siklus terjadi ketika suatu data dipengaruhi oleh naik turun dalam sebuah data dalam jangka panjang.



Gambar 2.4 Pola Data Siklis [41]

2.2.2 Metode Kausal

Pemodelan kausal adalah teknik peramalan yang menggunakan informasi tentang satu atau lebih faktor (variable) untuk memprediksi faktor lain dan mengetahui bagaimana masing-masing faktor berinteraksi satu sama lain[42].

1. Metode regresi dan korelasi: pendekatan ini diterapkan untuk meramalkan, baik dalam periode panjang maupun pendek. Ini berlandaskan pada rumus yang dianalisis secara statistik dengan menggunakan metode kuadrat kecil.
2. Pemodelan *input-output:* ini adalah teknik yang digunakan untuk ramalan jangka panjang, sering dimanfaatkan untuk menghasilkan pola ekonomi yang bertahan lama.
3. Model ekonometri: ini merupakan pendekatan peramalan yang memperkirakan sistem persamaan regresi untuk ramalan, baik jangka pendek maupun panjang. Metode ini sangat tepat dalam memberikan estimasi. Data triwulan selama beberapa tahun diperlukan untuk metode peramalan ini.

2.3 Preprocessing Data

Preprocessing data merupakan proses yang penting dilakukan sebelum melakukan proses pelatihan. Tujuan utama dari *preprocessing* data ini adalah untuk meningkatkan kualitas dan mengoptimalkan data sehingga algoritma pembelajaran dapat bekerja secara efektif dan efesien. Dengan dilakukan *preprocessing* yang tepat, maka model yang dihasilkan akan lebih akurat [43], [44], [45].

1. Data Cleaning

Data *cleaning* atau pembersihan data adalah proses memperbaiki atau menghapus kesalahan, ketidakkonsistenan, dan ketidak akuratan dalam kumpulan data. Proses ini merupakan langkah awal yang sangat penting dalam menciptakan model prediksi yang tepat. Dengan data yang telah dibersihkan, model akan lebih efektif dalam mengenali pola dan tren yang ada dalam data [46], [47], [48], [49].

2. Normalisasi Data

Normalisasi data adalah proses penyesuaian nilai data sehingga berada dalam rentang tertentu. Adapun metode dalam melakukan normalisasi data yaitu *Min-Max Normalization, Z-Score Normalization,* dan *Decimal Scaling Normalization.* Dengan memastikan bahwa data berada dalam rentang yang sesuai dapat meningkatkan akurasi model dan efisiensi proses pelatihan data [50], [51], [52]. Nilai normalisasi dengan metode *Min-Max Normalization* dapat dirumuskan ke dalam persamaan berikut:

$$x' = \frac{(x - x_{\min})}{(x_{\max} - x_{\min})} \quad (2.1)$$

dengan:

x' : data hasil normalisasi

x : data asli

x_{max} : nilai maksimum dari x

x_{min} : nilai minimum dari x

3. Segmentasi Data

Segmentasi data adalah langkah yang sangat penting dalam pengolahan data deret waktu (*time series)* untuk menciptakan struktur data yang dapat digunakan oleh model prediksi, terutama model seperti *Long Short-Term Memory* (LSTM). Tujuan dari segmentasi ini adalah mengelompokkan data masa lalu menjadi pasangan *input* dan *output* target berdasarkan urutan waktu yang penting, sehingga model dapat memahami pola waktu dari data sebelumnnya untuk meramalkan kondisi di masa depan. Secara umum, proses segmentasi dilakukan dengan menetapkan sejumlah data awal yang akan digunakan sebagai *input* (fitur) untuk memprediksi data di masa mendatang yang dijadikan sebagai sasaran (label). Jumlah data yang digunakan sebelumnya dikenal sebagai *lookback* atau ukuran jendela. Sebagai contoh, apabila periode pencarian ditentukan selama 1 hari, maka prediksi nilai hari ini dilakukan berdasarkan data dari satu hari sebelumnya. Jika periode melihat kembali adalah 3, maka model akan memanfaatkan tiga nilai sebelumya untuk memprediksi nilai yang akan datang[53], [54], [55]. Proses segmentasi dapat diilustrasikan pada Gambar 2.5 berikut:

Gambar 2.5 Segmentasi Data [54]

Pada Gambar 2.5, dilakukan pengelompokkan dengan total 30 data, maka data ke-1 sampai ke-30 merupakan bagian dari input, sedangkan data ke-31 adalah target yang ingin diprediksi. Dengan menggunakan metode ini, pemisahan data tidak hanya memfasilitasi persiapan data untuk pelatihan model, tetapi juga menjamin bahwa struktur waktu dari data tetap terjaga dan digunakan secara efektif.

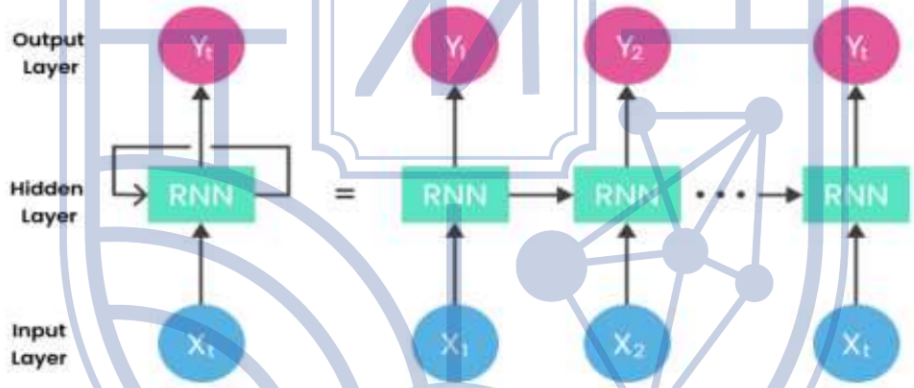
4. Pemisahan Data

Pemisahaan data dilakukan untuk memastikan model dapat belajar dari data latih dan data uji yang belum pernah dilihat sebelumnya. Hal ini bertujuan untuk menguji seberapa baik

model mampu membuat prediksi pada data baru yang tidak digunakan selama proses pelatihan [54].

2.4 Recurrent Network (RNN)

Recurrent Network (RNN) merupakan jaringan saraf berulang yaitu suatu jenis jaringan saraf tiruan yang dirancang khusus untuk mengelolah data sekuensial, sehingga dapat digunakan untuk mengelola data deret waktu. Data sekuensial mempunyai karakteristik di mana data diproses dengan suatu urutan (misalnya waktu), dan suatu sampel dalam urutan mempunya hubungan erat satu dengan yang lain. RNN memproses *input* secara sekuensial, sampel per sempel. Dalam tiap pemrosesan, *output* yang dihasilkan tidak hanya merupakan fungsi dari sampel itu saja, tapi juga berdasarkan state internal yang merupakan hasil dari pemrosesan sampel-sampel sebelumnya. Pada dasarnya, algoritma RNN beroperasi dalam empat tahap berurutan, dimana keluaran pada waktu t-1 dimasukkan ke waktu t berikutnya. Selanjutnya hasil yang diperoleh pada waktu t akan dijadikan *input* pada waktu t+1 [56], [57]. Adapun arsitektur dari RNN adalah sebagai berikut:



Gambar 2.6 Arsitektur *Recurrent Network* (RNN) [58]

Pada Gambar 2.6 merupakan diagram arsitektur pada pemrosesan RNN, dimana lingkaran biru atau yang disimbolkan dengan X_t merupakan *input layer* atau tempat masuknya data yang akan dilakukan komputasi pada RNN. Data ini kemudian diteruskan ke *hidden layer,* yang ditandai dengan kotak bewarna hijau toska. Pada *hidden layer* terjadinya pola *looping* berupa garis melingar yang kembali ke kotak hijau toska itu sendiri. Pola ini memungkinkan RNN untuk menyimpan memori sementara yang kemudian digunakan dalam pemrosesan data pada langkah berikutnya. Terakhir, terdapat *output layer* yang menjadi hasil akhir dari proses komputasi yang dilakukan oleh RNN [59].

Arsitektur RNN menggambarkan bagaimana jaringan ini beroperasi dengan melibatkan banyak layer, dimana proses perulangan terjadi pada *hidden state* sesuai dengan jumlah data

waktu yang tersedia. Berdasarkan Gambar 2.6, S_t dan Y_t dapat dipresentasikan sebagai berikut:

$$S_t = f\big((U * X_t) + (W * S_{t-1})\big) \quad (2.2)$$

$$Y_t = g(V * S_t) \quad (2.3)$$

dengan:

S_t : memori jaringan pada waktu ke-t

$X_t\ dan\ Y_t$: *input* dan *output* pada waktu ke-t

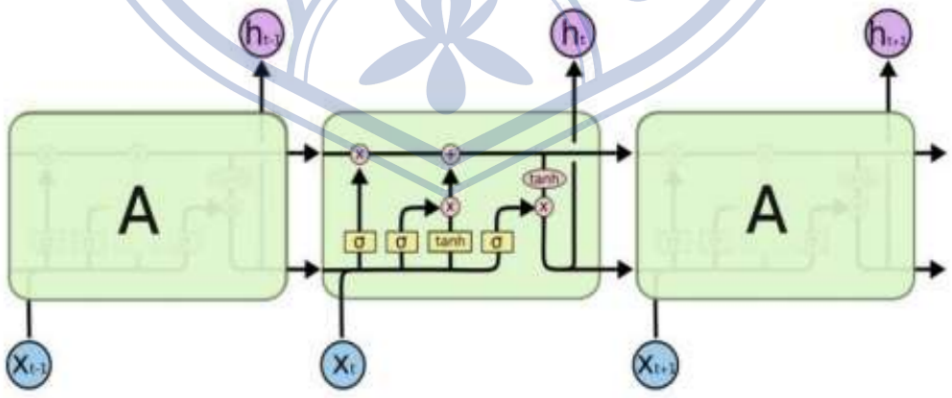
U, V, dan W : bobot setiap layer

$f(\dots)\ dan\ g(\dots)$: fungsi nonlinear

2.5 Long Short-Term Memory

Long Short-Term Memory (LSTM) adalah salah satu jenis *Recurrent Neural Network* (RNN) yang dirancang untuk mengatasi permasalahan ketergantungan jangka panjang pada data deret waktu[60]. *Long Short-Term Memory* (LSTM) pertama kali diperkenalkan oleh Sepp Hochreiter dan Jürgen Schmidhuber pada tahun 1997. Model ini menggunakan sel memori khusus yang menggantikan neuron standar di lapisan tersembunyi RNN. Elemen utama dari LSTM adalah keadaan memori yang diperbarui melalui mekanisme gerbang yang meliputi gerbang *input*, gerbang lupa *(forget gate),* dan gerbang *output*.

Pada dasarnya, setiap sel memori dalam *Long Short-Term Memory* (LSTM) memiliki tiga lapisan sigmoid dan satu lapisan tanh. Fungsi sigmoid menghasilkan *output* antara nol dan satu, yang menggambarkan seberapa besar informasi yang harus dilewatkan. Jika nilainya mendekati nol, maka informasi tersebut tidak diteruskan, sedangkan jika nilainya mendekati satu, sernua informasi dilewatkan.



Gambar 2.7 Struktur *Long Short-Term Memory* (LSTM) [61]

Dalam struktur cell *Long Short-Term Memory* (LSTM) Pada Gambar 2. 7 simbol σ melambangkan aktivitas sigmoid dan tanh melambangkan fungsi aktivitas *hyperbolic tangent.*

Garis melengkung dari x_t merepresentasikan penggabungan dari elemen x_{t-1} (*output* dari timestep sebelumnya) dan x_t (*input* timestep saat ini). Simbol $\oplus$ menunjukkan penjumlahan matriks, sedangkan simbol $\otimes$ merupakan perkalian *hadmard product*. *Hadmard product* merupakan operasi perkalian atau element- element matriks yang bersesuaian dari dua matriks dengan ukuran yang sama.

Fungsi sigmoid dapat dinyatakan dalam persamaan berikut:

$$\sigma(x)= \frac{1}{1+e^{-x}} \quad (2.4)$$

Sedangkan fungsi tanh di tulis sebagai:

$$\tanh(x) = 2\sigma(2x) - 1 \quad (2.5)$$

Untuk memperoleh *output* predksi dari model *Long Short-Term Memory* (LSTM), diperlakukan satu *layer Dense* yang dihubungkan dengan jaringan LSTM. Fungsi aktivitas dari *layer Dense* adalah fungsi aktivitasi linier. *Layer Dense* dapat dihitung dengan menggunakan rumus berikut.

$$\hat{y}i = wd\ ht + bd \quad (2.6)$$

Dengan $\hat{y}_i$ adalah prediksi data latih ke-i. W_d adalah bobot dari *layer Dense*, h_t adalah *output* dari *layer* LSTM pada timestep ke-t dan b_d adalah bias dari *layer Dense*.

Gambar 2.8 Struktur LSTM Forget Gate [61]

Forget gate bertugas menentukan informasi mana dari keadaan memori sebelumnya (C_{t-1}) yang harus dibuang atau dilanjutkan. Seperti yang ditunjukkan pada Gambar 2.8 *Input* untuk gerbang ini berasal dari keluaran sebelumnya (h_{t-1}) dan data *input* saat ini x_t.Kedua elemen ini digabungkan dan diproses oleh fungsi sigmoid, seperti pada persamaan berikut:

$$ft = \sigma\,(Wt\,.[ht-1, xt\,] + bf) \quad (2.7)$$

Keterangan:

f_t : *Forget gate*

σ : Fungsi sigmoid

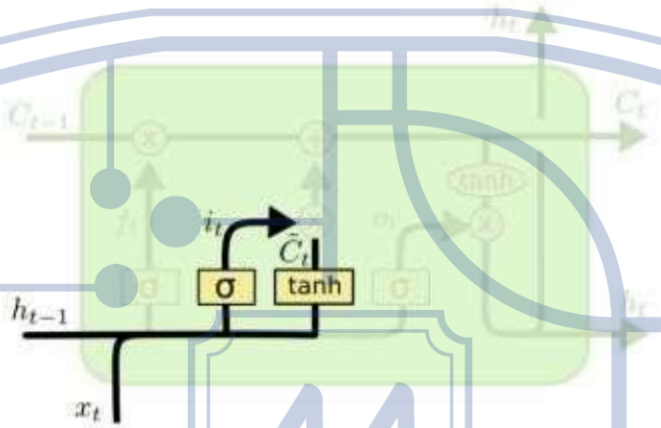
Wt : Nilai *weight* untuk *forget gate*

$h_t{-}_1$: Nilai *output* sebelum orde ke-t

x_t : Nilai *input* pada orde ke-t

bf : Nilai bias pada *forget gate*

Nilai f_t menentukan berapa banyak bagian dari (C_{t-1}) yang akan dipertahankan untuk waktu t.



Gambar 2.9 Struktur LSTM *Input Gate* [61]

Input Gate berfungsi untuk memproses *output* dari langkah sebelumnya dan *input* pada waktu saat ini melalui lapisan sigmoid. *Gate* ini menghasilkan nilai antara 0 dan 1.

$$i_t = \sigma\,(W_i.[h_t - 1, x_t] + b_i \quad (2.8)$$

Keterangan:

it : *Input gate*

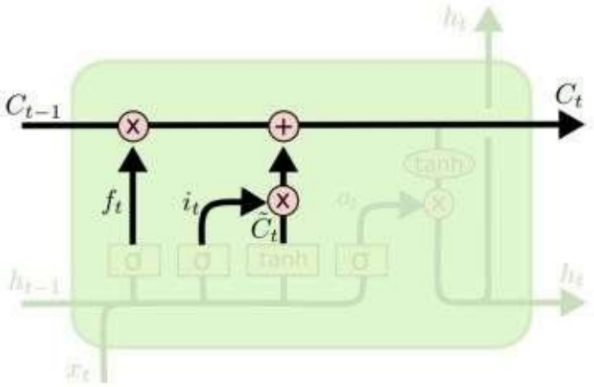
σ : Fungsi sigmoid

Wi : Nilai *weight* untuk *input gate*

$ht{-}1$: Nilai *output* sebelum orde ke-t

xt : Nilai *input* pada orde ke-t

bi : Nilai bias pada *input gate*



Gambar 2.10 Struktur LSTM *Cell State* [61]

Fungsinya adalah memperbarui infirmasi yang akan dimasukkan ke dalam keadaan sel. Vektor kandidat baru (Ct) dihasilkan melalui lapisan tanah untuk mengatur seberapa banyak informasi baru yang akan ditambahkan

$$C_t = \tanh(W_c . [h_{t-1}, x_t] + b_c) \quad (2.9)$$

Keterangan:

$\hat{C}t$: Nilai baru yang ditambahkan ke *cell state*

$Tanh$: Fungsi tanh

Wc : Nilai *weight* untuk *cell state*

$ht-1$: Nilai *output* sebelum orde ke-t

xt : Nilai *input* pada orde ke-t

bc : Nilai bias pada cell state $Ct = ft * Ct-1 + it * \hat{C}t$

Kedua hasil dari *input gate* dan *cell state* digabungkan untuk memperbarui memori

$$C_t = f_t * C_{t-1} + i_t * C_t \quad (2.10)$$

Keterangan:

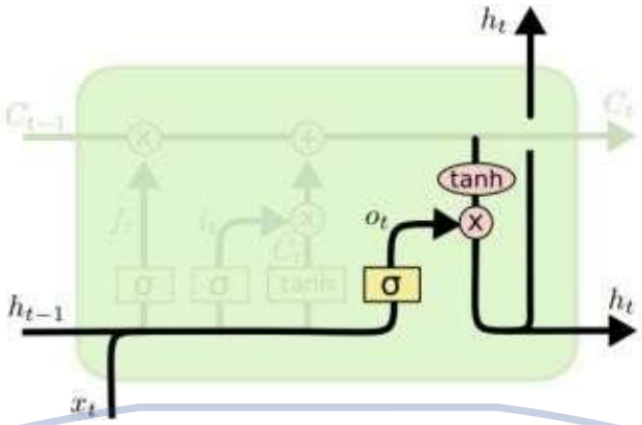
Ct : Nilai baru ditambahkan ke *cell state*

ft : *Forget gate*

$Ct-1$: *Cell state* sebelum orde ke-t

it : *Input gate*

$\hat{C}t$: Nilai baru yang ditambahkan ke *cell state*



Gambar 2.11 Struktur LSTM *Output Gate* [61]

Output gate bertugas untuk menentukan bagian informasi dari keadaan sel yang akan digunakan sebagai keluaran. Langkah pertama melibatkan aktivitas sigmoid untuk menghasilkan nilai awal, yang kemudian digabungkan dengan hasil aktivitas tanh dari keadaan sel untuk menghasilkan output akhir.

$$O_t = \sigma(W_o.[h_t\_1, x_t] + b_o) \quad (2.11)$$

Keterangan:

Ot : Output gate

σ : Fungsi sigmoid

Wo : Nilai *weight* untuk *output gate*

$ht-1$: Nilai *output* sebelum orde ke-t

xt : Nilai *input* pada orde ke-t

bo : Nilai bias pada *output gate*

Nilai *output* akhir sel didefinisikan sebagai:

$$h_t = O_t * \tanh(C_t) \quad (2.12)$$

Keterangan:

Ot : *Output gate*

σ : Fungsi sigmoid

Wo : Nilai *weight* untuk *output gate*

$ht-1$: Nilai *output* sebelum orde ke-t

xt : Nilai *input* pada orde ke-t

bo : Nilai bias pada *output gate*

Dengan ft, *i*t, $\tilde{\boldsymbol{C}}$t dan ot masing-masing adalah *forget gate, input gate,* calon kandidat *cell state*, dan *output gate* wf, w*i*, wc, dan wo masing-masing adalah matriks *weight* (bobot) input untuk *forget gate, input gate*, calon kandidat *cell state*, dan

output gate. Uf, u*i*, uc, dan uo masing-masing adalah matriks bobot input dari timestep sebelumnya untuk *forget gate*. *Input gate*, calon kandidiat *cell state*, dan *output gate*. bf, b*i*, bc, dan bo masing-masing adalah bias intuk forget gate, Input gate, calon kandidiat *cell state*, dan *output gate*. Simbol σ menunjukkan fungsi aktivasi sigmoid pada persamaan (2.5), dan tanh adalah fungsi aktivasi *hyperbolic tangent* pada persamaan (2.6).

2.6 Denormalisasi

Output yang dihasilkan dari proses prediksi masih dalam bentuk data normalisasi, sehingga perlu dilakukan konversi menggunakan denormalisasi. Denormalisasi dilakukan untuk mengembalikan nilai atau data ke bentuk semula, sehingga hasil nilai atau data dari proses prediksi dapat dengan mudah dibaca dan dimengerti [62]. Dikarenakan pada proses Normalisasi telah menggunakan *Min-Max*, seperti yang telah dijelaskan sebelumnya, maka pada Denormalisasi juga harus menggunakan *Min-Max.* Ini dikarenakan Denormalisasi merupakan kebalikan dari Normalisasi. Sehingga kedua proses ini mengacu kepada satu rentang nilai yang sama, yaitu [0, 1], yang artinya rentang nilai minimum adalah 0, dan rentang maksimum adalah 1. Adapun rumus Denormalisasi adalah sebagai berikut [63] :

$$x_t = x(X_{max} - X_{min}) + X_{min} \quad (2.13)$$

dimana:

x_t : nilai dari data normalisasi

x : hasil *output*

X_{max} : nilai *maximum* dari data actual keseluruhan

X_{min} : nilai *minimum* dari data actual keseluruhan

2.7 Adaptive Moment Estimation (Adam)

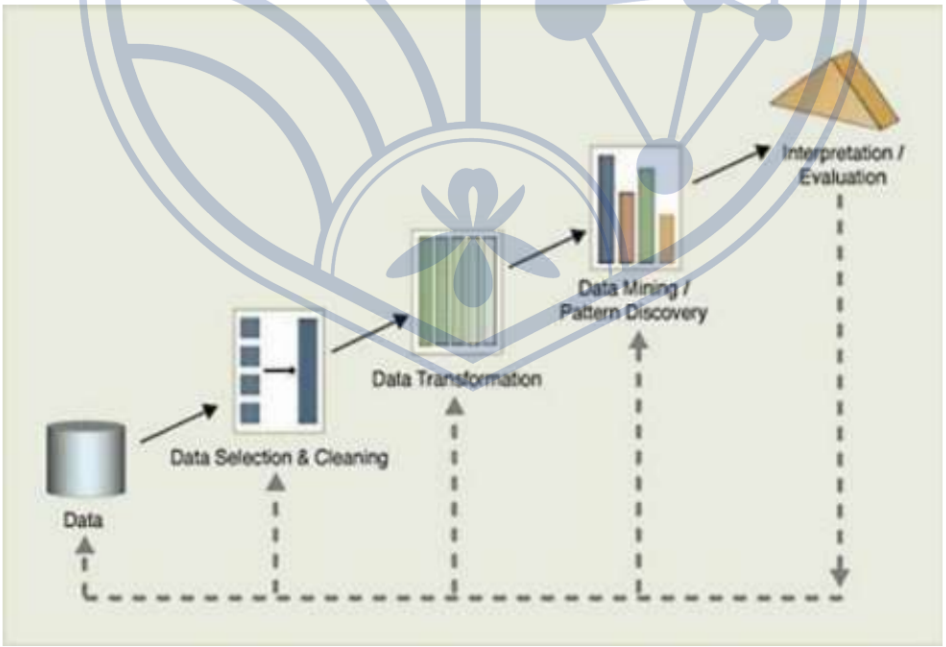
Adam (*Adaptive Moment Estimation*) merupakan algoritma optimisasi yang berlandaskan pada stochastic gradient dan sering dipakai dalam pelatihan model jaringan saraf. Adam menggabungkan keunggulan dari kedua algoritma terdahulu, yaitu AdaGrad (*Adaptive Gradient*) yang efektif untuk data dengan gradien yang jarang, dan RMSProp (*Root Mean Square Propagation*) yang lebih baik dalam pelatihan secara online serta pada data yang tidak stabil. Berbeda dengan RMSProp yang hanya menggunakan rata-rata pertama dari gradien, Adam memanfaatkan dua estimasi momen secara bersamaan: momen pertama (mean) dan momen kedua (varians tidak tersentralisasi) dari *gradien*. Perhitungan dua momen tersebut

dilakukan menggunakan pendekatan *exponential moving average* yang diatur oleh parameter β_1 dan β_2 untuk mengatur kecepatan peluruhan gradien historis.

Dalam pelaksanaannya, ada sejumlah hyperparameter krusial yang perlu ditentukan dari awal, antara lain: laju belajar (α), laju peluruhan eksponensial untuk estimasi momen pertama (β_1) dan kedua (β_2), serta konstanta kecil ϵ yang bertujuan untuk menghindari pembagian dengan nol *(zero division)* dalam proses pembaruan parameter. Keunggulan utama Adam terletak pada kemampuannya menyesuaikan laju pembelajaran secara adaptif untuk setiap parameter, sehingga pembaruan tetap stabil dan tidak terlampau bergantung pada skala gradien. Selain itu, Adam tidak membutuhkan data yang stasioner dan tidak memerlukan pengaturan parameter yang rumit [64], [65].

2.8 Knowledge Discovery in Database (KDD)

Knowledge Discovery in Database (KDD) adalah proses yang kompleks untuk menemukan dan mengenali pola-pola dalam data, dimana pola-pola tersebut harus valid, baru, berguna, dan mudah dipahami. KDD melibatkan teknik integrasi, penemuan ilmiah, interpretasi dan visualisasi pola-pola dari kumpulan data yang besar. Proses ini mencakup pengumpulan data serta penggunaan data historis untuk menentukan keteraturan, pola, atau hubungan dalam dataset yang kompleks. Hasil dari KDD, yang diperoleh melalui *data mining,* sering digunakan sebagai dasar untuk pengambilan keputusan di masa depan, menjadikannya alat penting dalam analisis data skala besar. Berikut merupakan tahapan dari KDD [66], [67] :



Gambar 2.12 Tahapan *Knowledge Discovery in Database* [68]

1. Pengumpulan data

Tahap pengumpulan data adalah langkah awal yang sangat penting dalam proses KDD. Pada tahap ini, data akan digunakan untuk analisis diambil dari berbagai sumber yang relevan, baik internal maupun eksternal. Proses pengumpulan ini bertujuan untuk memastikan bahwa data yang diperoleh mencakup informasi lengkap dan berkualitas sesuai dengan tujuan analisis.

2. Data *Selection*

Pada tahap selanjutnya dalam KDD adalah data *selection* yaitu memilih himpunan data target. Tahap ini bertujuan untuk memfokuskan analisis pada subset data tertentu, baik berupa *variable* maupun sampel data, yang dianggap relevan untuk proses penemuan pola atau informasi. Setelah data yang sesuai dipilih, data tersebut dipisahkan dari basis data operasional untuk menghindari interferensi dengan data utama. Selanjutnya, data hasil seleksi disimpan dalam berkas terpisah, sehingga memudahkan proses data mining yang akan dilakukan pada tahap berikutnya.

3. *Pre-processing / Cleaning*

Pemrosesan awal dan pembersihan data adalah langkah dasar yang mencakup penghapusan *noise*, serta proses *cleaning* yang perlu dilakukan sebelum data mining, yang melibatkan penghilangan duplikasi, pemeriksaan inkonsistensi, dan perbaikan kesalahan pada data, diikuti dengan proses *enrichment* untuk memperkaya data dengan informasi eksternal yang relevan.

4. *Transformation*

Pencarian fitur-fitur yang relevam untuk mempresentasikan data merupakan langkah penting dalam proses KDD, dimana fitur yang dipilih harus mendukung tujuan yang ingin dicapai. Selama proses ini, data yang disaring dan diubah sedemikian rupa sehingga dapat memberikan informasi yang lebih bermakna dan relevan untuk dianalisis.

5. Data Mining

Proses data mining adalah tahap di mana pola atau informasi yang menarik dicari dalam data terpolih dengan menggunakan teknik atau metode tertentu. Teknik, metode, dan algoritma yang digunakan dalam data mining sangat beragam, dan pemilihan metode yang tepat sangan bergantung pada tujuan analisis serta konteks keseluruhan proses KDD. Setiap algoritma memiliki kekuatan dan kelemahan masing-masing sehingga penting untuk memilih yang paling sesuai dengan jenis data dan masalah yang ingin diselesaikan, agar dapat menghasilkan wawasan yang akurat dan bermanfaat.

6. *Interpretation / Evaluation*

Tahap ini merupakan bagian dari proses KDD yang melibatkan pemeriksaan terhadap pola atau informasi yang ditemukan untuk memastikan bahwa hasilnya tidak bertentangan

dengan fakta atau hipotesis yang telah ada sebelumnya. Pada tahap ini, pola-pola yang diidentifikasi diuji validitasnya, apakah sesuai dengan pengetahuan yang sudah ada atau ini penting untuk memastikan bahwa informasi yang diperoleh relevan, dapat dipertanggungjawabkan dan konsisten dengan konteks yang ada.

2.9 Evaluasi Model

Menilai performa model prediksi membutuhkan alat ukur yang bisa menunjukkan seberapa mirip hasil prediksi dengan nilai sebenarnya. Dalam konteks regresi tidak diukur berdasarkan presentase keberhasilan atau kegagalan suatu prediksi seperti dalam klasifikasi, tetapi dengan mengukur ukuran kesalahan prediksi. Oleh karena itu, akurasi model regresi dinilai menggunakan metrik kesalahan seperti *Root Mean Square Error* (RMSE), MAPE, MSE, *Mean Absolute Error* (MAE), dan *R-Squared* [69], [70].

1. Root Mean Square Error (RMSE)

Root Mean Square Error mengukur akar kuadrat dari rata-rata jumlah kuadrat perbedaan antara nilai aktual dan nilai prediksi. Nilai ini dihitung dengan menjumlahkan kuadrat selisih antara nilai aktual dan prediksi, kemudian membaginya dengan jumlah data (n). Formula RMSE dirumuskan sebagai berikut [71] :

$$RMSE = \sqrt{\sum \frac{(Aktual - Prediksi)}{n}} \quad (2.14)$$

Keterangan:

Aktual : Nilai Aktual

n : Jumlah Data

Prediksi : Nilai Prediksi

2. Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentase Error adalah rata-rata absolut dari persentase kesalahan antara nilai aktual dan nilai prediksi. Metode ini digunakan untuk melihat tingkat keakuratan prediksi dalam bentuk persentase kesalahan. Rumus MAPE dinyatakan sebagai berikut [72] :

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_{i\hat{}} - y_i}{y_i} \right| \quad (2.15)$$

Keterangan:

yi^ : Nilai Prediksi

yi : Nilai Aktual

n : Jumlah Data

3. Mean Square Error (MSE)

Mean Square Error adalah metrik evaluasi yang mengukur rata-rata kuadrat dari kesalahan antara nilai aktual dan nilai prediksi. Nilai ini dihitung dengan menjumlahkan kuadrat selisih antara nilai aktual dan prediksi, kemudian membaginya dengan jumlah data (n). Rumus MSE dinyatakan sebagai berikut [71] :

$$MSE = \frac{\sum(Aktual - Prediksi)^2}{n} \quad (2.16)$$

4. Mean Absolute Error (MAE)

Mean Absolute Error adalah rata-rata dari kesalahan absolut antara nilai aktual dan nilai prediksi. Metode ini sering digunakan untuk mengukur tingkat kesalahan pada data time series [73] . Rumus *Mean Absolute Error* (MAE) dituliskan sebagai berikut:

$$MAE = \sum \frac{|Y' - Y|}{n} \quad (2.17)$$

Keterangan:

Y' : Nilai Prediksi

y : Nilai Aktual

n : Jumlah Data

5. R-Squared

R-*Squared* atau koefisien determinasi adalah ukuran statistik dalam regresi yang digunakan untuk menilai seberapa baik model menjelaskan variasi dalam data. Nilai R-*Squared* menunjukkan proporsi variabilitas dalam *variable* target yang dapat dijelaskan oleh *variable* predictor dalam model. R-*Squared* dihitung dengan membandingkan varians yang dijelaskan oleh model dengan total varians dalam data, menggunakan rumus sebagai berikut:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(yi - \hat{y}i)^2}{\sum_{i=1}^{n}(yi - \bar{y}i)^2} \quad (2.18)$$

$$y^- = \frac{\sum_{i=1}^{n} yi}{n} \quad (2.19)$$

Keterangan:

yi : Nilai Aktual

ŷi : Nilai Prediksi

ȳi : Rata-rata Nilai Aktual

Nilai R^2 berkisar antara 0 hingga 1:

R^2 mendekati 1: Model memiliki kemampuan prediksi yang baik.

R^2 mendekati 0: Model tidak mampu menjelaskan variasi data.

2.10 NFR Framework

Kebutuhan Non-Fungsional (*Non-Fungsional Requirements* atau NFR) merupakan bagian penting dalam rekayasa perangkat lunak karena berperan dalam menentukan kualitas

sistem, seperti performa, integritas, keamanan, kerahasiaan dan waktu respon. Berbeda dengan kebutuhan fungsional yang menjelaskan apa yang harus dilakukan system, NFR lebih menekankan pada bagaimana sistem tersebut harus melakukannya[74]. Namun, identifikasi dan verifikasi NFR seringkali mengahadapi tantangan seperti ambiguitas, ketidakkonsistenan, dan kurangnya definisi yang jelas. Untuk mengatasi tantangan ini, berbagai pendekatan telah dikembangkan, salah satunya adalah dengan menggunakan NFR *Framework* yang diperkenalkan oleh Chung et al. *Framework* yang diperkenalkan oleh Chung et al. *Framework* ini merepresentasikan NFR sebagai *softgoals* dalam struktur yang disebut *Softgoal Interdependency Graphs* (SIG) yang menggambarkan keterkaitan dan potensi konflik antar NFR[75].

Framework ini tidak hanya membantu dalam visualisasi dan pengelolaan NFR, tetapi juga memfasilitasi proses *reasoning* terhadap *trade-off* antar kebutuhan, yang seringkali menjadi sumber konflik dalam desain sistem perangkat lunak. Dengan menggunakan NFR *Framework*, pengembang perangkat lunak dapat lebih mudah menganalisis dan menyelesaikan konflik antar kebutuhan non-fungsional yang mungkin muncul selama perancangan sistem. Dalam perkembangan lebih lanjut, penerapan teknologi berbasis *machine learning* mulai dipertimbangkan untuk otomatisasi dalam klasifikasi dan analisis NFR. Namun, NFR *Framework* tetap menjadi pendekatan utama untuk visualisasi, analisis, dan pengelolaan kebutuhan non-fungsional, memungkinkan pengembang untuk membuat keputusan yang lebih baik dalam desain perangkat lunak[76]. Dengan menggabungkan pendekatan berbasis *framework* seperti NFR *Framework* dengan Teknik-teknik lain yang relevan, organisasi pengembangan perangkat lunak datap meningkatkan kualitas rekayasa kebutuhan mereka secara menyeluruh.

2.11 Penelitian Terdahulu

Penelitian terdahulu memiliki peran penting dalam mendukung penelitian ini, karena membantu memahami keterkaitan antara studi sebelumnya dengan studi yang sedang dilakukan. Hal ini juga memungkinkan untuk mengidentifikasi kontribusi penelitian ini terhadap perkembangan ilmu pengetahuan. Berikut disajikan beberapa penelitian serupa yang relevan dengan topik penelitian ini.

Tabel 2.1 Penelitian Terdahulu

No	Judul Penelitian	Metode	Hasil Penelitian
1	Penerapan Metode LSTM untuk Prediksi	*Long Short Term*	Penelitian ini memprediksi harga saham PT Bank

	Harga Saham PT Bank Central Asia	*Memory*	Central Asia (BCA) menggunakan metode LSTM yang dioptimalkan dengan algoritma "Adam". Data harga saham dari tahun 2020-2023 digunakan sebagai *input*. Hasil penelitian menunjukkan LSTM dapat memprediksi harga saham dengan tingkat akurasi yang baik, terlihat dari nilai RMSE (40.85), MAPE (0.71%), dan MSE (6662.76). Pendekatan ini diharapkan membantu investor dalam pengambilan keputusan berbasis data historis.
2	Prediksi Harga Jual Kakao dengan Metode LSTM	*Long Short Term Memory*	Studi ini menggunakan metode LSTM untuk memprediksi harga kakao, dengan tambahan optimasi *Root Mean Square Propagation* dan *Adaptive Moment Estimation*. Eksperimen dilakukan menggunakan kombinasi 27 *hyperparameter* untuk menghasilkan prediksi harga kakao dengan akurasi terbaik. Hasilnya menunjukkan LSTM dengan *Adaptive Moment Estimation* memberikan MSE sebesar 491505.1 dan

			MAPE sebesar 1.73%. Prediksi periode November-Desember 2021 menunjukkan tren penurunan harga.
3	Analisa Prediksi Harga Saham Sektor Perbankan Menggunakan algoritma LSTM	*Long Short Term Memory*	Pada penelitian ini didapatkan beberapa percobaan diantaranya melakukan analisis padamodel optimasi, variasai *epoch*, waktu komputasi, nilai *loss* dan akurasi, serta nilai RMSE. Variasi nilai *epoch* mempengaruhi waktu komputasi, semakin besar nilai *epoch*, maka semakin tinggi juga waktu komputasi yang dibutuhkan untuk menyelesaikan algoritma LSTM.
4	Penerapan Algoritma *Long Short Term* Memory (LSTM) Berbasis Multifungsi Aktivitas Terbobot Dalam Prediksi harga Ethereum	*Long Short Term Memory*	Penerapan algoritma *Long-Short Term Memory* dalam memprediksi harga ethereum menghasilkan nilai paling optimal dengan proporsi data *training* dan data *testing* sebesar 70:30, *sequence data* sebesar 14 yang menggambarkan banyak data dalam 2 minggu, nilai hidden unit sebesar 64, jumlah *epoch* sebesar 150, dan multi fungsi aktivasi yang digunakan adalah sigmoid.

5	Perbandingan Prediksi Penggunaan Listrik Dengan Menggunakan Metode *Long Short-Term Memory* (Lstm) Dan Recurrent Neural Network (RNN)	*Long Short Term Memory*	Berdasarkan pengujian dan analisis yang dilakukan dalam penerapan metode LSTM untuk melakukan prediksi penggunaan listrik, sehingga didapatkan kesimpulan: 1. Panjang *sequence time series* prediksi pengujian dengan rata-rata RMSE terbaik pada penggunaan panjang *sequence* 20 -30 *sequence*.