BAB I
PENDAHULUAN

1.1 Latar Belakang

Peramalan adalah salah satu metode statistik yang berperan penting dalam pengambilan keputusan. Peramalan berfungsi untuk memperkirakan apa yang akan terjadi pada masa depan berdasarkan data masa lalu. Salah satu metode yang digunakan dalam peramalan adalah metode *time series*. menyatakan bahwa pendugaan masa depan berdasarkan informasi masa lalu dari suatu variabel atau kesalahan masa lalu disebut sebagai deret berkala atau *time series*. Salah satu metode konvensional yang sering digunakan dalam peramalan deret waktu adalah metode *Autoregressive Integrated Moving Average* (ARIMA) yang diperkenalkan Box dan Jenkins (1976). Namun metode ini memiliki keterbatasan, yaitu harus memenuhi asumsi stasioner, residual bersifat *white noise* serta asumsi bahwa model yang terbentuk bersifat linier.

Saham (*stock*) merupakan salah satu instrumen pasar keuangan yang paling popular. Menerbitkan saham merupakan salah satu pilihan perusahaan ketika memutuskan untuk pendanaan perusahaan. Indeks Harga Saham Gabungan (IHSG) adalah indeks yang mengukur kinerja semua saham tercatat di Papan Utama dan Papan Pengembangan BEI (IDX, 2021). setiap saham memiliki pergerakan yang berbeda-beda dalam satu hari. Ada yang naik, turun, maupun stagnan. Jika tren IHSG sedang naik, artinya harga-harga saham di BEI juga sedang mengalami tren peningkatan. Sebaliknya, jika posisi IHSG sedang melemah, berarti harga-harga saham di BEI secara general juga sedang menurun (Arviana, Nerissa, 2023). Untuk melakukan transaksi saham para pelaku pasar modal harus memperhatikan harga saham yang ditawarkan. Karakteristik dari data finansial yang kompleks, memiliki banyak *noise*, berisifat *non linear* dan tidak stasioner membuat peramalan data saham memerlukan metode peramalan yang tepat dan akurat untuk melakukan peramalan.

Pendekatan *non-linier* sudah banyak dikembangkan menggunakan metode *artificial intelligence* diantaranya *Adaptive neural-based fuzzy inference system* (ANFIS), *Support Vector Machine* (SVM), *Genetic Programming* (GP) dan *Artificial Neural Network* (ANN). Penelitian dari Wang dkk (2009) yang membandingkan beberapa metode *artificial intelligence* untuk meramalkan data debit bulanan aliran sungai, diperoleh metode ANFIS dan SVM lebih baik dibandingkan ARMA, GP dan ANNs dengan melihat nilai *Coefficient of Correlation* (R^2), *Mean Absolute Percentage Error* (MAPE), dan *Root Mean Square Error* (RMSE) serta grafik yang mendekati nilai actual (Wang et al., 2009). Novitasari (2020) melakukan peramalan parameter cuaca sebagai variabel prediksi curah hujan menggunakan *Adaptive Neuro Fuzzy Inference System* (ANFIS) dan *Support Vector Regression* (SVR). Curah hujan

diprediksi berdasarkan data sinop seperti kelembaban relatif, angin, dan suhu. Setiap parameter harus diramalkan dengan menggunakan ANFIS dan hasilnya digunakan untuk memprediksi curah hujan menggunakan SVR. Prediksi akurat dihitung menggunakan MSE dan RMSE (Novitasari et al., 2020). Manisha Koranga (2021) melakukan penelitian menggunakan model SVM untuk memprediksi kualitas air berdasarkan parameter *Physicochemical.* Dalam penelitian ini, digunakan dua jenis fungsi kernel yaitu Fungsi Basis Radial dan Fungsi Polinomial. Hasil menunjukkan bahwa model optimal yang dikembangkan menggunakan LibSVM dengan penggunaan fungsi Kernel Polinomial memberikan akurasi sebesar 99,434% dalam memprediksi kualitas air (Koranga et al., 2021).

Salah satu metode yang digunakan dalam teknik peramalan adalah *Adaptive Neuro-Fuzzy Inference System* (ANFIS) dimana metode ini mampu menangkap pola hubungan *non-linier* dari data. *Adaptive Neuro Fuzzy Inference System* (ANFIS) adalah penggabungan mekanisme *fuzzy inference system* yang digambarkan dalam arsitektur jaringan saraf. Keunggulan *fuzzy inference system* adalah dapat menerjemahkan pengetahuan dari pakar dalam bentuk aturan-aturan, namun biasanya dibutuhkan waktu yang lama untuk menentukan fungsi keanggotaannya. Oleh sebab itu dibutuhkan teknik pembelajaran dari jaringan saraf tiruan untuk mengotomatisasi proses tersebut sehingga dapat mengurangi waktu pencarian, hal tersebut menyebabkan metode ANFIS sangat baik diterapkan dalam berbagai bidang. Peramalan menggunakan metode ANFIS dan analisis kualitas udara Kota Wuhan dampak COVID-19 pada kualitas udara. Metode ANFIS merupakan gabungan dari metode *artificial neural network* dan sistem inferensi yang baik digunakan untuk meramalkan data yang memiliki pola *non-linier* (Al-qaness et al., 2021)

Support Vector Machine (SVM) juga dapat memberikan *alternative* dalam memprediksi harga saham. Hal ini karena SVM merupakan *machine learning* yang dapat diaplikasikan dalam memprediksi data deret waktu. Metode SVM dapat menghasilkan permalan yang akurat jika memiliki kombinasi yang baik dalam menentukan parameter. Penelitian lainnya (Tarigan et al., 2021) melakukan perbandingan metode SVM dengan *Backpropogation* atau ANN dalam peramalan jumlah wisatawan mancanegara di Provinsi Bali. Hasil penelitian ini merekomendasikan peramalan paling baik menggunakan model SVM dengan fungsi kernel *radial* dengan tingkat akurasi terbaik SVM memperoleh nilai kesalahan terendah.

Salah satu faktor yang menarik minat investor adalah tingginya potensi *return* atau keuntungan yang dihasilkan jika dibandingkan bursa negara lain dalam kawasan regional. Dengan tingginya potensi tersebut, prediksi terhadap harga saham sangat diperlukan dengan tujuan mendapatkan keuntungan sebesar mungkin. Berdasarkan penelitian dari Wang (2009) yang yang membandingkan beberapa metode *artificial intelligence* dan diperoleh metode ANFIS dan SVM lebih baik dibandingkan ARMA, GP dan ANNs. Penelitian ini didukung oleh studi dari Al-

Qaness (2021) dan Tarigan (2021), yang menyatakan bahwa kedua metode, ANFIS dan SVM, cocok digunakan untuk meramalkan data dengan pola yang *nonlinear*. Metode *artificial intelligence* ini dapat diterapkan pada peramalan harga saham yang memiliki pola data yang kompleks dan tidak stasioner. Oleh karena itu, dalam penelitian ini akan mengkaji peramalan Indeks Harga Saham Gabungan (IHSG) menggunakan metode ANFIS dan SVM.

1.2 Rumusan Masalah

Berdasarkan latar belakang, berikut rumusan masalah pada penelitian ini:

1. Bagaimana model ANFIS dan SVM yang sesuai untuk meramalkan Indeks Harga Saham Gabungan?
2. Bagaimana tingkat akurasi peramalan indeks harga saham gabungan menggunakan model ANFIS dibandingkan dengan SVM?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai pada penelitian ini adalah sebagai berikut:

1. Mendapatkan model ANFIS dan SVM yang sesuai untuk meramalkan Indeks Harga Saham Gabungan.
2. Menentukan model peramalan yang lebih efektif antara ANFIS dan SVM dalam meramalkan Indeks Harga Saham Gabungan.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini sebagai berikut:

1. Menambah literatur dan pemahaman ilmiah tentang efektivitas model model ANFIS dan SVM.
2. Memudahkan investor dan pelaku bisnis saham di Indonesia dalam mengambil keputusan melalui model peramalan yang berbasis ANFIS dan SVM, yang telah diuji pada data historis IHSG.

1.5 Teori

Pada bagian ini akan dijelaskan mengenai kepustakaan yang mendukung penelitian antara lain mengenai *Adaptive Neuro Fuzzy Inference System* (ANFIS), *Support Vector Machine* (SVM), Indeks Harga Saham Gabungan (IHSG) dan studi literatur jurnal-jurnal sebelumnya yang dapat membantu penulis dalam menyelesaikan penelitian.

1.5.1 Peramalan

Peramalan adalah memperkirakan besarnya atau jumlah sesuatu pada waktu yang akan datang berdasarkan data pada masa lampau yang dianalisis secara alamiah khususnya menggunakan metode statistika. Peramalan adalah salah satu metode statistik yang berperan penting dalam pengambilan keputusan. Peramalan berfungsi untuk memperkirakan apa yang akan terjadi pada masa depan berdasarkan data masa lalu. Salah satu metode yang digunakan dalam peramalan adalah metode time

series. Pendugaan masa depan dilakukan berdasarkan informasi masa lalu dari suatu variabel atau kesalahan masa lalu ini dinamakan deret berkala atau *time series* (Utama, 2023).

Peramalan hanya dapat dilakukan apabila terdapat data dari masa lampau. Hal tersebut membuat semakin lama waktu masa depan yang diramalkan, maka semakin menurun tingkat akurasinya (Utama, 2023). Menurut horizon waktunya, peramalan dapat dibagi menjadi tiga, yaitu:

1. Peramalan jangka pendek yang memberikan hasil peramalan satu tahun mendatang atau kurang.
2. Peramalan jangka menengah untuk meramalkan keadaan satu hingga lima tahun ke depan.
3. Peramalan jangka panjang yang digunakan untuk pengambilan keputusan mengenai perencanaan produk dan perencanaan pasar, pengeluaran biaya perusahaan, studi kelayakan pabrik, anggaran, *purchase* order, perencanaan tenaga kerja dan perencanaan kapasitas kerja, serta pengambilan keputusan yang berhubungan dengan kejadian lebih dari lima tahun yang akan datang.

Adapun langkah-langkah dalam melakukan peramalan menurut (Utama, 2023) adalah sebagai berikut:

1. Definisikan tujuan peramalan.
2. Buatlah diagram pencar (plot data).
3. Memilih model peramalan yang tepat.
4. Melakukan Peramalan.
5. Hitung kesalahan ramalan (*forecast* error).
6. Mengevaluasi peramalan.

1.5.2 Pre-*Processing* Data

Pre-*processing* data adalah serangkaian langkah penting yang dilakukan sebelum data digunakan dalam analisis atau pemodelan, terutama dalam penelitian berbasis data seperti peramalan dan klasifikasi (Hedianti, 2019). Tahapan ini bertujuan untuk meningkatkan kualitas data dan mengurangi ketidaksesuaian yang dapat memengaruhi hasil analisis.. Secara umum, tahapan pre-*processing* data mencakup beberapa langkah utama berikut:Data *Cleaning*

Data yang diperoleh dari sumber eksternal sering kali mengandung data yang hilang (*missing values*), data duplikat, atau data anomali yang dapat menyebabkan bias dalam analisis. Data *cleaning* adalah langkah penting untuk mengatasi masalah-masalah tersebut, baik dengan metode penghapusan, interpolasi, maupun imputasi.

a. Pembagian Data

Pembagian data (*Data Splitting*) adalah proses memisahkan dataset menjadi beberapa bagian untuk tujuan pelatihan (*training*) dan pengujian (*testing*) model. Langkah ini bertujuan untuk mempersiapkan data sehingga model dapat dilatih dan

dievaluasi secara objektif. Pembagian data dalam penelitian ini meliputi Data *Training,* Data *Testing* dan Data Target. Data *Training* adalah bagian dari dataset yang digunakan untuk melatih model. Selama pelatihan, model menggunakan data ini untuk mempelajari pola-pola dan hubungan antara variabel *input* (fitur) dan variabel *output* (target). Model melakukan iterasi pada data training untuk memperbaiki bobot dan parameter sehingga dapat (Febriani et al., 2019).

Data *Testing* adalah bagian dari dataset yang digunakan untuk menguji kinerja model setelah pelatihan. Data ini tidak dilibatkan selama proses pelatihan model, sehingga memberikan gambaran mengenai seberapa baik model dapat menggeneralisasi pada data baru yang belum pernah dilihat. Data target adalah variabel yang ingin diprediksi oleh model, sering kali disebut sebagai variabel dependen atau variabel output. Data target disertakan dalam dataset dengan tujuan untuk menjadi acuan saat model mempelajari hubungan antara *input* dan *output* (Febriani et al., 2019).

b. Normalisasi Data

Normalisasi adalah proses dimana dapat mendekomposisi atau membagi relasi menjadi lebih dari satu relasi untuk menghilangkan anomali dalam database relasional (Kurniati et al., 2015). Normalisasi merupakan teknik yang sering diterapkan sebagai bagian dari penyiapan data untuk pembelajaran mesin. Tujuan normalisasi adalah mengubah nilai kolom numerik dalam himpunan data untuk menggunakan skala umum, tanpa mendistorsi perbedaan dalam rentang nilai atau kehilangan informasi.

Normalisasi data digunakan untuk menskalakan data suatu atribut sehingga berada dalam rentang yang lebih kecil, seperti -1 hingga 1 atau 0 hingga 1. Hal ini umumnya berguna untuk algoritma klasifikasi. Teknik normalisasi data dalam data mining sangat membantu karena memberikan banyak manfaat, sebagai berikut:

a. Penerapan algoritma data mining menjadi lebih mudah
b. Algoritma data mining menjadi lebih efektif dan efisien
c. Data dapat diekstraksi dari database dengan lebih cepat
d. Data yang sudah dinormalisasi memungkinkan untuk dianalisis dengan metode tertentu

Ada beberapa metode normalisasi data, yakni normalisasi *Z-score*, normalisasi *min-max*, dan normalisasi *decimal scaling*.

1. Normalisasi *Z-score*

Normalisasi Z-score atau dikenal dengan standarisasi merupakan teknik yang mana nilai pada atribut akan dinormalisasi berdasarkan mean dan standar deviasi. Inti dari teknik ini yakni mentransformasikan data dari nilai ke skala umum di mana angka rata-rata (*mean*) sama dengan nol dan standar deviasi adalah satu. Berikut rumus dari normalisasi *Z-score*:

$$Z = \frac{x - mean\ (x)}{stdev(x)}$$

Normalisasi Z-score dalam data mining bermanfaat untuk menganalisis data yang memerlukan perbandingan nilai dengan nilai rata-rata.

2. Normalisasi *min-max*

Metode normalisasi *min-max* mengubah sebuah kumpulan data menjadi skala mulai dari 0 (*min*) hingga 1 (*max*). Data asli mengalami modifikasi linear dalam prosedur normalisasi data ini. Nilai minimum dan maksimum dari data diambil, dan setiap nilai diubah menggunakan rumus di bawah ini:

$$Z = \frac{x - x_{min}}{[x_{max} - x_{min}]} \quad (1)$$

dimana Z adalah atribut data, x_{max} dan x_{min} adalah nilai absolut minimal dan maksimal dari x, dan x adalah nilai lama dari setiap entri dalam data.

3. Normalisasi *decimal scaling*

Pada data mining, decimal scaling merupakan cara lain untuk normalisasi. Metode ini bekerja dengan membulatkan bilangan desimal ke titik desimal terdekat. Metode ini menormalkan data dengan menggeser titik desimal dari angka. Nilai data, v dinormalisasi menjadi v' dengan menggunakan rumus di bawah ini:

$$v' = \frac{v}{10^j}$$

dimana v' adalah nilai baru setelah dilakukan penskalaan decimal, v merupakan nilai atribut dan j adalah bilangan bulat terkecil sehingga max (|v'|)<1

c. Denormalisasi Data

Pada konteks basis data, denormalisasi adalah teknik optimasi database di mana kita menambahkan data berlebihan ke satu atau lebih tabel. Tujuan utama denormalisasi adalah untuk meningkatkan kinerja query dengan mengurangi jumlah join yang diperlukan, meskipun hal ini dapat menyebabkan redundansi data (Kurniati et al., 2015). Denormalisasi dalam analisis data atau *machine learning* adalah dimana nilai hasil prediksi akan dikembalikan kedalam range sesungguhnya untuk mendapatkan nilai yang diharapkan dan dievaluasi dari model hasil evaluasi. Data asli yang dinormalisasi, dikembalikan ke data yang sebenarnya sehingga hasil perkiraan atau prediksi saat ini dapat ditemukan. Rumus denormalisasi pada range [0,1] adalah sebagai berikut.

$$x = xnorm\ (x_{max} - x_{min}) + \ x_{min} \quad (2)$$

dimana x adalah nilai data asli, xnorm adalah nilai yang telah dinormalisasi, x_{max} dan x_{min} adalah nilai maksimal dan nilai minimum dari data.

1.5.3 Adaptive Neural-Network Inference System

Adaptive Neuro Fuzzy Inference System (ANFIS) adalah penggabungan mekanisme *fuzzy inference system* yang digambarkan dalam arsitektur jaringan saraf. Keunggulan *fuzzy inference system* adalah dapat menerjemahkan pengetahuan dari pakar dalam bentuk aturan-aturan, namun biasanya dibutuhkan waktu yang lama untuk menentukan fungsi keanggotaannya. Oleh sebab itu dibutuhkan teknik pembelajaran dari jaringan saraf tiruan untuk mengotomatisasi proses tersebut sehingga dapat mengurangi waktu pencarian, hal tersebut menyebabkan metode ANFIS sangat baik diterapkan dalam berbagai bidang (Widyapratiwi et al., 2012)

Untuk memudahkan dalam menjelaskan arsitektur ANFIS di sini diasumsikan *fuzzy inference system* hanya mempunyai dua *input*, x_1 dan x_2, serta satu *output* yang dilambangkan Y. Pada model Sugeno orde satu, himpunan aturan menggunakan kombinasi *linier* dari *input-input* yang ada yang dapat diekspresikan sebagai :

$$\text{jika } x_1 = A_1 \text{ dan } x_2 = B_1, \text{maka } f_1 = p_{1x} + q_{1y} + r_1$$
$$\text{jika } x_1 = A_2 \text{ dan } x_2 = B_2, \text{maka } f_1 = p_{2x} + q_{2y} + r_2$$

dengan x dan y adalah masukan tegas pada node ke i, A_i dan B_i adalah label linguistik (rendah, sedang, tinggi, dan lain-lain) yang dinyatakan dengan fungsi keanggotaan yang sesuai, sedangkan p_i , q_i , dan r_i adalah parameter consequent (i = 1 atau 2) (Novitasari, 2020).

Data yang digunakan untuk proses pembelajaran (*training*) mencakup data masukan, parameter ANFIS, dan data uji pada periode *training* ANFIS. Proses pembelajaran menggunakan algoritma *hybrid* yang menggabungkan metode *Least-Squares Estimator* (LSE) untuk menghitung nilai consequent pada alur maju, serta *Error Backpropagation* (EBP) dan *gradient descent* pada alur mundur untuk menghitung *error* di setiap layer danmenghasilkan*output* prediksi.

1. Fungsi Keanggotaan ANFIS

Teori *fuzzy* merupakan perluasan dari teori himpunan klasik. Suatu nilai yang menunjukkan seberapa besar tingkat keanggotaan suatu elemen (x) dalam suatu himpunan (A), sering disebut dengan nilai keanggotaan atau derajat keanggotaan, dinotasikan dengan μ_A(x). Nilai keanggotaan tersebut dipetakan ke dalam suatu kurva yang disebut dengan fungsi keanggotaan (*membership function*) (widodo, 2012). Beberapa fungsi keanggotaan yang sering digunakan adalah sebagai berikut.

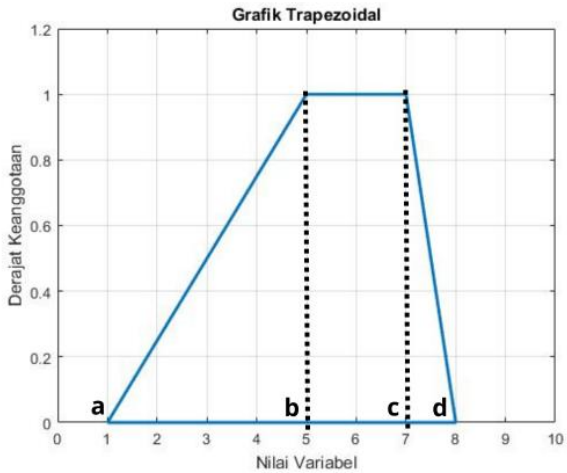
a. Fungsi Keanggotaan *Trapezoidal* yang ditentukan berdasarkan persamaan berikut

$$f(x,a,b,c,d) = \begin{cases} 0; & x \leq a \\ \dfrac{x-a}{b-a}; & a < x \leq b \\ 1; & b < x \leq c \\ \dfrac{d-x}{d-c}; & c < x \leq d \\ 0; & x > d \end{cases} \quad (3)$$

Keterangan:

x : Variabel input
a : Batas bawah (awal kenaikan kurva)
b : Awal bagian datar (*plateau*)
c : Akhir bagian datar (*plateau*)
d : Batas atas (akhir penurunan kurva)

Fungsi keanggotaan *trapezoidal* dapat divisualisasikan melalui grafik pada gambar 1. Grafik tersebut memperlihatkan bentuk kurva keanggotaan yang ditentukan oleh parameter a, b, c, dan d, yang masing-masing memengaruhi posisi awal kenaikan, awal dan akhir bagian datar, serta posisi akhir penurunan kurva.



Gambar 1. Grafik Fungsi *Trapezoidal*

Grafik fungsi keanggotaan trapezoidal dalam logika fuzzy ditentukan oleh empat parameter utama: a, b, c, dan d, yang mengatur bentuk trapesium. Parameter a menentukan titik awal dari derajat keanggotaan fuzzy di mana *input* mulai dianggap sebagai bagian dari suatu himpunan fuzzy. Semakin kecil nilai a, semakin cepat *input* mulai memiliki derajat keanggotaan yang signifikan. Parameter b menentukan kapan derajat keanggotaan mencapai nilai 1, yang berarti *input* tersebut sepenuhnya menjadi anggota suatu himpunan fuzzy. Parameter c adalah titik di mana derajat keanggotaan mulai turun dari 1, yang berarti *input* masih termasuk dalam himpunan fuzzy tetapi dengan keanggotaan

yang menurun. Semakin besar c, semakin lebar rentang *input* yang dianggap anggota penuh himpunan fuzzy, memberikan model fleksibilitas dalam menangkap variasi *input*. Parameter d menentukan titik akhir di mana nilai keanggotaan turun kembali ke 0, menunjukkan bahwa *input* tidak lagi menjadi bagian dari himpunan fuzzy.

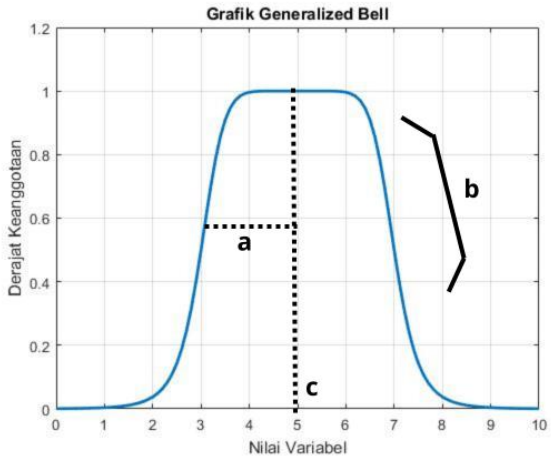
b. Fungsi Keanggotaan *Generalized Bell* yang memiliki parameter (a, b, c) yang didefinisikan sebagai berikut.

$$B(x, a, b, c) = \frac{1}{1 + \left[\left(\frac{x - c}{a}\right)^2\right]^b} \quad (4)$$

Keterangan:

x : Variabel input
a : Parameter lebar kurva (*width parameter*)
b : Parameter kelengkungan (*slope parameter*)
c : Parameter posisi pusat (*center parameter*)

Fungsi keanggotaan generalized bell dapat divisualisasikan melalui grafik pada gambar 2. Grafik tersebut memperlihatkan bentuk kurva keanggotaan dengan parameter a, b, dan c yang memengaruhi lebar, kelengkungan, dan posisi pusat kurva. Kurva ini merepresentasikan hubungan antara nilai variabel input (x) dengan derajat keanggotaan ($\mu(x)$).



Gambar. 2 Grafik Fungsi *Generalized Bell*

Pada grafik fungsi keanggotaan *generalized bell* dalam logika *fuzzy* ditentukan oleh tiga parameter nonlinear: a, b, dan c, yang mengontrol bentuk kurva lonceng. Parameter a mengatur lebar kurva, di mana semakin besar nilai a, semakin lebar lonceng tersebut. Parameter b mengontrol kelengkungan atau kemiringan sisi lonceng, dengan nilai b yang lebih besar membuat lonceng lebih landai dan nilai lebih kecil membuatnya lebih curam. Parameter c menentukan

pusat lonceng, yaitu posisi di mana nilai keanggotaan mencapai maksimum (1), menggeser lonceng ke kiri atau kanan pada sumbu horizontal. Ketiga parameter ini memberikan fleksibilitas untuk merepresentasikan berbagai derajat keanggotaan dalam himpunan *fuzzy*.

c. Fungsi Keanggotaan *Gaussian* dengan 2 parameter yaitu parameter mean (μ) dan parameter deviasi standar (σ) dengan persamaan sebagai berikut.

$$G(x, \mu, \sigma) = exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \quad (5)$$

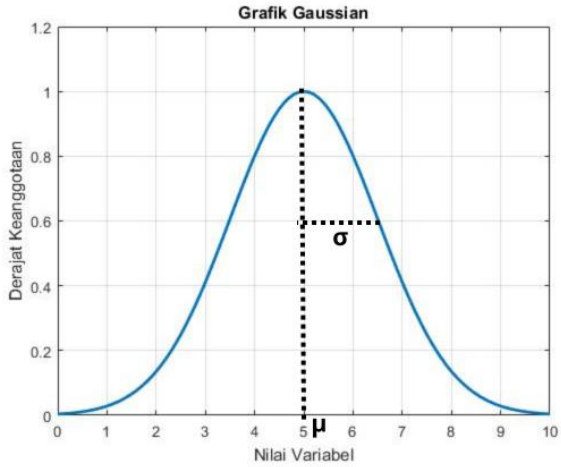
Keterangan:

x : Variabel input

μ : Parameter pusat (*mean*)

σ : Parameter penyebaran (*standar deviasi*)

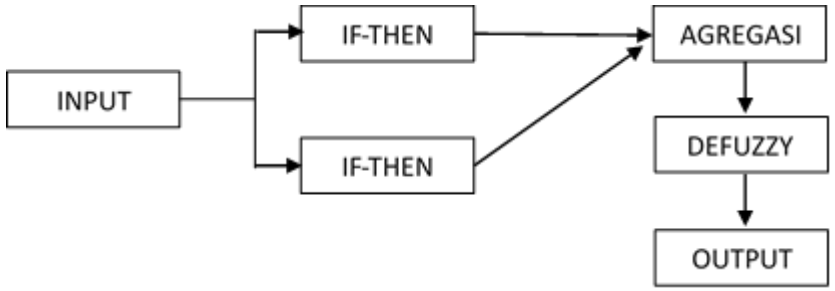
Fungsi keanggotaan Gausisian dapat divisualisasikan melalui grafik pada Gambar 3. Grafik tersebut menunjukkan kurva berbentuk lonceng yang ditentukan oleh dua parameter utama, yaitu μ dan σ.



Gambar 3. Grafik Fungsi *Gaussian*

Fungsi keanggotaan *Gaussian* dalam fuzzy logic memiliki dua parameter utama, yaitu μ dan σ. Parameter μ menentukan pusat kurva *Gaussian*, yang merupakan titik di mana derajat keanggotaan mencapai nilai maksimum. Parameter ini menggeser posisi kurva di sepanjang sumbu horizontal. Sementara itu, σ mengontrol lebar dan dispersi kurva. Semakin besar nilai σ, kurva *Gaussian* akan lebih lebar dan lebih datar, mencerminkan distribusi keanggotaan yang lebih menyebar. Sebaliknya, nilai σ yang kecil menghasilkan kurva yang lebih sempit dan tajam, dengan keanggotaan yang lebih terfokus pada nilai μ. Kombinasi kedua parameter ini menentukan bagaimana *input* dipetakan ke derajat keanggotaan fuzzy.

Sistem Inferensi *Fuzzy* (*Fuzzy Inference System*) merupakan kerangka komputasi yang didasarkan pada teori himpunan *fuzzy*, aturan *fuzzy* yang berupa IF-THEN, serta penalaran *fuzzy*. Kerangka tersebut dapat dilihat pada gambar berikut.

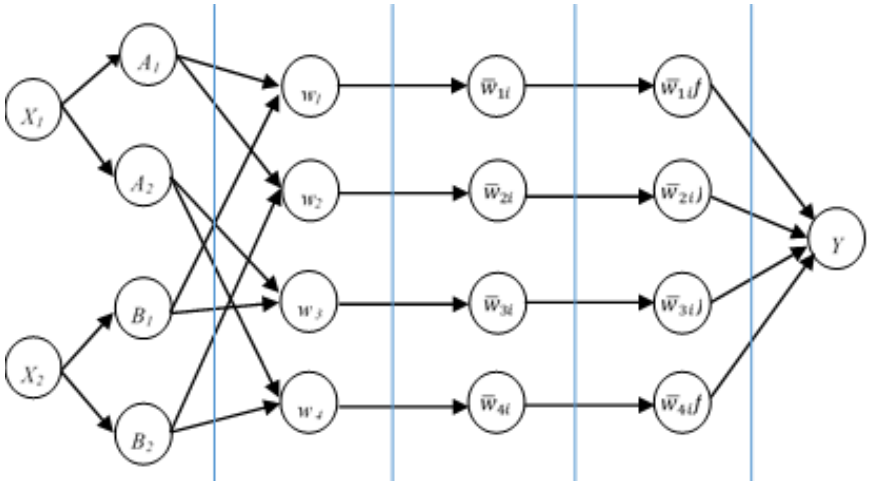


Gambar 4. Sistem Inferensi *Fuzzy*

Sistem inferensi *fuzzy* menerima *input crisp*. *Input* ini kemudian dikirim ke basis pengetahuan yang berisi jumlah aturan *fuzzy* dalam bentuk IF-THEN. *Fire strength* akan dicari pada setiap aturan. Apabila jumlah aturan lebih dari satu, maka akan dilakukan agregasi dari semua aturan. Selanjutnya, pada hasil agregasi akan dilakukan *defuzzy* untuk mendapatkan nilai *crisp* sebagai *output* sistem.

2. Arsitektur ANFIS

Arsitektur jaringan dari metode ANFIS terdiri atas 5 tahapan (lapisan) yang dapat dilihat pada Gambar 5 berikut (Roger Jang et al., 1997).



Lapisan 1 Lapisan 2 Lapisan 3 Lapisan 4 Lapisan 5

Gambar 5. Arsitektur ANFIS

Berdasarkan Gambar 5 arsitektur jaringan dari metode ANFIS terbagi atas 5 lapisan. Berikut adalah penjelasan tentang arsitektur ANFIS pada setiap lapisan:

a. Lapisan 1 (*Fuzzifikasi*)

Setiap *neuron* pada lapisan pertama adaptif terhadap parameter suatu fungsi aktivasi. *Output* dari setiap *neuron* berupa derajat keanggotaan yang diberikan oleh fungsi keanggotaan *input*. Derajat keanggotaan adalah ukuran atau nilai yang menunjukkan seberapa kuat suatu elemen atau variabel termasuk dalam suatu himpunan fuzzy. Nilai derajat keanggotaan $(\mu(x))$ berada dalam rentang 0 hingga 1, di mana 0 berarti elemen tersebut tidak termasuk dalam himpunan fuzzy sama sekali, sementara 1 menunjukkan bahwa elemen tersebut sepenuhnya termasuk dalam himpunan fuzzy. Nilai di antara 0 dan 1 (misalnya 0.5) menggambarkan bahwa elemen tersebut sebagian termasuk dalam himpunan, dengan derajat keanggotaan yang mencerminkan seberapa kuat keterlibatannya dalam himpunan fuzzy tersebut.

Pada tahap ini digunakan parameter nonlinier pada fungsi keanggotaan yang digunakan. Parameter pada lapisan ini sering disebut dengan *premise parameters*. Misalkan $x_1 = X_1$ dan $x_2 = X_2$, maka fungsi node dapat dijabarkan dengan persamaan

$$\begin{aligned} O_{1,1} &= \mu_{A_1}(X_1) \\ O_{1,2} &= \mu_{A_2}(X_1) \\ O_{1,3} &= \mu_{B_1}(X_2) \\ O_{1,4} &= \mu_{B_2}(X_2) \end{aligned} \tag{6}$$

b. Lapisan 2 (Operasi Logika *Fuzzy*)

Setiap *neuron* pada lapisan kedua berupa neuron tetap yang *output*nya adalah hasil dari masukan. Fungsi node dapat dijabarkan dengan persamaan berikut

$$O_{2,1} = w_i = \mu_{A_i}(X_1)\mu_{B_i}(X_2) \tag{7}$$

c. Lapisan 3 (*Normalized Firing Strength*)

Neuron pada lapisan ketiga berupa node tetap yang merupakan hasil perhitungan rasio dari pembobot (w_i) yang sering disebut *normalized firing strength. Firing strength* adalah ukuran seberapa kuat sebuah aturan fuzzy diaktifkan dalam sebuah sistem fuzzy. Nilai ini dihitung dari derajat keanggotaan setiap *input* yang terkait dengan aturan tersebut, menggunakan operasi seperti minimum atau perkalian. Semakin tinggi nilai firing strength, semakin besar pengaruh aturan tersebut terhadap hasil akhir sistem. Misalnya, jika suatu aturan berbunyi "Jika x adalah A dan y adalah B, maka z adalah C" maka *firing strength* dihitung berdasarkan derajat keanggotaan x dalam A dan y dalam B. Nilai ini kemudian digunakan dalam proses agregasi untuk menentukan *output* akhir dari sistem fuzzy (Muzani et al., 2022).

$$O_{3,i} = \bar{w}_i = \frac{w_i}{\sum_i w_i} \tag{8}$$

d. Lapisan 4 (*Defuzzifikasi*)

Neuron pada lapisan keempat merupakan node adaktif terhadap *output*. Parameter pada lapisan ini dinamakan *consequent parameters*

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i(c_{i1}x_1 + c_{i2}x_2 + c_{i0}) \tag{9}$$

e. Lapisan 5 (Perhitungan *Output*)

Setiap *neuron* pada lapisan kelima adalah node tetap yang merupakan jumlahan dari semua masukan dari lapisan 4, yaitu

$$O_5 = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{10}$$

Parameter linier (*consequent parameter*) pada lapisan keempat diestimasi menggunakan metode *Least Square Error* (LSE) *Recursive*. Metode ini menerapkan iterasi dengan menggunakan persamaan

$$\begin{aligned} \boldsymbol{P}^{(h+1)} &= \boldsymbol{P}^{(h)} - \frac{\boldsymbol{P}^{(h)}\boldsymbol{a}^{(h+1)}\boldsymbol{a}'^{(h+1)}\boldsymbol{P}^{(h)}}{1 + \boldsymbol{a}^{(h+1)}\boldsymbol{a}'^{(h+1)}\boldsymbol{P}^{(h)}} \\ \boldsymbol{c}^{(h+1)} &= \boldsymbol{c}^{(h)} + \boldsymbol{P}^{(h+1)}\boldsymbol{a}^{(h+1)}(\boldsymbol{a}^{(h+1)} - \boldsymbol{a}'^{(h+1)}\boldsymbol{c}^{(h)}) \end{aligned} \tag{11}$$

dengan nilai inisial untuk $\boldsymbol{P}^{(h)}$ dan $\boldsymbol{c}^{(h)}$ adalah

$$\begin{aligned} \boldsymbol{P}^{(h)} &= \left(\boldsymbol{A}'^{(h)}\boldsymbol{A}^{(h)}\right)^{-1} \\ \boldsymbol{c}^{(h)} &= \boldsymbol{P}^{(h)}\boldsymbol{A}'^{(h)}\boldsymbol{Y}^{(h)} \end{aligned} \tag{12}$$

$$\boldsymbol{A} = \begin{bmatrix} \overline{w}_{i,1}X_{1,1} & \overline{w}_{i,1}X_{1,i} & \overline{w}_{i,1} \\ \overline{w}_{i,2}X_{1,2} & \overline{w}_{i,2}X_{2,i} & \overline{w}_{i,2} \\ \vdots & \vdots & \vdots \\ \overline{w}_{i,n}X_{1,n} & \overline{w}_{i,1}X_{2,n} & \overline{w}_{i,n} \end{bmatrix} \quad \boldsymbol{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$$

Setelah didapatkan parameter linear, maka dapat dihitung nilai *output* pada lapisan 5. Parameter nonlinier (*premise parameter*) diestimasi menggunakan metode *Backpropagation Error. Error signal* yang didapatkan pada lapisan 5 akan berjalan mundur (*backward pass*) melalui setiap lapisan. *Error signal* merupakan hasil turunan (*derivative*) dari error terhadap *output* dari node. Secara matematis, *error signal* dapat ditulis menjadi

$$\xi_{1,i}^{(h)} = \frac{\partial E_p^{(h)}}{\partial O_{1,i}^{(h)}} \tag{13}$$

Pada persamaan 11, dimana $\xi_{1,i}^{(h)}$ melambangkan gradien dari fungsi kesalahan $E_p^{(h)}$ terhadap *output* neuron $O_{1,i}^{(h)}$ pada layer ke-h dan neuron ke-i. Pada

$\frac{\partial E_p^{(h)}}{\partial O_{1,i}^{(h)}}$ adalah turunan parsial dari fungsi kesalahan $E_p^{(h)}$ (fungsi error untuk data pelatihan p) terhadap *output* neuron ke-i di layer ke-h, $E_p^{(h)}$ adalah fungsi kesalahan (error function) untuk data pelatihan p pada layer ke-h dan $O_{1,i}^{(h)}$ adalah *output* dari neuron ke-i pada layer ke-h.

Hasilnya akan didapatkan *error* dari parameter yang digunakan. Apabila fungsi keanggotaan yang digunakan adalah Gaussian, maka *error* parameter yang didapatkan adalah $\xi_\mu^{(h)}$ dan $\xi_\sigma^{(h)}$. Selanjutnya parameter awal diperbarui dengan persamaan

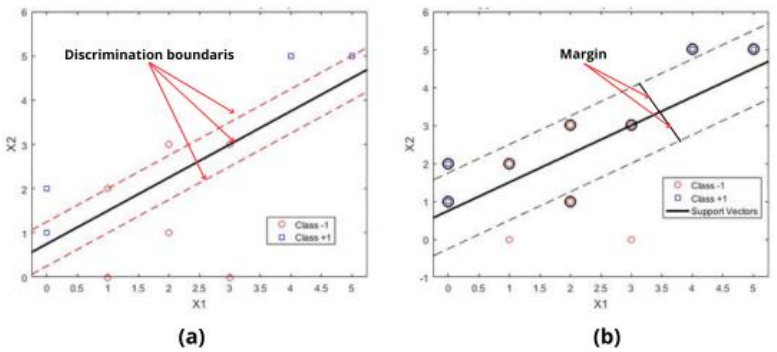
$$\begin{aligned} \Delta\mu^{(h)} &= -\eta\left(\xi_\mu^{(h)}\right) \\ \Delta\sigma^{(h)} &= -\eta\left(\xi_\sigma^{(h)}\right) \end{aligned} \tag{14}$$

dimana *h* menunjukkan jumlah iterasi (*epoch*) yang digunakan dan η adalah *learning rate* yang nilainya telah ditentukan.

1.5.4 Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah sistem pembelajaran yang menggunakan fungsi linier dalam ruang fitur berdimensi tinggi, dilatih dengan algoritma berdasarkan teori optimasi dan menggunakan bias pembelajaran dari teori statistic (Colin Campbell, 2011). *Support vector machine* berada dalam satu kelas dengan *Neural Network* dalam hal fungsi dan kondisi permasalahan yang bisa diselesaikan, keduanya masuk kedalam kelas *supervised learning* (Chopra & Khurana, 2023).

Secara teoritik SVM dikembangkan untuk menyelesaikan permasalahan klasifikasi pada dua kelas dengan mencari *hyperplane* terbaik. *Hyperplane* merupakan fungsi untuk memisahkan antara dua kelas pada *input space*, sehingga dari data yang tersebar dapat dilakukan klasifikasi dan analisa regresi. Ilustrasi SVM dapat dilihat pada Gambar 6.



Gambar 6. Illustrasi SVM
Sumber: (Nugroho et al., 2003)

Pada Gambar 6 (a) menunjukkan alternatif garis pemisah (*discrimination boundaries*) dimana *pattern* yang tergabung pada class –1 disimbolkan dengan warna merah (kotak), sedangkan *pattern* pada class +1, disimbolkan dengan warna kuning (lingkaran). Pada Gambar 6 (b) diperlihatkan bahwa terdapat garis *hyperplane* yang tepat berada diantara dua buah kelas. Prinsip dasar dari analisis ini adalah menemukan *hyperplane* terbaik yakni dengan meminimalkan kesalahan klasifikasi dan memaksimalkan margin geometriknya seperti pada Gambar 6 (b). Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM.

Hyperplane pemisah terbaik antara kedua *class* dapat ditemukan dengan mengukur *margin hyperplane* tersebut dan mencari titik maksimalnya (Nugroho et al., 2003). *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing *class*. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Garis solid pada gambar 1-b menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua kelas, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM.

Data yang tersedia dinotasikan dengan $x \in R^d$ sedangkan label masing-masing dinotasikan $y^i \in \{-1, +1\}$ untuk $i = 1,2,3, \dots n$ yang mana n adalah banyaknya data. Diasumsikan kedua kelas dapat terpisah secara sempurna oleh *hyperplane* yang berdimensi *d*, yang didefinisikan:

$$w.x + b = 0. \tag{15}$$

Pattern x_i yang termasuk kelas -1 (sampel negatif) dan dapat dirumuskan sebagai *pattern* yang memenuhi pertidaksamaan:

$$w.x + b \leq -1. \tag{16}$$

Pattern x_i yang termasuk kelas +1 (sampel positif)

$$w.x + b \geq +1, \tag{17}$$

dimana w adalah vektor bobot, x adalah nilai masukan atribut dan b adalah bias.

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara jarak dan titik terdekatnya, yaitu $1/\|w\|$. Hal ini dapat dirumuskan sebagai *Quadratic Programming (QP) problem*, yaitu mencari titik minimal persamaan 19, dengan memperlihatkan *constraint* persamaan 17.

$$min_w = \boldsymbol{\tau}(w) = \frac{1}{2}\|w\|^2 \tag{18}$$

$$y_i(x_i.w + b) - 1 \geq 0 \tag{19}$$

Masalah ini dapat dipecahkan dengan berbagai teknik komputasi, diantaranya *lagrange multiplier*.

$$L(w,b,a) = \frac{1}{2}\sum_{i=1}^{l} a_i \left(y_i\left((x.w+b)-1\right)\right) \quad (20)$$

dengan *i* = 1, 2, …, *l*

dimana a adalah *lagrange multiplier*, yang bernilai 0 atau positif $a_i \geq 0$.

Nilai optimal dari persamaan 20 dapat dihitung dengan meminimalkan L terhadap w dan b, dan memaksimalkan L terhadap a_i, dengan memperhatikan sifat bahwa pada titik optimal gradient L=0 persamaan 20 dapat dimodifikasi sebagai maksimal problem yang hanya mengandung a_i, sebagaimana terlihat pada persamaan 21 dan persamaan 22.

$$\sum_{i=1}^{l} a_i - \frac{1}{2}\sum_{i,j=1}^{l} a_i a_j y_i y_j x_i.x_j \quad (21)$$

Dengan *Constraint*:

$$a_i \geq 0 (i = 1, 2, \ldots, l) \sum_{i=1}^{l} a_i y_i = 0 \quad (22)$$

Jika dari hasil perhitungan, diperoleh banyak nilai a_i yang positif. Data yang berkaitan dengan nilai a_i positif ini disebut sebagai support vector. Penjelasan tersebut berdasarkan asumsi bahwa kedua belah kelas dapat terpisah secara sempurna oleh *hyperplane* (*linear separable*). Akan tetapi, pada umumnya dua belah kelas pada *input space* tidak dapat terpisah secara sempurna (*non linear separable*). Hal ini menyebabkan constraint pada persamaan 20 tidak dapat terpenuhi, sehingga optimisasi tidak dapat dilakukan. Untuk mengatasi masalah ini, SVM dirumuskan ulang dengan memperkenalkan teknik *softmargin*.

Dengan demikian, maka akan diperoleh a_i yang kebanyakan bernilai positif yang disebut sebagai *support vector* dan juga memperoleh persamaan 23 dan 24 sebagai solusi pemisah.

$$w = \sum a_i y_i x_i \quad (23)$$

$$b = y_k - w^T x_k \quad (24)$$

Pengerjaan SVM untuk untuk kasus *non linier* yaitu digunakan fungsi *mapping* ϕ, yang biasa disebut dengan fungsi *kernel* dan menggunakan algoritma *grid search* yang akan melatih banyak pasangan model dari *range* nilai yang telah ditentukan. Terdapat beberapa algoritma untuk menentukan parameter optimal pada model SVM, salah satunya adalah menggunakan *algoritma grid search*. Algoritma ini membagi jangkauan parameter yang akan dioptimalkan kedalam *grid* dan melintasi semua titik untuk mendapatkan parameter yang optimal. Dalam aplikasinya, *algoritma grid search* harus dipandu oleh beberapa metrik kinerja, biasanya diukur dengan *cross-validation* pada data *training* (Chopra & Khurana, 2023). Oleh karena

itu disarankan untuk mencoba beberapa variasi pasangan parameter pada hyperplane SVM.

Pasangan parameter yang menghasilkan akurasi terbaik yang didapatkan dari uji *cross-validation* merupakan parameter yang optimal. Parameter optimal tersebut yang selanjutnya digunakan untuk model SVM terbaik. Setelah itu, model SVM tersebut digunakan untuk memprediksi data testing untuk mendapatkan generalisasi tingkat akurasi model (Mutmainnah & Widodo, 2018). Tahapan yang dilakukan dalam pengoptimalan parameter adalah:

1. Inisialisasi Parameter

Menentukan nilai parameter C, γ dan ε yang kemudian akan di dapat parameter yang optimal untuk mendapatkan hasil terbaik. Semakin besar parameter C semakin besar finalisasi error untuk memaksimalkan margin. Nilai C digunakan untuk mengontrol *tradeoff* antara margin dan error klasifikasi. Nilai C relatif penting untuk memaksimumkan margin dan meminimumkan jumlah *slack*. Untuk parameter γ (*gamma*) dikarenakan semakin besar γ semakin bagus nilai prediksi pada training dan semakin buruk pada validasi, parameter gamma digunakan untuk mengontrol kecepatan proses learning dengan kecepatan tersebut ditunjukkan oleh jumlah iterasi yang dibutuhkan untuk mencapai konvergen. Untuk parameter epsilon (ε) yang semakin kecil dapat menghasilkan tingkat akurasi yang semakin meningkat, hal ini disebabkan karena nilai variabel epsilon (ε) yang semakin kecil dapat menghasilkan nilai α yang optimal sehingga dapat menentukan *support vector* yang tepat. *Support vector* yang tepat tersebut dapat menghasilkan *hyperplane* (garis pemisah) untuk memisahkan kedua kelas tersebut semakin optimal. Dari kombinasi-kombinasi parameter akan di masukkan kedalam algoritma SVM, lalu kombinasi yang akan memberikan nilai akurasi terbaik (Safira et al., 2023).

2. Pemilihan Kernel

Penentuan nilai parameter SVM terbaik dengan metode *grid search* menggunakan kernel. Fungsi kernel yang digunakan dalam penelitian ini adalah *radial*, *linear*, dan *Sigmoid*. Fungsi kernel digunakan untuk melakukan pemetaan dimensi awal (dari dimensi rendah) himpunan data ke dimensi baru (dimensi lebih tinggi). Algoritma SVM menggunakan sekumpulan fungsi matematika yang didefinisikan sebagai kernel. Penggunaan kernel bertujuan untuk mentransformasikan data ke ruang berdimensi tinggi, dengan menjadikan data *non linier* terpisah secara linier. Ketika terdapat permasalahan data yang tidak terpisah secara linear dalam ruang *input*, *soft margin* SVM tidak dapat menemukan *hyperplane* pemisah yang kuat yang meminimalkan misklasifikasi dari data *points* serta menggeneralisasi dengan baik. Untuk itu, kernel dapat digunakan untuk mentransformasi data ke ruang berdimensi lebih tinggi yang disebut sebagai ruang kernel, dimana akan menjadikan data terpisah secara *linear* (Awad & Khanna, 2015). Data disimpan dalam bentuk kernel yang mengukur kesamaan atau ketidaksamaan

objek data. Kernel dapat dibangun untuk berbagai objek data mulai dari data kontinu dan data diskrit melalui urutan data dan grafik.

Konsep substitusi kernel berlaku bagi metode lain dalam analisis data tetapi SVM merupakan yang paling terkenal dari metode dengan jangkauan kelas luas yang menggunakan kernel untuk merepresentasikan data dan dapat disebut sebagai metode berbasis kernel. Berikut merupakan ilustrasi contoh dalam melakukan pemisahan data menggunakan kernel. Diketahui bahwa data terdiri dari *input space* dengan dua buah $\boldsymbol{x_i} = \{x_{i1}, x_{i2}\}$ dan $\boldsymbol{x_j} = \{x_{j1}, x_{j2}\}$ Diasumsikan fungsi kernel akan dibuat dengan menggunakan *input* $\boldsymbol{x_i}$dan $\boldsymbol{x_j}$ seperti berikut (Colin Campbell, 2011).

$$\mathrm{K}\left(x_i, x_j\right) = \left(x_i{}^T x_j\right)^2$$

$$\mathrm{K}\left(x_i, x_j\right) = \left(x_{i1}x_{j1} + x_{i2}x_{j2}\right)^2$$

$$\mathrm{K}\left(x_i, x_j\right) = \left(x_{i1}^2 x_{j1}^2 + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2}\right)$$

$$\mathrm{K}\left(x_i, x_j\right) = \left(x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2\right)^T \left(x_{j1}^2, \sqrt{2}x_{j1}x_{j2}, x_{j2}^2\right)$$

$$\mathrm{K}\left(x_i, x_j\right) = \Phi(x_i)^T \Phi(x_j) \qquad (25)$$

Nilai K diatas secara implisit mendefinisikan pemetaan ke ruang dimensi yang lebih tinggi seperti berikut.

$$\mathrm{K}\left(x_i, x_j\right) = \left\{x_1^2, \sqrt{2}x_1x_2, x_2^2\right\} \qquad (26)$$

Kernel $\mathrm{K}\left(x_i, x_j\right)$ mengambil dua *input space* dan memberikan kesamaannya dalam *feature space* seperti berikut.

$\Phi : X \rightarrow F$

$\mathrm{K} : X \times X \rightarrow R, \mathrm{K}\left(x_i, x_j\right) = \Phi\,(x).\Phi\,(z)$

Berdasarkan pada fungsi kernel diatas, dapat dilakukan perhitungan untuk melakukan prediksi dari beberapa data dalam *feature space* seperti pada persamaan berikut (Colin Campbell, 2011) .

$$f\left(\Phi(x)\right) = sign\,(w.\Phi(z) + b)$$

$$f\left(\Phi(x)\right) = sign\,(\sum_{i=1}^{m} \alpha_i y_i\, \mathrm{K}\left(x_i, x_j\right) + b) \qquad (27)$$

dimana b adalah nilai bias, m adalah jumlah *support vector* dan $\mathrm{K}\left(x_i, x_j\right)$ adalah fungsi kernel.

Ada beberapa pilihan fungsi kernel yang dipakai pada sebuah aplikasi untuk mengatasi masalah pada metode *Support Vector Machine* (SVM) yaitu:

1. *Linear* Kernel

Kernel linear adalah fungsi kernel yang paling sederhana. Kernel linear digunakan ketika data yang dianalisis sudah terpisah secara linear. Kernel linear cocok ketika terdapat banyak fitur dikarenakan pemetaan ke ruang dimensi yang lebih tinggi tidak benar-benar meningkatkan kinerja. Persamaan untuk fungsi kernel adalah:

$$\mathrm{K}\left(x_i, x_j\right) = {x_i}^T x_j \tag{28}$$

2. *Sigmoid* Kernel

Sigmoid merupakan kernel trick SVM yang merupakan pengembangan dari jaringan saraf tiruan, dimana kernel ini dinyatakan dengan persamaan berikut

$$\mathrm{K}\left(x_i, x_j\right) = \tanh(\gamma \,.\, x_i \,.\, x_j + r) \tag{29}$$

dimana x_i dan x_j adalah vektor *input*, γ dan r adalah parameter, dan tanh adalah fungsi tangen hiperbolik.

Parameter gamma (γ) untuk mengontrol skala dari *input* data yang digunakan dalam kernel sigmoid dan parameter bias (r) ini yang menggeser fungsi sigmoid. Kernel sigmoid dapat memodelkan hubungan *non-linier* dan sigmoidal antara titik data, dan dapat memperkirakan fungsi aktivasi jaringan saraf. Kernel sigmoid cocok untuk data biner dan kategorikal, di mana titik datanya mungkin memiliki nilai diskrit dan logis. Kernel sigmoid juga dikenal sebagai kernel tangen hiperbolik atau kernel perceptron multilayer.

3. *Radial Basic Funcion* (RBF) Kernel

Kernel RBF atau juga disebut kernel *Gaussian* adalah konsep kernel yang paling banyak digunakan untuk memecahkan masalah klasifikasi data yang tidak dapat dipisahkan secara linear. Kernel ini dikenal memiliki performa yang baik dengan parameter tertentu, dan hasil dari pelatihan memiliki nilai *error* yang kecil dibandingkan dengan kernel lainnya. Rumus persamaan untuk fungsi kernel RBF yaitu:

$$\mathrm{K}\left(x_i, x_j\right) = exp\left\{-\gamma \left\|x_i - x_j\right\|^2\right\} \tag{30}$$

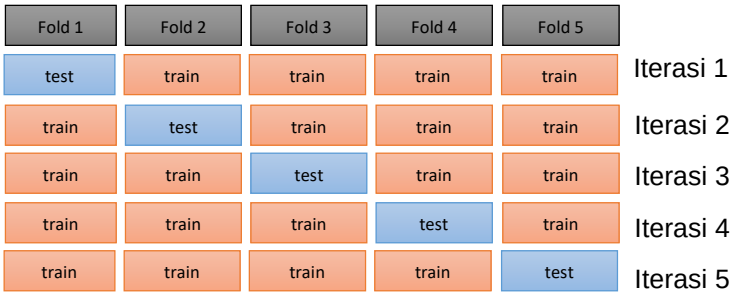
dimana $\mathrm{K}\left(x_i, x_j\right)$ adalah kernel, x_i adalah nilai *input*, x_j adalah jumlah nilai *input*.

1.5.5 K-Fold Cross Validation

Salah satu pendekatan alternatif untuk "*training* dan *testing*" yang sering di adopsi dalam beberapa kasus (dan beberapa lainnya terlepas dari ukurannya) yang di sebut dengan *K-fold cross validation*, dengan cara menguji besarnya error pada data testing (Santoso, 2007). Metode *K-fold cross validation* menetapkan K = N, ukuran dari data set. Metode ini dinamakan pendekatan *leave-one-out*, setiap *testing* set hanya mengandung satu *record*. Pendekatan ini memiliki keuntungan dalam

pengunaan sebanyak mungkin data untuk *training*. Data *testing* bersifat *mutually exclusive* dan secara efektif mencakup keseluruhan data set. Kekurangan dari pendekatan ini adalah banyaknya komputasi untuk mengulangi prosedur sebanyak N kali.

Pada *K-Fold Cross Validation*, dataset akan dibagi menjadi sejumlah *K-Folds*, di mana partisi tersebut terdiri atas data $D1$, data $D2, \dots, Dk$, masing-masing memiliki ukuran yang sama. Pada data *training* dan data *testing* akan dilakukan sebanyak *k*-kali. Pada iterasi ke-*i*, partisi D_i akan dijadikan sebagai data untuk pengujian, dan partisi yang tersisa lainnya secara kolektif digunakan untuk melatih model Penggambaran iterasi model tersebut dapat dilihat pada Gambar 7.



Gambar 7. Contoh Model 5-*Folds Cross Validation*

K-fold cross validation adalah salah satu teknik untuk mengevaluasi keakuratan model, dengan ciri-ciri:

1. Cocokkan model dengan data $y1,\dots,$ dan $y\hat{t}+1$ menunjukkan prediksi pengamatan selanjutnya. Kemudian menghitung nilai error ($et+1 = yt+1 - y\hat{t}+1$) untuk melakukan peramalan. (Koranga, et al., 2021)
2. Ulang langkah 1 untuk $t = m,\dots, - 1$ dimana m adalah adalah jumlah minimum pengamatan yang diperlukan untuk pemasangan model.
3. Hitung nilai MAPE dari $em+1,\dots,$.

1.5.6 Indeks Harga Saham Gabungan

Dilansir dari situs resmi Bursa Efek Indonesia (BEI), Indeks Harga Saham Gabungan (IHSG) adalah indeks yang mengukur kinerja semua saham tercatat di Papan Utama dan Papan Pengembangan BEI (IDX, 2021). IHSG pertama kali diperkenalkan ke masyarakat pada 1 April 1983. Saat itu, IHSG adalah indikator pergerakan saham di Bursa Efek Jakarta (BEJ) atau nama terdahulu dari BEI. Kemudian, hari dasar untuk perhitungan IHSG adalah 10 Agustus 1982. Pada tanggal tersebut, indeks ditetapkan dengan nilai dasar 100. Saham yang tercatat saat itu berjumlah 13 saham. Perhitungan pada IHSG adalah dengan menggunakan jumlah nilai pasar (harga pasar) dan jumlah nilai dasar (harga IPO) seluruh saham yang tercatat. *Jakarta Stock Exchange* (JKSE) adalah salah satu indeks saham gabungan yang digunakan oleh Bursa Efek Indonesia (BEI). Keterkaitan antar bursa yang direpresentasikan oleh hubungan antar indeks saham dapat terjadi karena investor menjadikan pergerakan

indeks saham di bursa lain sebagai salah satu informasi dalam proses pengambilan keputusan investasi.

Umumnya, setiap saham memiliki pergerakan yang berbeda-beda dalam satu hari. Ada yang naik, turun, maupun stagnan. Jika tren IHSG sedang naik, artinya harga-harga saham di BEI juga sedang mengalami tren peningkatan. Sebaliknya, jika posisi IHSG sedang melemah, berarti harga-harga saham di BEI secara general juga sedang menurun (Arviana, Nerissa, 2023). Indeks saham adalah ukuran statistik yang mencerminkan keseluruhan pergerakan harga atas sekumpulan saham yang dipilih berdasarkan kriteria dan metodologi tertentu serta dievaluasi secara berkala. Tujuan dan manfaat dari indeks saham antara lain:

1. Mengukur sentimen pasar
2. Dijadikan produk investasi pasif seperti Reksa Dana Indeks dan ETF Indeks serta produk turunan
3. *Benchmark* bagi portofolio aktif
4. Proksi dalam mengukur dan membuat model pengembalian investasi (*return*), risiko sistematis, dan kinerja yang disesuaikan dengan risiko
5. Proksi untuk kelas aset pada alokasi asset

1.5.7 Kriteria Pemilihan Model Terbaik

Pengujian akurasi dilakukan untuk mengukur performa dari setiap metode dengan menggunakan ukuran akurasi peramalan. Ukuran hasil peramalan juga dapat digunakan untuk memantau aktivitas peramalan dalam memastikan bahwa aktivitas peramalan yang dilakukan ini beroperasi dengan baik dan menghasilkan nilai yang akurat (Albert dkk, 2020). Kriteria pemilihan model terbaik yang digunakan pada penelitian ini yaitu:

1. *Root Mean Square Error* (RMSE)

Pengertian *Root Mean Square Error* (RMSE) adalah metode pengukuran dengan mengukur perbedaan nilai dari prediksi sebuah model sebagai estimasi atas nilai yang diobservasi. Keakuratan metode estimasi kesalahan pengukuran ditandai dengan adanya nilai RMSE yang kecil. Metode estimasi yang mempunyai RMSE lebih kecil dikatakan lebih akurat daripada metode estimasi yang mempunyai RMSE lebih besar.

Cara Menghitung RMSE adalah dengan mengurangi nilai aktual dengan nilai peramalan kemudian dikuadratkan dan dijumlahkan keseluruhan hasilnya kemudian dibagi dengan banyaknya data. Hasil perhitungan tersebut selanjutnya dihitung kembali untuk mencari nilai dari akar kuadrat seperti pada persamaan sebagai berikut:

$$RMSE = \left(\frac{\sum(y_i - \hat{y}_i)}{n}\right)^{\frac{1}{2}} \tag{31}$$

dimana y_i adalah nilai aktual, $\hat{y}_i$ adalah nilai hasil prediksi dan n adalah banyaknya data.

2. *Mean Absolute Percentage Error* (MAPE)

Mean Absolut Percentage error (MAPE) adalah persentase kesalahan rata-rata secara multak. Pengertian *Mean Absolute Percentage Error* adalah Pengukuran statistik tentang akurasi perkiraan (prediksi) pada metode peramalan. Pengukuranmenggunakan MAPE dapat digunakan secaraluas karena MAPE mudah dipahami dan diterapkan dalam menilai akurasi prediksi peramalan. Pada MAPE memberikan informasi seberapa besar kesalahan peramalan dibandingkan dengan nilai sebenarnya dari series tersebut. Semakin kecil nilai persentasi kesalahan (*percentage error*) pada MAPE maka semakin akurat hasil peramalan tersebut.

$$MAPE = \frac{\sum_{t=1}^{n} |(\frac{Z_t - \hat{Z}_t}{Z_t})\text{X}100\%|}{n} \quad (32)$$

Terdapat analisa rentang nilai MAPE sebagaimana tertulis dalam Tabel 1.

Tabel 1. Rentang Nilai MAPE

Nilai MAPE	Kriteria
< 10%	Kemampuan model peramalan sangat baik
10-20%	Kemampuan model peramalan baik
20-50%	Kemampuan model peramalan cukup
>50%	Kemampuan model peramalan buruk

BAB II
METODE PENELITIAN

2.1 Sumber Data dan Variabel Penelitian

Data yang digunakan adalah data sekunder yang diperoleh dari *yahoo finance* yaitu data penutupan saham harian Indeks Harga Saham Gabungan (IHSG) yang merupakan salah satu indeks saham gabungan mulai tanggal 2 Januari 2020 sampai 29 Desember 2023 dengan jumlah data 974 data yang diakses dari *www.finance.yahoo.com* (Finance, 2024). Data harga saham dibagi menjadi 80% data *training* dan 20% data *testing*. Proporsi 80% untuk pelatihan membantu model memahami pola dalam data, sementara 20% untuk pengujian memastikan bahwa evaluasi kinerja model dilakukan pada data yang tidak terlihat sebelumnya, sehingga mampu memberikan gambaran yang akurat tentang kemampuan generalisasi model.

Data harga penutupan saham IHSG yang digunakan dalam penelitian ini memiliki karakteristik fluktuasi yang signifikan, bersifat kompleks, *non-linear*, dan tidak stasioner. Karakteristik ini sesuai dengan persyaratan yang diperlukan untuk penerapan metode *machine learning* seperti ANFIS dan SVM, yang dirancang untuk menangani pola data non-linear dan kompleks. Selain itu, proses normalisasi data telah memastikan bahwa data berada dalam rentang yang mendukung efisiensi komputasi kedua metode tersebut. Dengan demikian, data ini dapat diolah secara efektif untuk membangun model prediksi yang andal.

Variabel yang terdapat pada data tersebut adalah data harian harga pembuka (*Open*) dalam satuan rupiah (Rp), harga tertinggi (*High*) dalam satuan rupiah (Rp), harga terendah (*Low*) dalam satuan rupiah (Rp), dan harga penutupan (*Close*) dalam satuan rupiah (Rp), yang diperoleh dari *Jakarta Stock Exchange Composite Index* (JKSE) atau Indeks Harga Saham Gabungan (IHSG). Adapun data yang digunakan dalam peramalan yaitu data harga penutupan saham (*Close*). Harga penutupan saham dapat mencerminkan nilai terakhir di mana saham diperdagangkan pada akhir hari perdagangan dan sering dianggap sebagai harga yang paling representatif dari saham tersebut untuk hari itu.

Tabel 2. Struktur Data Saham JKSE

No	Tanggal	*Open*	*High*	*Low*	*Close*
1.	1/2/2020	6313.128	6317.013	6263.676	6283.581
2.	1/3/2020	6306.187	6323.466	6287.706	6323.466
3.	1/4/2020	6293.499	6300.436	6252.635	6257.403
4.	1/7/2020	6272.220	6284.892	6246.129	6279.346
5.	1/8/2020	6248.443	6250.122	6218.130	6225.686
6.	1/9/2020	6248.661	6274.493	6238.983	6274.493
7.	1/10/2020	6287.167	6295.37	6271.98	6274.941
8.	1/13/2020	6287.912	6297.776	6269.481	6296.567

9.	1/14/2020	6308.890	6325.406	6298.608	6325.406
10.	1/15/2020	6326.166	6348.526	6255.498	6283.365
11.	1/16/2020	6275.956	6299.541	6255.493	6286.048
12.	1/17/2020	6293.782	6301.485	6266.932	6291.657
…	…	…	…	…	…
974.	12/29/2023	7307.111	7309.782	7259.682	7272.797

Sumber: (Yahoo finance, 2024)

2.2 Metode Penelitian

Langkah-langkah analisis yang akan dilakukan pada penelitian ini adalah:

1. Deskripsi Data
2. *Pre-Processing* Data
3. Pembagian Data
4. Pemodelan SVM:
   a. Membuat model awal dari data *training*.
   b. Menentukan tipe kernel beserta nilai parameter dengan menggunakan *Grid Search Optimization*.
   c. Menguji model dengan Data *Testing*.
5. Pemodelan ANFIS:
   a. Menentukan fungsi keanggotaan.
   b. Melatih model dengan parameter yang disesuaikan (*epoch*, *error tolerance*).
   c. Menguji model dengan Data *Testing*.
6. Evaluasi Peramalan
7. Denormalisasi Data