**Perbandingan Algoritma Support Vector Regression dan Artificial Neural Network dalam Prediksi Jumlah Pendaftar Calon Mahasiswa Baru**

**Institut Teknologi Del**

Dokumen Tugas Akhir

Disampaikan Sebagai Bagian dari Persyaratan Kelulusan Sarjana

Program Studi Sarjana Informatika

Oleh :

|  |  |
| --- | --- |
| 11S19034 | Hotmangasi Manurung |
| 11S19061 | Agnes Bertua Nababan |
|  |  |



Institut Teknologi Del

2022/2023

# DAFTAR ISI

[DAFTAR ISI ii](#_Toc138247882)

[DAFTAR GAMBAR v](#_Toc138247883)

[DAFTAR TABEL vi](#_Toc138247884)

[BAB 1 PENDAHULUAN 1](#_Toc138247885)

[1.1 Latar Belakang 1](#_Toc138247886)

[1.2 Rumusan Masalah 4](#_Toc138247887)

[1.3 Tujuan Penelitian 4](#_Toc138247888)

[1.4 Ruang Lingkup Penelitiann 4](#_Toc138247889)

[1.5 Hasil yang Diharapkann 5](#_Toc138247890)

[1.6 Tahapan Penelitiann 5](#_Toc138247891)

[1.7 Sistematika Penyajian 6](#_Toc138247892)

[BAB 2 TINJAUAN PUSTAKA 9](#_Toc138247893)

[2.1 Penelitian Terkait 9](#_Toc138247894)

[2.2 Pendaftaran Calon Mahasiswa Baru 11](#_Toc138247895)

[2.3 Machine Learning 12](#_Toc138247896)

[2.4 Data Preprocessing 14](#_Toc138247897)

[2.5 Label Encoding 15](#_Toc138247898)

[2.6 Normalisasi 16](#_Toc138247899)

[2.7 K-Fold Cross Validation 16](#_Toc138247900)

[2.8 *Grid Search* 17](#_Toc138247901)

[2.9 Support Vector Regression (SVR) 18](#_Toc138247902)

[2.10 Artificial Neural Network 21](#_Toc138247903)

[2.11 Backpropagation 23](#_Toc138247904)

[2.12 Early Stopping 28](#_Toc138247905)

[2.13 Evaluasi Hasil 29](#_Toc138247906)

[2.13.1 Mean Square Error 30](#_Toc138247907)

[2.13.2 Root Mean Square Error 30](#_Toc138247908)

[2.14 Flask 30](#_Toc138247909)

[BAB 3 ANALISIS 32](#_Toc138247910)

[3.1 Analisis Domain 32](#_Toc138247911)

[3.2 Analisis Data 32](#_Toc138247912)

[3.2.1 Sumber Data 33](#_Toc138247913)

[3.2.2 Seleksi Data 33](#_Toc138247914)

[3.3 Preprocessing Data 34](#_Toc138247915)

[3.3.1 Data Integration 34](#_Toc138247916)

[3.3.2 Data Cleaning 35](#_Toc138247917)

[3.3.3 Data Encoding 38](#_Toc138247918)

[3.3.4 Normalisasi Dataset 38](#_Toc138247919)

[3.4 Analisis Algoritma SVR 40](#_Toc138247920)

[3.4.1 Perhitungan SVR 40](#_Toc138247921)

[3.5 Analisis Algoritma ANN Backpropagation 46](#_Toc138247922)

[3.5.1 Inisialisasi Bias dan Bobot 47](#_Toc138247923)

[3.5.2 Inisialisasi *Hyperparameter* 48](#_Toc138247924)

[3.5.3 Feedforward Propagation 48](#_Toc138247925)

[3.5.4 Backpropagation Propagation 50](#_Toc138247926)

[3.6 Analisis Evaluasi Model 51](#_Toc138247927)

[3.7 Hipotesis Penelitian 51](#_Toc138247928)

[3.7.1 Perumusan H0 dan H1 52](#_Toc138247929)

[BAB 4 PERANCANGAN 52](#_Toc138247930)

[4.1 Rancangan Desain Penelitian 53](#_Toc138247931)

[4.2 Rancangan Pre-Processing Data 55](#_Toc138247932)

[4.3 Rancangan Normalisasi 56](#_Toc138247933)

[4.4 Rancangan Implementasi Algoritma 56](#_Toc138247934)

[4.4.1 Rancangan Implementasi Algoritma SVR 56](#_Toc138247935)

[4.4.2 Rancangan Implementasi Algoritma ANN Backpropagation 58](#_Toc138247936)

[4.5 Rancangan Aplikasi Del Predict 61](#_Toc138247937)

[4.5.1 Use Case Diagram 62](#_Toc138247938)

[4.5.2 Use Case Scenario 63](#_Toc138247939)

[4.5.3 Design Aplikasi 64](#_Toc138247940)

[BAB 5 IMPLEMENTASI 67](#_Toc138247941)

[5.1 Lingkungan Implementasi 67](#_Toc138247942)

[5.2 Batasan Implementasi 68](#_Toc138247943)

[5.3 Implementasi Preprocessing 68](#_Toc138247944)

[5.3.1. Encoding 68](#_Toc138247945)

[5.3.2. Normalisasi 69](#_Toc138247946)

[5.4 Implementasi Algoritma Support Vector Regression 69](#_Toc138247947)

[5.5 Implementasi Algoritma Artificial Neural Network 72](#_Toc138247948)

[5.5.1 Percobaan Early Stopping 76](#_Toc138247949)

[5.5.2 Percobaan Variasi Nilai Epoch 77](#_Toc138247950)

[BAB 6 HASIL DAN PEMBAHASAN 78](#_Toc138247951)

[6.1. Hasil dan Pembahasan Preprocessing Dataset 78](#_Toc138247952)

[6.2. Hasil dan Pembahasan Algoritma Support Vector Regression 78](#_Toc138247953)

[6.3. Hasil dan Pembahasan Algoritma Artificial Neural Network 80](#_Toc138247954)

[6.4. Pembahasan Evaluasi Model 81](#_Toc138247955)

[6.5. Hasil dan Pembahasan Prototype 81](#_Toc138247956)

[BAB 7 KESIMPULAN DAN SARAN 82](#_Toc138247957)

[7.1. Kesimpulan 82](#_Toc138247958)

[7.2. Saran 82](#_Toc138247959)

[LAMPIRAN 82](#_Toc138247960)

[Lampiran A Source Code 82](#_Toc138247961)

[Source Code 1. Encoding Data 82](#_Toc138247962)

[Source Code 2. *Grid Search* ANN 83](#_Toc138247963)

[Source Code 3. *Grid Search* SVR 85](#_Toc138247964)

[Source Code 4. Pemodelan Algoritma ANN 86](#_Toc138247965)

[Source Code 5. Pemodelan Algoritma SVR 89](#_Toc138247966)

[Lampiran B Hasil Implementasi 91](#_Toc138247967)

[B.1. Hasil Implementasi Early Stopping 91](#_Toc138247968)

[B.2. Hasil Implementasi Artificial Neural Network 91](#_Toc138247969)

[B.3. Hasil Implementasi Support Vector Regression 107](#_Toc138247970)

[Lampiran C Aplikasi Web Sistem Prediksi Del Predict 108](#_Toc138247971)

[DAFTAR REFERENSI 109](#_Toc138247972)

# DAFTAR GAMBAR

[Gambar 2. 1 Jumlah Pendaftar Calon Mahasiswa Baru IT Del 2016-2022 11](#_Toc138319397)

[Gambar 2. 2 Skema 10 Fold Cross Validation 17](#_Toc138319398)

[Gambar 2. 3 Ilustrasi SVR 20](#_Toc138319399)

[Gambar 2. 4 Arsitektur Algoritma Artificial Neural Network 22](#_Toc138319400)

[Gambar 2. 5 Model Overfitting 29](#_Toc138319401)

# DAFTAR TABEL

[Tabel 3. 1 Informasi yang diperoleh dari BAAK IT Del 31](#_Toc136384174)

[Tabel 3. 2 Atribut pada dataset 32](#_Toc136384175)

[Tabel 3. 3 Dataset hasil integration 33](#_Toc136384176)

[Tabel 3. 4 Dataset sebelum dilakukan missing value 34](#_Toc136384177)

[Tabel 3. 5 Dataset sesudah dilakukan missing value 35](#_Toc136384178)

[Tabel 3. 6 Sampel Dataset sebelum normalisasi 36](#_Toc136384179)

[Tabel 3. 7 Sampel Dataset setelah normalisasi 37](#_Toc136384180)

[Tabel 3. 8 Inisialisasi nilai parameter 39](#_Toc136384181)

[Tabel 3. 9 Data Uji yang sudah dinormalisasikan 39](#_Toc136384182)

[Tabel 3. 10 Data Uji 40](#_Toc136384183)

[Tabel 3. 11 Pasangan Data 40](#_Toc136384184)

[Tabel 3. 12 Hasil Perhitungan RBF 41](#_Toc136384185)

[Tabel 3. 13 Hasil Perhitungan Matriks Hessian 41](#_Toc136384186)

[Tabel 3. 14 Hasil Perhitungan Nilai Error 42](#_Toc136384187)

[Tabel 3. 15 Hasil Perhitungan Delta Alpha 42](#_Toc136384188)

[Tabel 3. 16 Hasil Perhitungan Nilai Alpha Baru 43](#_Toc136384189)

[Tabel 3. 17 Hasil Perhitungan Nilai Alpha Star 43](#_Toc136384190)

[Tabel 3. 18 Data Uji yang Ditesting 44](#_Toc136384191)

[Tabel 3. 19 Hasil Alpha Star yang didapat sebelumnya 44](#_Toc136384192)

[Tabel 3. 20 Nilai bias dari input layer ke *hidden layer* 45](#_Toc136384193)

[Tabel 3. 21 Nilai bobot dari inputan layer ke *hidden layer* 45](#_Toc136384194)

[Tabel 3. 22 Nilai bias dari *hidden layer* ke output layer 46](#_Toc136384195)

[Tabel 3. 23 Nilai bobot dari *hidden layer* ke output layer 46](#_Toc136384196)

# BAB 1 PENDAHULUAN

Pada bab ini dijelaskan mengenai latar belakang topik penelitian, rumusan permasalahan penelitian, tujuan penelitian, ruangg lingkup penelitian, metodee penelitian dan sistematika penulisan dalam menyusun Tugas Akhir ini.

## 1.1 Latar Belakang

Pembukaan pendaftaran mahasiswa baru adalah rutinitas setiap perguruan tinggi. Rutinitas ini juga dilakukan oleh Institut Teknologi Del setiap tahunnya dengan membuka beberapa jalur yaitu PMDK, USM 1, USM 2, USM 3, dan jalur nilai UTBK (Itdel, 2022). Jumlah penerimaan mahasiswa baru mencerminkan bagaimana pandangan dan minat masyarakat pada perguruan tinggi tersebut. Perguruan tinggi swasta saling bersaing untuk meningkatkan jumlah pendaftar mahasiswa baru sehingga jumlah pendaftar mahasiswa baru dapat meningkat maupun menurun setiap tahunnya (Khoirudin et al., 2019). Beberapa faktor yang mempengaruhi daya tarik untuk mendaftar ke suatu perguruan tinggi yaitu besarnya biaya pendidikan, beasiswa yang disediakan, dan bagaimana perguruan tinggi mempromosikan kampusnya (Barus & Simamora, 2020).

Salah satu kesulitan yang dihadapi oleh calon pendaftar atau siswa SMA dalam melanjutkan studinya adalah kesulitan untuk menentukan perguruan tinggi yang akan dipilih. Kesulitan ini sering terjadi dikarenakan kurangnya informasi yang dimiliki untuk mengenal perguruan tinggi yang ada. Pada bentuk kesulitan seperti ini peran dari pihak promosi perguruan tinggi bekerja dalam menarik perhatian pendaftar sangat penting (Barus & Simamora, 2020). Pelaksanaan promosi ini sangat membantu para calon pendaftar atau siswa SMA untuk mengenal perguruan tinggi dan membantu untuk memutuskan perguruan tinggi yang akan dipilih.

Kegiatan promosi juga dilakukan oleh Institut Teknologi Del (IT Del) untuk meningkatkan daya tarik calon pendaftar untuk berkuliah di institusinya. Ada banyak bentuk promosi yang dilakukan pihak IT Del, salah satu bentuk promosi yang dilakukan adalah pengenalan institut ke sekolah-sekolah. Bentuk promosi ini dapat dimulai dengan melakukan promosi pada sekolah yang kemungkinan jumlah pendaftar ke kampus Institut Teknologi Del yang masih sedikit. Untuk mengetahui kemungkinan jumlah pendaftar dari suatu sekolah, maka perlu dibuat sebuah sistem yang dapat memprediksi jumlah pendaftar berdasarkan sekolah asal untuk tahun selanjutnya.

Prediksi jumlah pendaftar telah dilakukan oleh beberapa peneliti sebelumnya diantaranya, Sistem Prediksi untuk Menentukan Jumlah Pendaftaran Mahasiswa Baru pada Universitas Catur Insan Cendekia Menggunakan Metode *Least Square* (Muhadzdzab et al., 2020) yang membuat sistem prediksi untuk mengetahui jumlah calon mahasiswa baru sebagai upaya untuk mengetahui bagaimana prioritas atau banyak jumlah calon mahasiswa baru yang akan diterima dan sebagai sarana peminat trend prodi terbanyak pada setiap tahunnya. Pada penelitian tersebut, peneliti melakukan prediksi menggunakan algoritma *Least Square.* Berdasarkan hasil prediksi yang dilakukan diperoleh bahwa prediksi jumlah pendaftar pada tahun 2019 adalah 144 orang dengan perolehan MAPE 8,87%.

Penelitian lain yang membahas mengenai prediksi jumlah pendaftar antara lain adalah Implementasi Algoritma Regresi Linear dalam Sistem Prediksi Pendaftar Mahasiswa Baru Sekolah Tinggi Teknologi Indonesia Tanjungpinang (Setyoningrum et al., 2022). Pada penelitian ini peneliti membuat sebuah sistem prediksi jumlah pendaftar dengan menggunakan algoritma regresi linear, dimana dari hasil penelitian ini peneliti berhasil membangun sebuah sistem yang digunakan dalam memprediksi jumlah pendaftar mahasiswa Baru sekolah Tinggi Teknologi Indonesia.

Pada penelitian ini, penulis akan melakukan sistem prediksi jumlah pendaftar calon mahasiswa baru dengan menggunakan hasil perbandingan algoritma Support Vector Regression (SVR) dan Artificial Neural Network (ANN). Sebelum melakukan penelitian, peneliti juga melakukan eksplorasi mengenai algoritma ini dan memperoleh beberapa hasil penelitian dengan kedua algoritma tersebut. Beberapa penelitian yang membahas mengenai kedua algoritma ini antara lain, Perbandingan Algoritma Backpropagation Neural Network dan Long Short Term Memory dalam Memprediksi Harga Bitcoin (Andreas et al., 2022). Pada penelitian ini, peneliti membandingkan dua algoritma regresi yaitu algoritma Backpropagation dengan algoritma LSTM (*Long Short-Term Memory*) dalam melakukan prediksi harga Bitcoin. Pada penelitian ini algoritma dibandingkan dengan melakukan perhitungan RMSE dari kedua algoritma, kemudian algoritma yang memiliki RMSE terkecil dipilih sebagai algoritma yang lebih baik, dimana dari hasil penelitian ini algoritma dengan RMSE terkecil adalah algoritma Backpropagation.

Penelitian lainnya adalah Perbandingann Regresi Linear, Backpropagation, dan Fuzzyy Mamdanii dalam Prediksi Harga Emass(Nafi’iyah, 2016). Pada penelitian ini dilakukan pemodelan dengan menggunakan algoritma regresii linear, backpropagation, dan *fuzzy mandani* dalam melakukann prediksii harga emas. Ketiga algoritma ini dibandingkan berdasarkan nilai persentasi akurasi yang didapatkan. Berdasarkan hasil penelitian didapatkan bahwa dari hasil prediksi ketiga algoritma bahwa algoritma *backpropagation* merupakan algoritma yang paling baik dalam memprediksiiemas, kemudian dilanjutkan dengan algoritma regresi linier. Sedangkan untuk algoritma *fuzzy mamdani* tidak dapat melakukan prediksi harga emas secara baik, yang dibuktikann dengan nilai akurasi yang tidak sampai 1%.

Penelitian lainnya yang telah dilakukan yaitu Analisis Perbandingan Prediksi Obat dengan Menggunakan Metode ABC Analysis dan SVR pada Aplikasi “MORBIS”(Maryana et al., 2019). Pada penelitian ini didapatkan hasil bahwa optimasi peramalan penjualan obat dengan menggunakan SVR adalah dengan menggunakan proses *preprocessing* data dengan scaling linier mendapatkan hasil yang lebih optimal dengan nilai MAPE 17,51%.

Dari hasil kajian beberapa penelitian di atas diperoleh bahwa algoritma SVR dan ANN merupakan algoritma terbaik untuk melakukan prediksi dibandingkan algoritma lainnya. Oleh karena itu, penulis mencoba untuk menganalisis algoritma SVR dan ANN lalu menentukan algoritma yang terbaik dari kedua algoritma tersebut. Setelah ditentukan algoritma terbaik, maka penulis akan membuat sebuah aplikasi website untuk memprediksi jumlah pendaftar calon mahasiswa dengan menggunakan algoritma tersebut.

## 1.2 Rumusan Masalah

Rumusannmasalah dariipenelitiannini antaraalain :

1. Bagaimana cara memprediksi jumlah pendaftar calon mahasiswa baru Institut Teknologi Del di tahun selanjutnya dengan menggunakan algoritma *Support Vector Regressionn*(SVR) dan *ArtificiallNeurall Networkk*(ANN) ?
2. Bagaimana nilai RMSE yang diperoleh dari hasil pembentukan model algoritma yang telah dibuat?

## 1.3 Tujuan Penelitian

Tujuanndari penelitian ini antara lainn:

1. Melakukan prediksi jumlah pendaftar calon mahasiswa baru Institut Teknologi Del dengan menggunakan algoritma *SupporttVector Regression* (SVR)dan *ArtificiallNeural Network* (ANN).
2. Membandingkan nilai performansi model yang dilakukan setelah pembentukan model algoritma *Support Vector Regression* (SVR) dan *Artificial Neural Network* (ANN).

## 1.4 Ruang Lingkup Penelitiann

Ruang lingkup penelitian ini adalah sebagai berikut :

1. Penelitian ini berfokus pada penggunaan algoritma *Support Vector Regression (*SVR*)* dan algoritma *ArtificiallNeural Network* (ANN) *Backpropagationn*dalam prediksi jumlah pendaftar calon mahasiswa baru di Institut Teknologi Del.
2. Data yang digunakan adalah data yang diterima dari BAAK IT Del.
3. Variabel yang digunakan yaitu nama sekolah, wilayah kabupaten sekolah, jumlah pendaftar tahun 2016, jumlah pendaftar tahun 2017, jumlah pendaftar tahun 2018, jumlah pendaftar tahun 2019, jumlah pendaftar tahun 2020, jumlah pendaftar tahun 2021, jumlah pendaftar tahun 2022.
4. Penelitian ini menggunakan model normalisasi *min-max normalization.*
5. Sistem prediksi yang dilakukan hanya akan memprediksi jumlah pendaftar untuk 1 tahun selanjutnya.
6. Simulator algoritma akan dibuat dengan membuat aplikasi website.
7. Aplikasi yang dibangun hanya untuk melakukan prediksi jumlah pendaftar calon mahasiswa baru di Institut Teknologi Del dengan menggunakan algoritma yang menghasilkan evaluasi model terbaik.

## 1.5 Hasil yang Diharapkann

Hasillyang diharapkan dalammpelaksanaan penelitiannTugas Akhir ini adalah mampu melakukan prediksi jumlah pendaftar calon mahasiswa baru di Institut Teknologi Del dan menemukan algoritma terbaik dari antara algoritma *Support VectorrRegression* dan *ArtificiallNeurallNetwork* (ANN) berdasarkan hasil performansi model yang diperoleh.

## 1.6 Tahapan Penelitiann

Tahapan penelitianndalam Tugas Akhir ini dideskripsikan pada pernyataan dibawah ini:

1. Studi Literatur

Pada tahapannini dilakukan pengumpulanninformasi terkaitttopik penelitian mengenai algoritma *Support Vector Regression* dan *Artificial Neural Network* dalam melakukan prediksi melalui sumber referensi jurnal, *paper*, buku, situs perguruan tinggi, dan *website* yang terpercaya (seperti *website* milik pemerintah dan situs berita yang bereputasi).

1. Perumusan Masalah

Pada tahapan ini yang dilaksanakan adalah merumuskan masalah dan merumuskan tujuan penelitian tersebut. Hasil analisis dari sumber referensi-referensi dan data yang sudah disiapkan akan membantu membuat rumusan masalah penelitian dan hasil rumusan masalah tersebut yang nantinya dijadikan tujuan penelitian.

1. Pengumpulan Dataa

Pada tahapan ini yang dilaksanakan adalah pengumpulan data-data sebagai bahan pemecahan masalah yang yang telah dirumuskan sebelumnya pada tahap kedua. Untuk pengumpulan data dilakukan peneliti dengan mengambil data dari kantor BAAK kampus Institut Teknologi Del.

1. *Preprocessing* Data

Tahapan *preprocessing* ini dilakukan untuk menyiapkan data yang dapat diproses oleh algoritma *machine learning*. Memproses data terlebih dahulu penting dilakukan sehingga data yang akan digunakan tidak mengandung kata-kata ataupun karakter yang tidak diperlukan. Sehingga data akan lebih mudah untuk diproses dan meningkatkan akurasi prediksi.

1. Analisis Data dan Algoritma

Pada tahapan ini peneliti melakukan analisis terhadap dataset yang akan digunakan dan akan menganalisis algoritma *SupporttVector Regression* dan *ArtificiallNeural Networkk*(ANN). Pada analisis algoritma, peneliti memperdalam bagaimana proses kerja dari kedua algoritma. Peneliti juga membuat perhitungan matematis dari kedua algoritma tersebut.

1. Implementasi Algoritma

Pada tahapan ini, peneliti melakukan implementasi kedua algoritma berdasarkan hasil analisis yang telah dilakukan pada tahapan sebelumnya.

Implementasi algoritma dilakukan dengan menggunakan bahasa python yang menjadi bahasa pembelajaran *machine learning*.

1. Kesimpulan dan Saran Penelitian

Pada tahapan ini peneliti melakukan pembahasan dan penarikan kesimpulan terhadap proses penelitian yang telah dilaksanakan yaitu dengan melakukan penerapan algoritma *SupporttVector Regressionn*(SVR) dan *Artificial Neural Network* (ANN) dalam melakukan prediksi jumlah pendaftar calon mahasiswa baru Institut Teknologi Del.

## 1.7 Sistematika Penyajian

Penelitian Tugas Akhir ini terdiri dari beberapa bab yang disusun berdasarkan sistematika sebagai berikut :

|  |  |
| --- | --- |
| Bab I  Pendahuluan | Pada bab ini dijelaskan mengenai latar belakang topik penelitian, rumusan permasalahan penelitian, tujuan penelitian, ruang lingkup penelitian, metode penelitian dan sistematika penyajian dalam menyusun Tugas Akhir ini. |
| Bab II  Tinjauan Pustaka | Pada bab ini menjelaskan rangkuman informasi dari beberapa dasar teori berupa daftar pustaka yang mendasari dan mendukung penelitian Tugas Akhir ini. Penjelasan rinci yakni penjelasan penelitian sebelumnya yang mirip dengan penelitian Tugas Akhir dilakukan dalam dokumen ini, penjelasan teoritis dari dasar acuan jurnal, artikel, atau buku yang diangkat peneliti dalam menelaah topik dalam Tugas Akhir ini. |
| Bab III  Analisis | Pada bab ini, peneliti memberikan penjelasan mengenai analisis terhadap proses yang telah ditentukan sebelumnya dengan metode yang telah dipilih untuk melakukan proses prediksi jumlah pendaftar calon mahasiswa baru. |
| Bab IVV  Perancangann | Pada Bab ini, peneliti memberikan penjelasan mengenai rancangan penelitian yang akan dilakukan sehingga dapat menjelaskan secara rinci mengenai proses penelitian yang akan dilakukan dengan menggunakan *flowchart*. |
| Bab V  Implementasii | Pada bab ini dijelaskan bagaimana proses mengenai implementasi yang dilakukan berdasarkan analisis bab III dan perancangan yang telah dibahas pada Bab IV. |
| Bab VI  Hasill dan Pembahasan | Pada bab ini akan diuraikan hasil dan pembahasan dari implementasi dan percobaan yang dilakukan untuk mendapatkan hasill yang diharapkan dari penelitiannTugass Akhir ini. |
| Bab VII  Kesimpulan dan Saran | Pada bab ini akan disampaikan kesimpulan dari hasil implementasi dan percobaan dalam penelitian yang dilakukan. Dan juga akan disampaikan beberapa saran yang diperlukan untuk penelitian selanjutnya. |

# BAB 2 TINJAUAN PUSTAKA

Pada bab 2 Tinjauan Pustaka ini berisi landasan teori yang memuat data dan informasi terkait yang dikumpulkan dari berbagai jurnal dan pustaka yang berhubungan dengan penelitian dalam pengerjaan Tugas Akhir untuk memperluas informasi dalam melakukan kajian.

## 2.1 Penelitian Terkait

Sebelum penulis melakukan penelitian Tugas Akhir ini, penulis terlebih dahulu melakukan eksplorasi masalah perbandingan algoritma dalam pembuatan sebuah model untuk mendapatkan hasil yang terbaik terhadap serangkaian kasus. Kemudian berdasarkan permasalahan tersebut, penulis melakukan analisis terhadap beberapa paper/jurnal untuk mendapatkan metode, objek penelitian, hasil penelitian serta relevansi dan juga perbedaan dengan penelitian sebelumnya. Berdasarkan hasil analisis yang didapatkan, penulis melanjutkan penelitian dengan melakukan pengembangan menggunakan metode dan objek penelitian yang berbeda, sehingga dilakukan perbandingan untuk membandingkan satu metode dengan metode lainnya untuk mendapatkan model terbaik.

* + - 1. Perbandingan Algoritma Backpropagation Neural Network dan Long Short Term Memoryydalam Memprediksi Harga Bitcoinn

Pada penelitian ini, peneliti membandingkan dua algoritma regresi yaitu algoritma Backpropagation dengan algoritma LSTM (*LonggShort-Term Memory*) dalam melakukan prediksi harga Bitcoin. Pada penelitian ini, algoritma dibandingkan dengan melakukan perhitungan RMSE dari kedua algoritma, kemudian algoritma yang memiliki RMSE terkecil dipilih sebagai algoritma yang lebih baik, dimana dari hasil penelitian ini algoritma dengan RMSE terkecil adalah algoritma Backpropagation (Andreas et al., 2022).

* + - 1. Machine Learning menggunakan Perbandingan Dua Algoritma Antara Support Vector Regression (SVR) dan Decision Tree untuk Memprediksi Harga Saham Agro. Penelitian ini dilakukan dengan menggunakan metode Algoritma SVR dan Decision Tree. Dari hasil penelitian ini didapatkan kesimpulan bahwa tingkat akurasi pada SVR sebesar 96,30% dan akurasi pada decision tree sebesar 90,66% sehingga kedua algoritma ini baik digunakan untuk prediksi saham BRI Agroniaga (AGRO) (Lembong, 2022).
      2. Analisis Support Vector Regression (SVR) dengan Kernel Radial Basis Function (RBF) untuk Memprediksi Laju Inflasi di Indonesia.

Penelitian ini dilakukan dengan menggunakan metode algoritma *Support Vector Regression* dengan menggunakan kernel RBF. Dari hasil penelitian ini didapat kesimpulan bahwa Akurasi hasil prediksi yang dilakukan dengan melihat nilai dari Root Mean Square Error. Dengan nilai RMSE yang dihasilkan sebesar 0,0020 yang memiliki pengertian bahwa kemampuan model dapat mengikuti pola data dengan baik (Rais, 2022).

* + - 1. Application of artificial neural network and support vector regresion in predicting mass of ber fruits (Ziziphus mauritiana Lamk) based on fruit axial dimensions. Penelitian ini dilakukan dengan menggunakan algoritma Artificial Neural Network dan Support Vector Regression. Dari hasil penelitian ini didapatkan bahwa nilai RMSE dari ANN yaitu 1.8479, dengan R2 yaitu 0.9771 dan untuk algoritma SVR dengan nilai RMSE 1.8814, dengan nilai R2 yaitu 0.9947. Berdasarkan hasil yang didapatkan kemudian disimpulkan bahwa algoritma ANN mampu meningkatkan kemampuan untuk merancang sistem penilaian kultivar buah secara akurat (Abdel-Sattar et al., 2021)
      2. PerbandingannRegresi Linear, Backpropagation, dan Fuzzy Mamdani dalam Prediksi HargaaEmass(Nafi’iyah, 2016). Pada penelitian ini dilakukan pemodelan dengan menggunakan algoritma regresiilinear, backpropagation, dan fuzzy mandani dalam melakukan prediksiihargaaemas. Ketiga algoritma ini dibandingkan berdasarkan nilai persentasi akurasi yang didapatkan. Berdasarkan hasil penelitian didapatkan bahwa dari hasil prediksi ketiga algoritma bahwa algoritma *backpropagation* merupakan algoritma yang paling baik dalam memprediksi emas, kemudian dilanjutkan dengan algoritma regresi linier. Sedangkan untuk algoritmaa*fuzzy mamdanii*tidak dapat melakukan prediksiiharga emas secaraabaik, yang dibuktikanndengan nilai akurasiiyang tidak sampaii1%.

## 2.2 Pendaftaran Calon Mahasiswa Baru

Setiap perguruan tinggi melakukan rutinitas pembukaan pendaftaran calon mahasiswa baru di setiap tahunnya. Rutinitas ini juga dilakukan oleh Institut Teknologi Del disetiap tahunnya dengan membuka beberapa jalur yaitu PMDK, USM 1, USM 2, USM 3, dan jalur nilai UTBK (Del, 2022). Jumlah pendaftar calon mahasiswa baru pun setiap tahunnya berubah dan perubahan yang terjadi pun tidak konsisten.



Gambar 2. 1 Jumlah Pendaftar Calon Mahasiswa Baru IT Del 2016-2022

Sumber : Data BAAK IT Del

Berdasarkan hasil diagram di atas bahwa jumlah pendaftar calon mahasiswa IT Del selalu mengalami perubahan untuk setiap tahunnya. Perubahan jumlah yang selalu naik turun ini juga berlaku ketika kita melakukan *breakdown* data jumlah pendaftar berdasarkan asal sekolah pendaftar.

## 2.3 Machine Learning

*Machine learning* ditemukan pertama kali pada tahun 1959 oleh Arthur Samuel melalui jurnalnya yang berjudull “*Some Studies in Machine Learning Using the Game of Checkers*” (IBM Journal of Research and Development). *Machine learning* adalah salah satu cabang ilmu komputer yang memberikan kemampuan untuk belajar tanpa diprogram secara langsung (eksplisit).

*Machine Learning* membutuhkan model untuk didefinisikan sesuai dengan parameter tertentu. Proses *learning* adalah eksekusi program komputer untuk mengoptimalkan parameter model dengan menggunakan data *training* atau *past experience.* Sederhananya, machine learning merupakan sebuah pemrograman komputer untuk mencapai kriteria atau kinerja tertentu dengan menggunakan sekumpulan data *training* atau pengalaman di masa lalu (*past experience*).

*Machine* *learning* mempelajari agar komputer mampu “belajar” dari data. *Machine learning* melibatkan berbagai disiplin ilmu seperti statistika, ilmu komputer, matematika, dan bahkan neurologi. *Machine learning* menggunakan teori-teori statistika untuk membentuk model matematika. Model dapat bersifat *predictive* atau *descriptive*, atau bisa juga gabungan dari *predictive* dan *descriptive*. Secara umum, algoritma *machine learning* dikelompokkan menjadi empat kategori, yaitu:

1. *Supervised learning*

*Supervised learning* merupakan algoritma *machine learning* yang mengidentifikasi fitur secara eksplisit dan melakukan prediksi atau klasifikasi yang sesuai. *Supervised learning* menggunakan *data* *training* yang sudah diberi label untuk mempelajari *mapping* *function*, dari *input* variabel (*x*) ke *output* variabel (*y*). Permasalahan-permasalahannyang terkait dengan *supervised learningg*dapat dikategorikannmenjadi duaajenis, yaitu *classificationn*dan *regression*. *Classificationn*bertujuan untukkmemprediksi *outcomee*dari *input*, dimanaavariabel berbentukkkategori-kategori. Contoh : pria/wanita, sakit/sehat, tinggi/rendah, dan sebagainya. Sedangkan *regression* bertujuan untuk memprediksi *outcome* dari *input*, dimana *outcome* variabel berbentuk nilai aktual (*real values*). Contoh : tinggi badan seseorang, curah hujan, dan sebagainya. Beberapa algoritma yang termasuk *supervised learning* antara lain adalah Decision tree, Naive Bayes Classifier, Artificial Neural Network, Support Vector Machine, Linear Regression, Logistic Regression, CART, KNN (K-Nearest Neighbor), dsb.

2. *Unsupervised learning*

*Unsupervised learning* merupakan algoritma *machine learning* yang mengidentifikasi data berdasarkan kepadatan, struktur, segmen serupa, dan fitur serupa lainnya. *Unsupervised* *learningg*menggunakan *unlabeleddtraining dataset* untukkmemodelkan strukturrdari data, sehingga *unsupervised learning* bersifat lebih subjektif dibandingkan dengan *supervised learning*. Permasalahan seputar *unsupervised learning* dapat dikelompokkan menjadi tiga kategori, yaitu *association, clustering, dimensionality reduction*. Beberapa algoritma yang termasuk dalam *unsupervised learning* antara lain K-Means, *Hierarchical Clustering*, *Fuzzy* C-*Means*, *Self-Organizing Map*, dan sebagainya.

3. *Reinforcement learning*

*Reinforcement learning* merupakan algoritma yang mengumpulkan informasi atau pengalaman untuk menentukan tindakan yang baik untuk mencapai target atau tujuan. Dengan algoritma ini, agen dapat mengevaluasi perilaku ideal dalam suatu kasus. Berbeda dengan algoritma *supervised learning* dan *unsupervised learning* yang sudah memiliki tujuan (*goal*), algoritma *reinforcement learning* tidak memiliki tujuan yang jelas. Oleh sebab itu, algoritma *reinforcement learning* ini dipaksa untuk belajar menemukan nilai optimal dengan mencari *trial* dan *error*. Beberapa algoritma yang dikelompokkan ke dalam *reinforcement learning* antara lain Algoritma Genetika *(GA)*, Pemrograman Dinamis *(DP),* Iterasi Kebijakan Umum *(GPI), Monte Carlo*.

4. *Deep Learning*

*Deep learning* merupakan metode learning yang memanfaatkan *artificial neural network* yang berlapis-lapis (*multi layer*). *Artificial neural network* ini dibuat mirip dengan otak manusia, dimana meuron-neuron terkoneksi satu sama lain sehingga membentuk sebuah jaringan neuron yang sangat rumit.

(Primartha, 2018)

## 2.4 Data Preprocessing

Data preprocessing adalah langkah awal data mining yang digunakan untuk mengubah data mentah menjadi format data dan informasi yang lebih efisien dan bermanfaat. Format pada data mentah yang diambil dari berbagai macam sumber seringkali mengalami error, missing value, dan tidak konsisten. Oleh karena itu, perlu dilakukan pengolahan data terlebih dahulu sebelum dilakukan pemodelan agar dapat menghasilkan output yang tepat dan akurat.

Berikut ini adalah tahapan yang dilakukan dalam data preprocessing antara lain.

1. Data Cleaning

Data yang dikumpulkan kemungkinan tidak relevan, terdapat kesalahan, dan duplikasi, sehingga dibutuhkan proses pembersihan data. Dalam tahapan ini, data dibersihkan dengan beberapa proses seperti missing value dan noise.

1. Data Integration

Tahapan ini merupakan lanjutan dari tahap data cleaning yang bertujuan untuk mengintegrasikan data dari sumber yang berbeda. Pada tahap data integration ini, perlu dilihat kembali sumber data yang diperoleh. Hal ini penting dilakukan agar data yang digunakan memiliki format yang sama.

1. Data Reduction

Pada tahapan ini dilakukan pemilahan data dengan volume besar yang akan memakan waktu yang cukup lama. Maka dari itu, perlu dilakukan proses data reduction untuk membatasi kumpulan data, dengan tujuan untuk meningkatkan efisiensi penyimpanan, sekaligus mengurangi biaya dan menghemat waktu. Proses pengurangan atau data reduction dilakukan tanpa mengubah hasil dari analisis data. Pengurangan data ini dapat dilakukan dengan tiga cara, yaitu pengurangan dimensi, pengurangan jumlah, dan kompresi data.

1. Data Transformation

Pada tahapan ini data akan diubah menjadi bentuk yang sesuai dalam proses pemodelan data. Data akan dinormalisasikan, dimana normalisasi ini adalah proses memberikan rentang nilai data tertentu untuk memastikan bahwa data tidak ada yang berlebihan. Tahapan ini perlu dilakukan dikarenakan banyak sumber data yang memiliki format yang berbeda.

Adapun manfaat yang diperoleh dengan melakukan data preprocessing adalah :

1. Mempercepat proses data mining
2. Data yang dikumpulkan menjadi lebih mudah dimengerti
3. Mengurangi representasi dalam data
4. Mempermudah proses machine learning dan analisis data

## 2.5 Label Encoding

Dalam pemrosesan data sendiri, sebuah dataset seringkali memiliki data dengan nilai-nilai kategori pada sebuah kolom atau atribut. Dalam melakukan pemrosesan data dengan menggunakan algoritma pada pembelajaran mesin, perlu dilakukan pengubahan nilai-nilai kategori tadi menjadi nilai numerik. Salah satu teknik yang digunakan yaitu Label Encoder. Label Encoder merupakan teknik pada pembelajaran mesin yang digunakan untuk mengubah data kategori menjadi data numerik.

Pada label encoding, nilai-nilai kategori akan diubah menjadi bilangan bulat dengan menggunakan suatu aturan tertentu. Misalnya nama sekolah tertentu akan diubah menjadi sebuah bilangan tertentu. Dengan demikian seluruh kolom Nama Sekolah pada dataset akan dirubah menjadi kolom bertipe integer. Dan setelah itu dapat digunakan untuk melakukan pemodelan pada pembelajaran mesin (Santoso & Putri, 2023)

## 2.6 Normalisasi

Normalisasi data merupakan bagian dari data transformasi yang digunakan untuk mengubah data menjadi nilai yang mudah dipahami. Adapun tujuan dari normalisasi ini dilakukan yaitu mendapatkan bobot yang sama dari semua atribut data dan tidak bervariasi atau hasil dari pembobotan tersebut tidak terdapat atribut yang lebih dominan atau dianggap lebih utama dari pada yang lain. Teknik ini dibutuhkan ketika data yang digunakan untuk prediksi memiliki rentang yang drastis yang mana data dengan ciri seperti ini dapat memberikan hasil yang kurang akurat (Trivusi, 2022).

Metode normalisasi yang digunakan dalam penelitian ini adalah *min-max normalization*. *min-max* *normalization* merupakan metode normalisasi dengan melakukan transformasi linier terhadap data asli sehingga menghasilkan keseimbangan nilai perbandingan antar data saat sebelum dan sesudah proses, adapun persamaannya adalah sebagai berikut:

(Pers. 2.6.1)

(Nasution et al., 2019)

## 2.7 K-Fold Cross Validation

*K-Fold cross validation* adalah salah satu metode yang digunakan untuk mengevaluasi algoritma dengan membagi jumlah sebanyak *k* ke dalam data trakn dan data test. Cara kerja dari *k-fold cross validation* adalah pertama-tama data dipartisi menjadi *k* bagian yang berukuran sama (atau hampir sama). Kemudian setiap data ke-*k* setiap bagian dijadikan sebagai data testing.



Gambar 2. 2 Skema 10 Fold Cross Validation

Dari **Gambar 2.2** di atas kita dapat melihat langkah-langkah proses pembagian data menjadi 10 bagian. Pada bagian pertama atau *fold*-1 data ke-1 dijadikan sebagai data uji dan sisanya menjadi data latih. Kemudian pada *fold*-2 data ke-2 dijadikan sebagai data uji dan sisanya menjadi data latih, hal ini terjadi sampai mencapai nilai fold-10 .

## 2.8 *Grid Search*

Metode *Grid Search* merupakan alternatif untuk menemukan parameter terbaik dari sebuah model. Metode ini dikategorikan sebagai metode lengkap untuk nilai parameter terbaik yang harus dieksplorasi masing-masing dengan menetapkan jenis prediksi terlebih dahulu. Kemudian metode akan menunjukkan skor untuk setiap nilai parameter untuk mempertimbangkan mana yang akan dipilih (Gunawan et al., 2020).

Metode *Grid Search* memiliki cara kerja kombinasi parameter satu persatu dan membandingkan nilai galat terkecil pada parameter tersebut. Pasangan kombinasi dari parameter terlebih dahulu disimpan dalam grid-grid. Baris ke-i dan kolom ke-j yang nilai galatnya terkecil merupakan kombinasi parameter ke-i dan parameter ke-j terpilih (Saputra et al., 2019).

## 2.9 Support Vector Regression (SVR)

Support Vector Regression (SVR) merupakan penerapan dari algoritma Support Vector Machine (SVM) untuk kasus regresi. Algoritma yang diperkenalkan oleh Vapink pada tahun 1995 ini didasarkan pada *risk minimization*, yang dipergunakan untuk mengestimasi sebuah fungsi dengan cara meminimalkan batas atas generalization error.

Adapun keunggulan dari algoritma SVR yaitu:

1. SVR mampu memberikan model pelatihan lebih cepat. Dengan karakteristiknya untuk memecahkan permasalahan linear, maka algoritma ini lebih cocok untuk data set dengan dimensi lebih tinggi
2. SVR mampu menyelesaikan permasalahan minimal lokal dari optimasi nonlinier
3. SVR mampu melakukan penyelesaian norm error pada saat pinalti outlier selama fase pelatihan. Hal ini yang diketahui dengan menggunakan trik kernel
4. SVR efektif untuk melakukan generalisasi sampel data yang sedikit
5. SVR mampu menghindari overfitting

(Saadah et al., 2021)

Tujuan dari Algoritma SVR adalah membuat data lebih acak agar dapat menerima regresi yaitu dengan memetakan pada dimensi yang lebih tinggi (Ridwan et al., 2021). Pada regresi linear persamaan yang digunakan dijabarkan sebagai berikut:

(Pers. 2.9.1)

dengan :

w = vektor pembobot

x = vektor input berupa data

b = nilai bias

Pada SVR nonlinear pertama-tama data input x dipetakan oleh fungsi ke ruang feature. Dengan demikian fungsi regresi memiliki bentuk umum sebagai berikut:

(Pers 2.9.2)

dengan :

= fungsi yang memetakan x pada sebuah dimensi

w = vektor bobot

b = bias

min = (Pers. 2.9.3)

Adapun tujuan untuk meminimalkan fungsi tersebut adalah untuk membuat fungsi menjadi setipis mungkin dan mengurangi kompleksitas perhitungan waktu sehingga waktu yang digunakan menjadi lebih cepat. Berikut merupakan fungsi agar dapat melakukan generalisasi dengan baik dengan batasan yang harus dipenuhi adalah titik harus berada pada rentang seperti pada persamaan berikut:

<= (Pers. 2.9.4)

<= , dimana i=1,2,... (Pers. 2.9.5)

Pada persamaan yang digunakan untuk meminimalkan fungsi regresi diasumsikan bahwa semua titik yang berada pada rentang disebut dengan istilah *feasible*, sedangkan apabila terdapat titik yang berada diluar rentang maka disebut dengan istilah *infeasible*, dimana dapat ditambahkan variabel slack untuk mengatasi masalah titik yang berada di rentang *f(x)*.



Gambar 2. 3 Ilustrasi SVR

Semua titik yang berada di luar margin akan dikenai penalti sebesar C dengan syarat C > 0. Masalah optimasi untuk meminimalkan fungsi regresi dapat dijabarkan sebagai berikut:

**Gambar 2. SEQ Gambar\_2. \\* ARABIC 3 Fungsi Regresi Pada SVR**

Sumber : (Lumbantobing & Pasaribu, 2018)

C > 0 (Pers. 2.9.6)

dengan :

C = nilai penalti

N = jumlah seluruh data

= titik di luar margin yang berada di atas hyperplane

= titik di luar margin yang berada berikut hyperplane

dengan ketentuan:

(Pers. 2.9.7)

(Pers. 2.9.8)

>= 0, dimana=1,2,... (Pers. 2.9.9)

Sehingga diperoleh fungsi linear SVR seperti berikut:

(Pers. 2.9.10)

Untuk kasus yang non-linear SVR digunakan persamaan seperti berikut:

(Pers. 2.9.11)

dengan :

= fungsi kernel yang digunakan

= dual variabel lagrange multiplier

b = bias

Ada tiga bentuk fungsi kernel pada model SVR, antara lain :

1. Kernel Linear

Kernel ini berfungsi untuk pemetaan ke ruang dimensi yang lebih tinggi dan yang sudah dianalisis sudah terpisah secara linier.

(Pers. 2.9.12)

1. Kernel Polynomial

Kernel ini berfungsi untuk melakukan pemetaan ketika data tidak terpisah secara linier.

(Pers. 2.9.13)

1. Kernel Sigmoid

Kernel ini befungsi untuk melakukan pemetaan data ke dalam ruang fitur yang lebih tinggi.

*k(x,y) = tanh(* (Pers. 2.9.14)

1. Kernel Radial Basis Function (RBF)

Kernel ini dapat mengatasi permasalahan yang bersifat non-linear.

(Pers. 2.9.15)

Penelitian ini akan dilakukan pencarian kernel terbaik untuk melakukan pemodelan. Algoritma SVR sendiri juga memiliki beberapa parameter yang dapat disesuaikan untuk mendapatkan model terbaik dengan nilai error yang rendah. Pada penelitian ini, beberapa parameter yang digunakan yaitu paramer cost, gamma, dan epsilon. Parameter cost merupakan parameter yang mengendalikan seberapa keras SVR untuk melakukan prediksi dengan benar setiap titik data pelatihan, parameter gamma berfungsi untuk mempengaruhi fleksibilitas SVR dalam menentukan batas pemisah antar kelas yang berbeda. Dan untuk parameter epsilon berfungsi untuk memutuskan seberapa sering algoritma harus mencoba tindakan acak dan menjelajahi lingkungan (Handayani et al., 2017).

## 2.10 Artificial Neural Network

*Artificial Neural Network* (ANN) atau jaringan saraf tiruan adalah pemodelan kompleks yang dapat memprediksi bagaimana ekosistem akan merespon perubahan variabel lingkungan, dan berguna dalam memproses informasi, terutama dalam sistem saraf biologis dalam sel otak manusia. Arsitektur jaringan syaraf tiruan memiliki beberapa lapisan, yang mana setiap lapisan memiliki jumlah node dan neuron yang berbeda - beda.



Gambar 2. 4 Arsitektur Algoritma Artificial Neural Network

Berdasarkan gambar di atas terlihat bahwa jaringan syaraf tiruan terdiri dari tiga lapisan layer, diantaranya adalah :

1. Input Layer

Input layer merupakan lapisan yang terdiri dari neuron - neuron yang berfungsi menerima sinyal dari luar dan kemudian meneruskan ke neuron lain dalam jaringan (Yuniarti & Gunawan, 2021).

1. *Hidden layer*

*Hidden layer* merupakan tiruan dari sel - sel syaraf konektor pada jaringan syaraf biologis. yang mana lapisan ini berfungsi untuk meningkatkan kemampuan jaringan dalam memecahkan permasalahan (Yuniarti & Gunawan, 2021).

1. Output Layer

Output layer merupakan tiruan sel-sel syaraf motor pada jaringan syaraf biologis. lapisan ini berfungsi untuk menyalurkan sinyal-sinyal keluaran hasil proses jaringan (Yuniarti & Gunawan, 2021).

## 2.11 Backpropagation

*Backpropagation* adalah salah satu bentuk metode *artificial neural network*. Metode ini menggunakan arsitektur *multilayer* dengan menggunakan metode pelatihan *supervised training* (Situmorang, 2022). Istilah *Backpropagation* mengacu pada bagaimana sebuah *error* dihitung mulai dari *output layer*, kemudian disebarkan ke *hidden layerr*dan berakhir pada *inputt layer*. Pada input layer tidak terjadi prosesskomputasi, namunnterjadi pengiriman sinyal *input* X ke *hiddennlayer*. Pada *hidden layer* dann*output layer* terjadiiproses komputasiiterhadap bobottdan bias, selain itu dilakukannperhitungan terhadap hasil dari *hidden layerr*ke *output layer* tersebuttdengan algoritma *Backpropagation*.

Langkah-langkah yang perlu dilakukan untuk pelatihan Backpropagation Neural Network adalah sebagai berikut :

**Fase 0 :**

Langkah 1 : Melakukan inisialisasi bobot, bias, *learning rate*, dan *epoch*.

Langkah 2 : Jika kondisi berhenti tidak dipenuhi, maka kerjakan langkah 3 dan

seterusnya, yang mana dikatakan berhenti jika nilai *error* “<=” dari nilai target *error* (*threshold*) atau jika jumlah *epoch* “>=” maksimum *epoch* (iterasi).

**Fase 1 : Feed Forward Propagation**

Langkah 3 : Setiap unit pada *input layer* (p = 1, 2, 3, ….., n) akan menerima sinyal

*inputan* () untuk diteruskan ke masing-masing *unit* yang berada di *hidden layer*.

Langkah 4 : Menghitung keluaran dari setiap *unit* tersembunyi (q = 1, 2, 3, …. , n)

dengan menggunakan *dot product*, penambahan matrix, dan fungsi

aktivasi.

Rumus perhitungan menggunakan *dot product* dan penambahan matrix :

+

(Pers. 2.11.1)

dengan :

= keluaran untuk unit

= unit ke -q pada *hidden layer*

= neuron ke-p pada *input layer*

= bobot *neuron* *input* ke *neuron hidden*

= bias *neuron hidden* ke-q

Rumus perhitungan untuk mengaktifkan fungsi aktivasi dari nilai keluaran di masing-masing *neuron hidden* dengan menggunakan persamaan sigmoid :

) =

(Pers. 2.11.2)

dengan :

= sinyal yang dikirim untuk *layer* berikutnya

e = nilai eksponensial dengan 2,718281….

Langkah 5 : Lakukan hal yang sama untuk menghitung keluaran setiap *unit* pada

layer berikutnya hingga mencapai keluaran pada *output layer* dengan

menggunakan rumus berikut :

 . +

(Pers. 2.11.3)

Gunakan fungsi aktivasi untuk menghitung sinyal output, seperti pada persamaan :

) =

(Pers. 2.11.4)

dengan :

= *unit* ke-r pada *output layer*

= keluaran untuk unit

= sinyal yang dikirim dari *hidden layer* ke *output layer*

= bobot *neuron hidden* ke *neuron output*

= bias *neuron output* ke-r

= keluaran untuk *output layer* yang akan menjadi nilai prediksi

e = nilai eksponensial dengan 2,718281….

Langkah 6 : Menghitung nilai *error/loss* dengan menggunakan persamaan *Mean Square Error* (MSE). Bentuk persamaan dari MSE adalah sebagai berikut.

*loss =*  (Pers. 2.11.5)

dengan :

aktual = nilai aktual

= nilai prediksi

Bila nilai *error* yang dihasilkan > target *error* (*threshold*), lakukan *backpropagation* untuk memperbaharui parameter. Sebaliknya jika nilai error yang dihasilkan < target error maka nilai bobot dan bias disimpan untuk digunakan dalam pemodelan.

(Situmorang, 2022)

**Fase 2 : Backporpagation**

Langkah 7: Pencarian nilai dari *local gradient* dari setiap unit-unit di *output layer* dan *hidden layer*

Pada tahap *backpropagation* ini hal yang paling utama adalah menyesuaikan bobot yang akan menghasilkan sebuah output atau keluaran dengann *error* yang paling minimal. Dalam menyesuaikan bobot perlu dilakukan pencarian nilai dari *local gradient* dari setiap unit-unit di *output layer* dan *hidden layer*.

Pada pencarian *local gradient* di *hidden layer* dengan *output layer* berbeda, dilakukan pencarian dengan persamaan berikut :

. (Pers. 2.11.6)

dengan :

aktual = nilai aktual

= nilai prediksi

= turunan aktivasi keluaran untuk unit

= selisih nilai aktual dan nilai prediksi

Langkah 8 : Menghitung delta bobot dan delta bias pada *output layer*

Selanjutnya dilakukan perhitungan delta bobot atau delta *weight* dan delta bias pada *output layer*. Perhitungan delta bobot ditunjukkan dengan persamaan:

(Pers. 2.11.7)

dengan :

= menghitung korelasi bobot

= *learning rate*

= *local gradient* dikalikan dengan nilai dari unit pada

*hidden layer*.

Untuk menghitung nilai dari delta bias ditunjukkan dengan persamaan :

(Pers. 2.11.8)

dengan :

= menghitung delta bias

= *learning rate* dengan *local gradient* pada *output*

Langkah 9 : Menghitung informasi *local gradient* di *hidden layer*

Pada *hidden layer* juga dilakukan perhitungan *local gradient* dan delta bobot, namun pada *hidden layer* terdapat perbedaan dalam perhitungan keduanya. Pada unit-unit *hidden layer* melakukan penjumlahan dari perkalian antara *local gradient output layer* dengan *weight*. Persamaan untuk menghitung informasi *local gradient* di *hidden layer* ditunjukkan dengan persamaan berikut ini :

(Pers. 2.11.9)

dengan :

= bobot *neuron hidden* ke *neuron output*

= menghitung *local gradient* pada *output layer*

= turunan fungsi aktivasi pada *hidden layer*

Langkah 10 : Selanjutnya dilakukan perhitungan delta bobot dan delta bias pada *hidden layer*. Perhitungan delta bobot dihitung dengan persamaan :

(Pers. 2.11.10)

dengan :

= delta bobot pada *hidden layer*

= *learning rate*

= *local gradient* dikalikan dengan *input* data

Untuk menghitung nilai dari delta bias ditunjukkan dengan persamaan :

(Pers. 2.11.11)

dengan :

= menghitung delta bias

= *learning rate*

= *local gradient* pada *hidden layer*

**Fase 3: Update Bobot dan Bias**

Langkah 11 : Perubahan nilai bobot dan nilai bias

Pada perubahan bobot di output layer dan *hidden layer* dilakukan penjumlahan nilai dari bias lama dijumlahkan dengan nilai delta bobot. Pada setiap unit-unit ouput layer dilakukan perubahan bobot yang ditunjukkan dengan persamaan:

(Pers. 2.11.12)

dengan :

= delta bobot baru pada *output layer*

= nilai bobot lama pada *output layer*

= delta bobot pada *output layer*

Untuk menghitung perubahan nilai bias ditunjukkan dengan persamaan :

(Pers. 2.11.13)

dengan :

= delta bias baru pada *output layer*

= nilai bias lama pada *output layer*

= delta bias pada *output layer*

Pada *hidden layer* juga dilakukan perubahan nilai bobot dan perubahan nilai bias. Pada setiap unit-unit *hidden layer* dilakukan perubahan nilai bobot yang ditunjukkan dengan persamaan:

(Pers. 2.11.14)

dengan :

= delta bobot baru pada *hidden layer*

= nilai bobot lama pada *hidden layer*

= delta bobot pada *hidden layer*

Untuk menghitung perubahan nilai bias ditunjukkan dengan persamaan :

(Pers. 2.11.15)

dengan :

= delta bias baru pada *hidden layer*

= nilai bias lama pada *hidden layer*

= delta bias pada *hidden layer*

Langkah 12 : Test kondisi berhenti

Setelah dilakukan perbaikan bobot dan bias dilakukan test kondisi berhenti. Selain berdasarkan nilai error seperti pada langkah 6, perhitungan akan berhenti ketika nilai *epoch* (iterasi) yang dilakukan sudah mencapai nilai maksimum *epoch* yang sudah ditentukan.

Kelebihan dari algoritma *Backpropagation* ini antara lain adalah :

1. Algoritma *Backpropagation* mudah, cepat, dan simple untuk diprogram
2. Hanya perlu mengatur nilai inputan, tanpa mengatur parameter lain
3. Algoritma fleksibel dan efisien dalam penggunaannya.
4. Pengguna algoritma tidak perlu mempelajari fungsi spesial.

(Ishukatiyar, 2023)

## 2.12 Early Stopping

Pada saat melakukan pelatihan neural network, sangat penting dilakukan pencarian cara untuk mengindari *overfitting* pada data pelatihan. *Overfitting* mengacu pada model yang mempelajari data latih data latih dengan sangat baik (Mahyunis, 2022) . Keadaan *overfitting* dapat dilihat pada grafik nilai loss di bawah ini.



Gambar 2. 5 Model Overfitting

Berdasarkan **Gambar 2.5** di atas, *overfitting* terjadi ketika nilai *loss* pada data latih sudah membaik sedangkan nilai *loss* pada data uji mengalami kenaikan dibandingkan nilai *loss* data latih. Salah satu *hyperparameter* yang mempengaruhi keadaan model yang *overfitting* atau tidak adalah jumlah *epoch*. Jika jumlah *epoch* yang digunakan terlalu banyak, maka kemungkinan neural network akan *overfitting*. Di sisi lain, jika *epoch* yang digunakan terlalu sedikit, neural network memiliki kemungkinan untuk tidak memiliki kesempatan untuk mempelajari data latih sepenuhya (*Early Stopping*, 2020). *Early stopping* adalah teknik optimasi yang digunakan untuk mengatasi *overfitting* tanpa mengorbankan akurasi model. *Early stopping* akan menghentikan pelatihan ketika model mulai mengalami *overfitting* (Mustafeez, 2023).

## 2.13 Performansi Model

Evaluasi hasil pada sebuah sistem adalah sebuah hal yang sangat penting untuk mengetahui tingkat keberhasilan dari suatu sistem. Pengukuran evaluasi hasil ini dapat dilakukan untuk menentukan model terbaik yang dapat digunakan. Untuk mengukur performa dari model prediksi yang digunakan adalah dengan membandingkan nilai aktual dan nilai prediksi.

### 2.13.1 Mean Square Error

*Mean Square Error* (MSE) merupakan model evaluasi yang digunakan untuk

melakukan evaluasi terkait hasil yang diberikan oleh model dalam meramalkan

nilai atribut dependen yang merupakan data numeric berdasarkan nilai atribut

independen.Untuk melakukan perhitungan *Mean Square Error* dapat dilakukan dengan menggunakan persamaan berikut.

(Pers. 2.13.1)

*Mean Square Error* (MSE) digunakan untuk memeriksa kedekatan nilai prediksi atau *forecast* dengan nilai sebenarnya. Semakin rendah nilai MSE maka akan semakin menunjukkan kecocokan yang lebih baik antara nilai ramalan dengan nilai aktual.

### 2.13.2 Root Mean Square Error

*Root Mean Square Error* (RMSE) merupakan model evaluasi hasil yang menghitung tingkat kesalahan hasil prediksi dengan mencari nilai akar MSE antara nilai prediksi dengan nilai aktual yang diinterpretasikan dengan persamaan berikut ini.

(Pers. 2.13.2)

(Pers. 2.13.3)

Semakin kecil nilai RMSE yang dihasilkan model, maka nilai prediksi semakin akurat (Andriyani, 2021). RMSE yang lebih tinggi menunjukkan bahwa ada penyimpangan yang besar dari residual ke kebenaran dasar.

## 2.14 Flask

Flask merupakan sebuah *web framework* yang diimplementasikan menggunakan bahasa pemrograman Python dan tergolong sebagai salah satu jenis *microframework* dikarenakan tidak memerlukan suatu alat atau pustaka tertentu dalam penggunaannya.Framework yang dipelopori oleh Armin Ronacher ini jauh lebih ringan dan cepat karena dibuat dengan ide menyederhanakan inti framework-nya seminimal mungkin. Flask berfungsi sebagai kerangka kerja aplikasi dan tampilan dari suatu web. Dengan menggunakan Flask dan bahasa Python, pengembang dapat membuat sebuah web terstruktur dan dapat mengatur *behavior* suatu *web* dengan lebih mudah (Somya, 2018).

Sebagian besar fungsi dan komponen umum seperti validasi form, database, dan sebagainya tidak terpasang secara default pada Flask dikarenakan fungsi dan komponen-komponennya sudah dapat menggunakan ekstensi yang membuat fitur dan komponen-komponen tersebut seakan diimplementasikan oleh flask sendiri.

Tahapan singkat mengenai pembuatan *website* sederhana menggunakan *flask* :

1. Melakukan instalasi *python.*
2. Melakukan *setting Environment* atau yang sering disebut dengan mengatur *virtual environment* secara sederhana.
3. Melakukan *instalasi* *flask* melalui pip pada *command prompt*.
4. Membuat *code server*
5. Membuka URL hasil *coding.*

# BAB 3 ANALISIS

Pada bab ini akan dijelaskan mengenai analisis dan perancangan yang akan dilaksanakan dalam pengerjaan penelitian Tugas Akhir ini.

## 3.1 Analisis Domain

Penenitian ini bertujuan untuk menghasilkan model untuk melakukan prediksi jumlah pendaftar calon mahasiswa baru di Institut Teknologi Del. Hasil penelitian ini diharapkan dapat membantu pihak institusi untuk melakukan promosi pada sekolah-sekolah yang nilai pendaftar dari asal sekolah tersebut masih sedikit. Dengan menggunakan data pendaftar pada masa PMB IT Del tahun 2016 sampai 2022, maka akan dilakukan penggunaan *Machine Learning* untuk mempelajari data yang ada.

Algoritma *Machine Learning* yang digunakan dalam penelitian ini ada dua yaitu, algoritma *Support Vector Regression* (SVR) dan algoritma *Artificial Neural Network* (ANN). Kedua algoritma ini akan digunakan untuk melakukan prediksi dengan tujuan untuk melihat bentuk algoritma yang dapat mempelajari bentuk data yang diberikan. Bentuk data yang akan digunakan pada penelitian ini adalah bentuk data *time series*, dimana kedua model algoritma akan mempelajari pola jumlah pendaftar dari tahun sebelumnya dan meramalkan berdasarkan pola tersebut. Pembagian data dilakukan dengan menggunakan metode *K-Fold Cross-Validation* untuk menemukan jumlah perbandingan *train set* dan *test set* yang paling optimal untuk kedua algoritma. Setelah dilakukan pembentukan model dengan algoritma-algoritma tersebut, akan dilakukan perbandingan hasil berdasarkan nilai RMSE terkecil dan menentukan model algoritma yang terbaik berdasarkan nilai RMSE tersebut.

## 3.2 Analisis Data

Pada subbab ini akan menjelaskan tentang data yang akan digunakan untuk mengetahui prediksi jumlah pendaftar calon mahasiswa baru. Data yang digunakan untuk memprediksi jumlah pendaftar merupakan data pendaftar calon mahasiswa baru yang diolah menjadi dataset yang dapat digunakan untuk pemodelan algoritma. Adapun sumber data dan keterangan lain dari data akan dijelaskan pada subbab berikut ini.

### 3.2.1 Sumber Data

Sumber data yang dibutuhkan dalam penelitian ini yang merupakan penentu metode serta analisis pengolahan data diperoleh dari pusat akademik (BAAK) Institut Teknologi Del. Data yang diperoleh dari pusat akademik Institut Teknologi Del adalah data informasi peserta calon mahasiswa baru yang mendaftar melalui jalur PMDK, USM 1, USM 2, USM 3, USM 4, dan jalur UTBK. Adapun informasi yang didapatkan dari data tersebut adalah nama peserta, alamat peserta, nama orang tua peserta ,asal sekolah peserta, prodi pilihan peserta baik pilihan 1 maupun pilihan 2.

Tabel 3. 1 Informasi yang diperoleh dari BAAK IT Del

|  |  |  |
| --- | --- | --- |
| **Variabel** | **Type data** | **Deskripsi** |
| No Pendaftar | Integer | No peserta pendaftar calon mahasiswa baru |
| Nama Siswa | String | Nama peserta calon mahasiswa baru |
| Asal Sekolah | String | Sekolah asal peserta pendaftar calon mahasiswa baru |
| Pilihan I | Categorical | Jurusan pilihan pertama calon peserta |
| Pilihan II | Categorical | Jurusan pilihan kedua calon peserta |
| Jenis Kelamin | Categorical | Jenis kelamin pendaftar calon mahasiswa baru |
| No Telepon | Integer | Nomor telepon dari peserta calon mahasiswa baru |

### 3.2.2 Seleksi Data

Dalam penelitian ini akan dilakukan penyeleksian data yang akan digunakan dalam prediksi. Seleksi data ini dilakukan untuk membentuk dataset yang akan digunakan dalam membentuk model dan prediksi. Berdasarkan data pendaftar yang telah diterima, data yang digunakan adalah tahun pendaftar dan data asal sekolah peserta.

## 3.3 Preprocessing Data

Preprocessing data adalah suatu teknik yang dilakukan untuk mengolah data dengan tujuan data tersebut siap digunakan pada pemodelan dengan algoritma yang akan digunakan. Pada penelitian ini tahapan preprocessing data yang dilakukan peneliti adalah sebagai berikut.

### 3.3.1 Data Integration

Pada tahapan ini peneliti melakukan penggabungan data jumlah pendaftar yang diperoleh dari jalur seluruh jalur yang dibuka oleh institusi dan data pendaftar dari tahun 2016 sampai 2022. Berdasarkan penggabungan data ini dihasilkan dataset yang berisi nama sekolah dan jumlah pendaftar dari asal sekolah untuk setiap tahunnya. Berdasarkan proses data *integration* yang telah dilakukan maka diperoleh dataset dengan tiga atribut sebagai berikut ini :

Tabel 3. 2 Atribut pada dataset

|  |  |  |
| --- | --- | --- |
| **Atribut** | **Tipe Data** | **Deksripsi** |
| Nama Sekolah | Character | Nama asal sekolah yang mendaftar |
| Jumlah Pendaftar pada tahun 2016 | Numerik | Jumlah pendaftar pada tahun 2016 |
| Jumlah Pendaftar pada tahun 2017 | Numerik | Jumlah pendaftar pada tahun 2017 |
| Jumlah Pendaftar pada tahun 2018 | Numerik | Jumlah pendaftar pada tahun 2018 |
| Jumlah Pendaftar pada tahun 2019 | Numerik | Jumlah pendaftar pada tahun 2019 |
| Jumlah Pendaftar pada tahun 2020 | Numerik | Jumlah pendaftar pada tahun 2020 |
| Jumlah Pendaftar pada tahun 2021 | Numerik | Jumlah pendaftar pada tahun 2021 |
| Jumlah Pendaftar pada tahun 2022 | Numerik | Jumlah pendaftar pada tahun 2022 |

Sehingga Dataset yang dihasilkan adalah sebagai berikut:

Tabel 3. 3 Dataset hasil integration

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nama Sekolah** | **Jumlah Pendaftar** | | | | | | |
| 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 |
| SMA Bintang Timur Balige | 70 | 54 | 53 | 110 | 46 | 48 | 53 |
| SMA Bintang Timur Pematang Siantar | 30 | 19 | 6 | 7 | 18 | 8 | 2 |
| SMK Bintang Timur Pematang Siantar | 7 | 6 | - | 1 | 2 | 1 | 1 |

### 3.3.2 Data Cleaning

Pada tahapan ini data yang telah digabungkan dilakukan pembersihan data. Pembersihan data atau data cleaning yang dilakukan oleh peneliti adalah *missing value*. *Missing value* dilakukan disaat adanya penemuan tidak ada data/nilai yang terdeteksi untuk suatu atribut atau fitur pada beberapa sampel pengamatan di dataset. Adanya *missing value* dapat memberikan pengaruh pada hasil prediksi. *Missing value* dapat diatasi dengan beberapa teknik, diantaranya yaitu menghapus kolom atau baris yang memiliki *missing value*  dan mengisi *missing value.*

Pada penelitian ini, *missing value* dilakukan dengan memberikan nilai default di dalam data yang kosong. Nilai default yang diberikan adalah 0. Pada dataset yang dimiliki, kita bisa menemukan atribut yang memiliki nilai *missing value* adalah data yang memiliki tanda strip (-). Atribut-atribut yang memiliki nilai *missing value* antara lain adalah jumlah pendaftar 2016, jumlah pendaftar 2017, jumlah pendaftar 2018, jumlah pendaftar 2019, jumlah pendaftar 2020, jumlah pendaftar 2021, dan jumlah pendaftar 2022.

Tabel 3. 4 Dataset sebelum dilakukan missing value

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jumlah Pendaftar 2016 | Jumlah Pendaftar 2017 | Jumlah Pendaftar 2018 | Jumlah Pendaftar 2019 | Jumlah Pendaftar 2020 | Jumlah Pendaftar 2021 | Jumlah Pendaftar 2022 |
| 18 | 15 | 6 | 5 | 12 | 16 | 1 |
| 16 | 40 | 39 | 19 | 17 | 22 | 11 |
| 32 | 38 | 18 | 23 | 28 | 21 | 7 |
| - | - | - | 2 | 4 | - | - |
| 28 | 24 | 63 | 38 | 45 | 44 | 32 |
| 22 | 45 | 17 | 12 | 6 | 10 | 6 |
| 1 | - | - | - | - | - | - |
| 2 | - | 12 | 3 | - | 2 | 3 |
| - | 1 | - | - | - | - | 12 |
| - | 2 | - | 2 | - | - | 2 |
| 1 | 4 | 1 | 1 | 1 | 4 | - |
| 3 | 1 | - | 2 | 1 | 2 | - |
| 1 | 1 | 1 | 1 | 6 | 1 | - |
| 53 | 48 | 58 | 28 | 17 | 24 | 8 |
| 1 | 3 | 2 | 3 | - | 2 | 8 |
| 1 | - | - | 1 | - | - | - |
| 28 | 43 | 68 | 6 | 2 | 21 | 4 |

Tabel 3. 5 Dataset sesudah dilakukan missing value

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jumlah Pendaftar 2016 | Jumlah Pendaftar 2017 | Jumlah Pendaftar 2018 | Jumlah Pendaftar 2019 | Jumlah Pendaftar 2020 | Jumlah Pendaftar 2021 | Jumlah Pendaftar 2022 |
| 18 | 15 | 6 | 5 | 12 | 16 | 1 |
| 16 | 40 | 39 | 19 | 17 | 22 | 11 |
| 32 | 38 | 18 | 23 | 28 | 21 | 7 |
| 0 | 0 | 0 | 2 | 4 | 0 | 0 |
| 28 | 24 | 63 | 38 | 45 | 44 | 32 |
| 22 | 45 | 17 | 12 | 6 | 10 | 6 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 12 | 3 | 0 | 2 | 3 |
| 0 | 1 | 0 | 0 | 0 | 0 | 12 |
| 0 | 2 | 0 | 2 | 0 | 0 | 2 |
| 1 | 4 | 1 | 1 | 1 | 4 | 0 |
| 3 | 1 | 0 | 2 | 1 | 2 | 0 |
| 1 | 1 | 1 | 1 | 6 | 1 | 0 |
| 53 | 48 | 58 | 28 | 17 | 24 | 8 |
| 1 | 3 | 2 | 3 | 0 | 2 | 8 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 28 | 43 | 68 | 6 | 2 | 21 | 4 |

### 3.3.3 Data Encoding

Pada tahapan ini peneliti melakukan pengubahan variabel teks ke dalam bentuk numerik dengan tujuan agar variabel tersebut dapat diolah dalam pembelajaran mesin. Pada data yang ada, variabel yang diubah adalah variabel nama sekolah. Pengubahan bentuk data teks ini dilakukan dengan menggunakan teknik encoding, dimana data nama sekolah diubah ke dalam bentuk numerik 1,2,3,.., dst.

### 3.3.4 Normalisasi Dataset

Setelah dilakukan encoding tahapan *preprocessing* selanjutnya adalah melakukan normalisasi data agar nilai data memiliki range yang sama sehingga *output* yang didapatkan model dapat mengklasifikasikan data secara tepat (Jabbar, 2022).

Berikut adalah data yang akan dilakukan normalisasi.

Tabel 3. 6 Sampel Dataset sebelum normalisasi

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nama Sekolah | Jumlah Pendaftar 2016 | Jumlah Pendaftar 2017 | Jumlah Pendaftar 2018 | Jumlah Pendaftar 2019 | Jumlah Pendaftar 2020 | Jumlah Pendaftar 2021 | Jumlah Pendaftar 2022 |
| SMA Kampus FKIP HKBP Nomensen | 18 | 15 | 6 | 5 | 12 | 16 | 1 |
| SMA Kristen Kalam Kudus Pematang Siantar | 16 | 40 | 39 | 19 | 17 | 22 | 11 |
| SMA Budi Mulia Pematang Siantar | 32 | 38 | 18 | 23 | 28 | 21 | 7 |
| SMA Yadika 6 | 0 | 0 | 0 | 2 | 4 | 0 | 0 |
| SMA Bintang Timur Pematang Siantar | 28 | 24 | 63 | 38 | 45 | 44 | 32 |

Dari sampel dataset di atas terlihat bahwa nilai dari setiap data memiliki rentang nilai yang tidak sama, sehingga sebelum melakukan pemodelan perlu dilakukan normalisasi terlebih dahulu.

Perhitungan dilakukan dengan tahapan berikut.

1. Menentukan nilai max dan min dari setiap kolom.

Jumlah Pendaftar 2016 : max =32, min=0

Jumlah Pendaftar 2017 : max =40, min=0

Jumlah Pendaftar 2018 : max = 63, min=0

Jumlah Pendaftar 2019 : max = 38, min = 2

Jumlah Pendaftar 2020 : max = 45, min = 4

Jumlah Pendaftar 2021 : max = 44, min = 0

Jumlah Pendaftar 2022 : max = 32, min = 0

1. Melakukan perhitungan setiap kolom dengan menggunakan persamaan *min-max normalization* pada persamaan 2.6.1.

Dari hasil perhitungan normalisasi yang dilakukan maka akan didapatkan nilai seperti pada tabel berikut ini.

Tabel 3. 7 Sampel Dataset setelah normalisasi

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nama Sekolah** | Jumlah Pendaftar 2016  (X1) | Jumlah Pendaftar 2017  (X2) | Jumlah Pendaftar 2018  (X3) | Jumlah Pendaftar 2019  (X4) | Jumlah Pendaftar 2020  (X5) | Jumlah Pendaftar 2021  (X6) | Jumlah Pendaftar 2022  (Y) |
| SMA Kampus FKIP HKBP Nomensen | 0,29 | 0,00 | 0,00 | 0,08 | 0,20 | 0,18 | 0,00 |
| SMA Kristen Kalam Kudus Pematang Siantar | 0,19 | 1,00 | 0,58 | 0,47 | 0,32 | 0,35 | 0,32 |
| SMA Budi Mulia Pematang Siantar | 1,00 | 0,92 | 0,21 | 0,58 | 0,59 | 0,32 | 0,19 |
| SMA Yadika 6 | 0,00 | 0,02 | 0,19 | 0,00 | 0,00 | 0,00 | 0,15 |
| SMA Bintang Timur Pematang Siantar | 0,79 | 0,36 | 1,00 | 1,00 | 1,00 | 1,00 | 1,00 |

## 3.4 Analisis Algoritma SVR

SVR merupakan penerapan support vector machine (SVM) pada kasus regresi. Dalam kasus regresi output yang digunakan berupa bilangan riil atau continue. Dalam mengatasi permasalahan overfitting SVR dapat digunakan sehingga mendapatkan nilai performansi yang bagus. Model SVR sering digunakan untuk meminimalkan jumlah square error, yang mana SVR ini merupakan teori yang diadaptasi dari teori machine learning yang sudah digunakan untuk mengatasi masalah klasifikasi.

**Perhitungan SVR**

Untuk melakukan perhitungan terdapat beberapa langkah yang perlu diperhatikan antara lain :

1. Menginisialisasi awal untuk nilai α, C, epsilon, gamma, lambda:

Tabel 3. 8 Inisialisasi nilai parameter

|  |  |
| --- | --- |
| **Inisialisasi Parameter** | |
| C | 1 |
| alpha | 0,5 |
| epsilon | 0,1 |
| gamma | 0,167 |
| lambda | 0,5 |
| alpha star | 1 |
| Bias | 0,5 |

1. Masukkan data uji

Jumlah data yang digunakan untuk hitungan manual yaitu 5 data, dimana 4 data sebagai data latih dan 1 sebagai data uji dan jumlah variabel yang digunakan yaitu 6 variabel.

Data yang telah dilakukan normalisasi dapat dilihat pada tabel berikut:

Tabel 3. 9 Data Uji yang sudah dinormalisasikan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nama Sekolah** | Jumlah Pendaftar 2016  (X1) | Jumlah Pendaftar 2017  (X2) | Jumlah Pendaftar 2018  (X3) | Jumlah Pendaftar 2019  (X4) | Jumlah Pendaftar 2020  (X5) | Jumlah Pendaftar 2021  (X6) | Jumlah Pendaftar 2022  (Y) |
| SMA Kampus FKIP HKBP Nomensen | 0,29 | 0,00 | 0,00 | 0,08 | 0,20 | 0,18 | 0,00 |
| SMA Kristen Kalam Kudus Pematang Siantar | 0,19 | 1,00 | 0,58 | 0,47 | 0,32 | 0,35 | 0,32 |
| SMA Budi Mulia Pematang Siantar | 1,00 | 0,92 | 0,21 | 0,58 | 0,59 | 0,32 | 0,19 |
| SMA Yadika 6 | 0,00 | 0,02 | 0,19 | 0,00 | 0,00 | 0,00 | 0,15 |
| SMA Bintang Timur Pematang Siantar | 0,79 | 0,36 | 1,00 | 1,00 | 1,00 | 1,00 | 1,00 |

Data yang digunakan adalah :

Tabel 3. 10 Data Uji

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **X1** | **X2** | **X3** | **X4** | **X5** | **X6** | **Y** |
|  | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** | **Y** |
| A1 | 0,29 | 0,00 | 0,00 | 0,08 | 0,20 | 0,18 | 0,00 |
| A2 | 0,19 | 1,00 | 0,58 | 0,47 | 0,32 | 0,35 | 0,32 |
| A3 | 1,00 | 0,92 | 0,21 | 0,58 | 0,59 | 0,32 | 0,19 |

1. Pada metode kernel, data tidak dipresentasikan secara individual, melainkan melakukan perbandingan antara sepasang data. Setiap data akan dibandingkan dengan dirinya sendiri dan data lainnya. Maka untuk data latih yang berjumlah 3 data dan data uji yang berjumlah 2 data, maka hasil perbandingan dapat dilihat pada tabel berikut:

Tabel 3. 11 Pasangan Data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 | A5 | A6 |
| A1 | K(A1,A1) | K(A1,A2) | K(A1,A3) | K(A1,A4) | K(A1,A5) | K(A1,A6) |
| A2 | K(A2,A1) | K(A2,A2) | K(A2,A3) | K(A2,A4) | K(A2,A5) | K(A2,A6) |
| A3 | K(A3,A1) | K(A3,A2) | K(A3,A3) | K(A3,A4) | K(A3,A5) | K(A3,A6) |

Semua data dihitung dengan cara yang sama, baris x kolom dengan sehingga dikarenakan adanya 3 data uji, maka diperoleh matriks 3 x 6. Pada tugas akhir ini kernel yang digunakan yaitu kernel radial basis function, sehingga:

(Pers. 3.4.1)

Maka untuk perhitungan dengan kernel rbf yaitu:

**K(A1,A1)** = exp(-0,167||0,29-0,29||2)+exp(-0,167||0,19-0,19||2)+exp(-0,167||1,00-1,00||2) = 3

**K(A1,A2)** = exp(-0,167||0,29-0||2)+exp(-0,167||0,19-1||2)+exp(-0,167||1,00-0,92||2) = 3,881139032

maka untuk hasil perhitungan dengan menggunakan kernel rbf yaitu:

Tabel 3. 12 Hasil Perhitungan RBF

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** |
| **A1** | 3 | 2,88120583 | 2,861995465 | 2,95062577 | 2,96814757 | 2,99798134 |
| **A2** | 2,881139302 | 3 | 2,885417662 | 2,93391731 | 2,900938976 | 2,868062698 |
| **A3** | 2,855984901 | 2,885417662 | 3 | 2,968299609 | 2,95227962 | 2,977779196 |

1. Menghitung matriks dengan rumus:

(Pers. 3.4.2)

Dimana:

Dij = elemen matriks ke-ij

yi= kelas data ke-i

yj= kelas data ke-j

λ = batas teoritis yang akan diturunkan

Contoh perhitungan:

D11 = (0) (0) (4+0,25) = 0

Sehingga didapatkan hasil perhitungan matriks Hessian adalah sebagai berikut:

Tabel 3. 13 Hasil Perhitungan Matriks Hessian

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** |
| **A1** | 0 | 0 | 0 | 0 | 0 | 0 |
| **A2** | 0,4230286645 | 0,4352 | 0,4234667686 | 0,4284331323 | 0,4250561511 | 0,421689620 |
| **A3** | 0,1482260549 | 0,1492885776 | 0,153425 | 0,1522806159 | 0,1517022943 | 0,152622829 |

1. Mencari nilai error menggunakan rumus:

Ei = (Pers. 3.4.3)

Dimana Ei = nilai error data ke-i

E1 = 0 + 0 + 0 + 0 + 0 + 0 = 0

Maka didapatkan setiap nilai error data:

Tabel 3. 14 Hasil Perhitungan Nilai Error

|  | **Error** |
| --- | --- |
| A1 | 0 |
| A1 | 1,278437168 |
| A3 | 0,4537726859 |

1. Menghitung nilai delta alpha

(Pers. 3.4.4)

min{max[0,167(1 - 0),- 0,5],1 - 0,5}

min{max(0,167,- 0,5),0,5}

min{0,167,0,5}

0,167

Hasil perhitungan delta alpha:

Tabel 3. 15 Hasil Perhitungan Delta Alpha

|  |  |
| --- | --- |
|  | **Delta Alpha** |
| **A1** | 0,167 |
| **A2** | -0,04649900706 |
| **A3** | 0,09121996145 |
| **A4** | 0,1195388175 |

Karena nilai maksimum delta alpha adalah 0,167 dan lebih dari epsilon (0,1) maka iterasi berlanjut.

1. Menghitung nilai baru dengan menggunakan rumus:

(Pers. 3.4.5)

Contoh perhitungan:

0,035

Tabel 3. 16 Hasil Perhitungan Nilai Alpha Baru

|  |  |
| --- | --- |
| **Alpha** | |
| A1 | 0,667 |
| A2 | 0,4535009929 |
| A3 | 0,5912199615 |

1. Melakukan pencarian nilai w dengan rumus:

(Pers. 3.4.6)

Contoh perhitungan:

W1 = 0,667-1 = -0,333

W2 = 0,4535009929 - 1 = -0,5464990071

Maka hasil perhitungan w adalah sebagai berikut:

Tabel 3. 17 Hasil Perhitungan Nilai Alpha Star

|  |  |
| --- | --- |
| **Alpha Star** | |
| A1 | -0,333 |
| A2 | -0,5464990071 |
| A3 | -0,4087800386 |

Setelah nilai w ditemukan maka dapat dilakukan perhitungan dalam fungsi pada persamaan () yaitu:

(Pers. 3.4.7)

Nilai bias(b) merupakan nilai yang diambil secara random pada komputer. Misalkan inisialisasi nilai b = 0,5

Contoh perhitungan 2 data untuk melakukan testing:

Tabel 3. 18 Data Uji yang Ditesting

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nama Sekolah** | Jumlah Pendaftar 2016  (X1) | Jumlah Pendaftar 2017  (X2) | Jumlah Pendaftar 2018  (X3) | Jumlah Pendaftar 2019  (X4) | Jumlah Pendaftar 2020  (X5) | Jumlah Pendaftar 2021  (X6) | Jumlah Pendaftar 2022  (Y) |
| SMA Kristen Kalam Kudus Pematang Siantar | 0,19 | 1,00 | 0,58 | 0,47 | 0,32 | 0,35 | 0,32 |

Perhitungan kernel (*dot product* baris data testing dengan kolom data training), dengan nilai yang telah didapatkan sebelumnya, maka diperoleh hasil menggunakan fungsi y sebagai berikut:

Tabel 3. 19 Hasil Alpha Star yang didapat sebelumnya

|  |  |
| --- | --- |
|  | |
| A1 | -0,333 |
| A2 | -0,5464990071 |
| A3 | -0,4087800386 |

Untuk data testing maka diperoleh hasil prediksi sebagai berikut:

y = (- 0,333 \* 0,19) + (- 0,333 \* 1,00) + (- 0,333 \* 0,58) + (- 0,333 \* 0,47) + (- 0,333 \* 0,32) + (- 0,333 \* 0,35) + 0,5 = 0,13603

Kemudian dilakukan proses denormalisasi, untuk mendapat kan nilai real kembali. Maka prediksi jumlah pendaftar calon mahasiswa baru tahun 2022 dari sekolah SMA Kalam Kudus Pematang Siantar adalah sebanyak 15 jumlah pendaftar calon mahasiswa.

## 3.5 Analisis Algoritma ANN Backpropagation

Backpropagation merupakan salah satu bentuk metode dalam *artificial neural network* (ANN) yang populer. Metode ini memiliki kemampuan yang unggul untuk belajar, berdaptasi akan suatu objek, dan memiliki toleransi terhadap *error*. Pada prosesnya algoritma *Backpropagation* ini dimulai dari tahapan *feedforward propagation*, kemudian melakukan perhitungan *error* dari *output*, dan dilanjutkan tahapan backpropagation.

* + - 1. **Inisialisasi Bias dan Bobot**

Sebelum dilakukan pemodelan, perlu dilakukan inisialisasi bobot dan bias terlebih dahulu. Bobot dan bias dipilih secara acak yang diambil dari rentang 0 sampai 1, seperti yang tampak pada tabel berikut :

Tabel 3. 20 Nilai bias dari input layer ke *hidden layer*

|  |  |  |  |
| --- | --- | --- | --- |
| (bias, *hidden layer*) | (,1) | (,2) | (,3) |
| (1, ) | 0,48 | 0,50 | 0,65 |

Tabel 3. 21 Nilai bobot dari inputan layer ke *hidden layer*

|  |  |  |  |
| --- | --- | --- | --- |
| (input, *hidden layer*) | (,1) | (,2) | (,3) |
| (1, ) | 0,20 | 0,50 | 0,50 |
| (2, ) | 0,15 | 0,20 | 0,40 |
| (3, ) | 0,30 | 0,10 | 0,35 |
| (4, ) | 0,40 | 0,25 | 0,15 |
| (5, ) | 0,18 | 0,30 | 0,25 |
| (6, ) | 0,22 | 0,45 | 0,30 |

Setelah bias dan bobot dari *input layer* ke *hidden layer* ditentukan selanjutnya kita menentukan nilai bias dan bobot dari *hidden layer* ke *output layer*.

Tabel 3. 22 Nilai bias dari *hidden layer* ke output layer

|  |  |
| --- | --- |
| (bias, output) | (,1) |
| (2, ) | 0,50 |

Tabel 3. 23 Nilai bobot dari *hidden layer* ke output layer

|  |  |
| --- | --- |
| (*hidden layer*, output) | (,1) |
| (1, ) | 0,60 |
| (2, ) | 0,35 |
| (3, ) | 0,50 |

* + - 1. **Inisialisasi *Hyperparameter***

Setelah dilakukan inisialisasi *parameter* bobot dan bias, maka selanjutnya dilakukan inisialisasi *hyperparameter* seperti *learning rate, epoch*, dan target *error* (threshold). Nilai *hyperparameter* yang akan digunakan pada analisis ini adalah *learning rate*=0.1 , epoch=1, target error (*threshold*) = 0.3, dan jumlah neuron pada *hidden layer* =3 neuron.

* + - 1. **Feedforward Propagation**

Pada tahap ini akan dilakukan perhitungan dengan menjumlahkan nilai bobot dan

bias yang telah ditentukan sebelumnya untuk memperoleh nilai keluaran yang

dihasilkan di setiap layer berdasarkan nilai yang dikirim mulai dari input layer ke

*hidden layer* dan kemudian dari *hidden layer* ke output layer. Berikut tahapan feed

forward propagation yang dilakukan:

1. **Feedforward Propagation dari Input Layer ke *Hidden layer***

Nilai data input pada baris kedua (data sekolah SMA Kalam Kudus Pematang Siantar) akan dikirimkan untuk menghasilkan keluaran pada output layer, yang mana sebelum sampai ke output layer, sinyal yang dikirimkan dari input layer akan terlebih dahulu melalui *hidden layer*. Untuk itu pada tahap ini akan dihitung berapa sinyal yang diterima oleh masing-masing unit yang ada di *hidden layer* dengan menggunakan persamaan 2.11.1 .



+



.

+ [ 0.48 0.50 0.65]

= [0.685 0.724 0.954] + [ 0.48 0.50 0.65]

= [1.165 1.224 1.604]

Setelah dilakukan penghitungan sinyal-sinyal inputan terbobot, selanjutnya akan dilakukan penghitungan fungsi aktivasi dari sinyal inputan terbobot. Untuk melakukan perhitungan fungsi aktivasi digunakan persamaan 2.11.2 .

=

=

=

=

1. **Feedforward Propagation dari *Hidden layer* ke Output Layer**

Dari hasil perhitungan *feedforward propagation* dari *input layer* ke *hidden layer,* akan dilanjutkan dengan menghitung nilai keluaran untuk masing-masing unit di *hidden layer* yang di kirimkan ke *output layer*. Persamaan yang digunakan untuk perhitungan ini adalah persamaan 2.11.3 .

. +

 . +

[1.144] +

[1.644]

Selanjutnya, sinyal yang diterima dari *hidden layer* ke *output layer* dihitung kembali dengan menggunakan fungsi aktivasi sigmoid seperti berikut, yang mana hasil dari perhitungan ini akan menjadi nilai y atau prediksi jumlah pendaftar terhadap data yang dilatih. Perhitungan ini menggunakan persamaan 2.11.4 seperti berikut :

= =

=

1. **Menghitung Nilai Error / Loss**

Berdasarkan perhitungan *feedforward propagation* yang dilakukan, diperoleh nilai prediksi jumlah pendaftar untuk data yang dipilih adalah 0.838. Selanjutnya dengan menggunakan nilai prediksi yang dihasilkan, akan dilakukan perhitungan nilai *error* dengan menggunakan *Mean Square Error* (MSE). Bila nilai MSE yang diperoleh lebih kecil dengan nilai target error yang telah ditentukan sebelumnya, maka nilai prediksi tersebut dapat dipastikan akan menjadi hasil akhir. Namun, jika nilai MSE yang diperoleh lebih besar dari nilai target error, maka akan dilakukan pembaruan bobot dan bias dengan menggunakan *backpropagation*.

Dengan menggunakan persamaan 2.11. 5 , nilai *loss* dapat ditentukan seperti berikut ini.

*loss* =

**=**

**= (**0.268)

= 0.134

* + - 1. **Backpropagation**

Karena nilai *loss <* nilai target *error*, yaitu 0.134 < 0.3, maka untuk tahapan backpropagation tidak perlu dilakukan. Sehingga nilai prediksi yang dilakukan pada tahapan sebelumnya merupakan hasil akhir prediksi.

## 3.6 Analisis Evaluasi Model

Setelah dilakukan analisis terhadap kedua model, selanjutnya akan dilakukan evaluasi terhadap kedua model tersebut. Pada evaluasi model akan menggunakan model RMSE (*Root Mean Square Error*). RMSE digunakan untuk mengukur tingkat kesalahan model dalam prediksi suatu nilai numerik. Seperti yang telah disampaikan pada [**subbab 2.13.2**](#_2.11.2__Root) mengenai RMSE, bahwa semakin kecil nilai RMSE yang dihasilkan model, maka nilai prediksi semakin akurat. Evaluasi model RMSE memiliki beberapa kelebihan diantaranya adalah :

1. RMSE memberikan hasil dalam satuan yang sama dengan target, sehingga mudah dipahami dan diinterpretasikan
2. RMSE memperhitungkan semua nilai residual (perbedaan nilai antara nilai aktual dan nilai prediksi) sehingga lebih sensitif terhadap perbedaan besar antara nilai aktual dan nilai prediksi.
3. RMSE menunjukkan performa model secara keseluruhan, bukan hanya performa pada titik tertentu.

## 3.7 Hipotesis Penelitian

Berdasarkan hasil literatur dan hasil analisis dari beberapa algoritma yang sudah dijelaskan diatas, maka peneliti dapat menyusun hipotesis sebagai berikut:

**Hipotesis** : Algoritma Artificial Neural Network (ANN) atau Jaringan Syaraf Tiruan merupakan suatu sistem pemrosesan informasi dengan suatu karaketeristik menyerupai sistem saraf pada manusia yang dapat memecahkan masalah dengan melakukan training data yang besar, dan juga ANN memiliki kemampuan untuk mentoleransi kesalahan sehingga dapat menghasilkan prediksi yang baik (Putra & Ulfa Walmi, 2020). Berdasarkan penelitian yang dilakukan oleh (Abdel-Sattar et al., 2021) dengan judul penelitian ” Application of artificial neural network and support vector regresion in predicting mass of ber fruits (Ziziphus mauritiana Lamk) based on fruit axial dimensions ” dengan hasil penelitian yaitu algoritma ANN mendapatkan hasil yang lebih baik dibandingkan dengan algoritma Support Vector Regression dengan nilai RMSE 1.8479 dan nilai R2 dengan nilai 0.9771 pada dataset yang memiliki tingkat kompleksitas dan atribut yang kompleks. Sehingga dugaan pada penelitian ini algoritma Artificial Neural Network juga memiliki performa terbaik dalam melakukan prediksi jumlah pendaftar di Institut Teknologi Del.

### 3.7.1 Perumusan H0 dan H1

Berdasatkan Hipotesis yang sudah ditentukan, maka Hipotesis Awal (H0) dan Hipotesis Alternatif (H1) adalah sebagai berikut:

|  |  |  |
| --- | --- | --- |
| **H0** | : | model Algoritma Artificial Neural Network lebih baik dibandingkan model Support Vector Regression. |
| **H1** | : | model Algoritma Artificial Neural Network tidak lebih baik dibandingkan model Support Vector Regression. |

# BAB 4 PERANCANGAN

Pada bab ini akan dijelaskan mengenai rancangan penelitian yang dilakukan dalam melakukan pengerjaan Tugas Akhir.

## 4.1 Rancangan Desain Penelitian



Gambar 4. 1 Rancangan desain penelitian yang akan dilakukan

Berikut penjelasan terkait rancangan penelitian yang akan dilakukan :

1. Pengumpulan data

Pada penelitian ini data yang digunakan diperoleh dari kantor pusat akademik (BAAK) Institut Teknologi Del. Data yang diperoleh dari BAAK Institut Teknologi Del adalah data pendaftar calon mahasiswa baru yang mendaftar pada seluruh jalur pendaftaran yang dibuka institusi. Data pendaftar diperoleh secara terpisah berdasarkan jalur pendaftarannya. Data yang dikumpulkan merupakan data mulai dari 2016 sampai dengan 2022.

1. Data Preprocessing

Pada tahapan ini data yang telah dikumpulkan akan dilakukan tahapan preprocessing data untuk mempersiapkan data sebelum digunakan dalam implementasi algoritma. Setelah melewati tahapan ini akan menghasilkan dataset yang telah siap untuk digunakan dalam pemodelan.

1. Normalisasi Data

Pada tahapan ini data yang telah melewati tahapan preprocessing data akan dilanjutkan tahapan normalisasi data. Normalisasi data dilakukan dengan tujuan untuk menyamakan range nilai untuk setiap feature yang digunakan untuk prediksi. Teknik normalisasi data yang digunakan adalah *min-max normalization*.

1. Implementasi Algoritma

Pada tahapan ini data yang telah dinormalisasi akan dimasukkan ke dalam bentuk pemodelan algoritma. Pemodelan algoritma akan dilakukan secara terpisah antara pemodelan algoritma *Support Vector Regression* dan pemodelan algoritma *Artificial Neural Network*. Implementasi setiap algoritma akan dijelaskan pada subbab berikutnya.

1. Evaluasi Model

Pada tahapan ini akan dilakukan evaluasi terhadap kedua model algoritma yang telah dibuat pada tahapan sebelumnya. Evaluasi model ini akan dilakukan pencarian nilai RMSE dan akurasi dari setiap model algoritma.

1. Pemilihan Algoritma Terbaik

Pada tahapan ini akan dilakukan pemilihan algoritma terbaik dari antara algoritm *Support Vector Regression* dan *Artificial Neural Network*. Penentuan algoritma terbaik ini dilihat berdasarkan nilai RMSE terkecil. Seperti yang dijelaskan pada [**subbab 2.13.2**](#_2.13.2__Root), bahwa semakin kecil nilai RMSE yang dihasilkan maka model tersebut akan semakin baik.

1. Pembuatan Simulator

Setelah dilakukan pembentukan model dan penentuan model algoritma terbaik, selanjutnya akan dilakukan pembangunan aplikasi web yang akan digunakan sebagai simulator. Pembentukan aplikasi ini dilakukan dengan menggunakan bahasa pemograman *python* dan framework *flask*. Pada aplikasi akan terdapat fitur SVR dan ANN, dimana pada kedua fitur akan melakukan prediksi dengan bentuk modelnya masing-masing.

## 4.2 Rancangan Pre-Processing Data



Gambar 4. 2 Rancangan pelaksanaan preprocessing data

Berdasarkan **Gambar 4.2** di atas, terlihat bahwa rancangan preprocessing data yang dilakukan dengan penjelasan seperti berikut :

1. Data Integration

Pada tahapan ini data yang telah dikumpulkan akan digabungkan seperti pada [**subbab 3.3.1**](#_3.3.1_Data_Integration).Penggabungan data ini dilakukan secara manual oleh peneliti dan akan menghasilkan bentuk atribut seperti pada **tabel 3.2** sehingga bentuk dataset yang dihasilkan adalah seperti pada **tabel 3.3.**

1. Data Cleaning

Pada tahapan ini, peneliti melakukan data cleaning dengan menggunakan teknik *missing value*. Seperti yang sudah dijelaskan pada [**subbab 3.3.2**](#_3.3.2_Data_Cleaning), bahwa data cleaning dilakukan dengan melakukan pengisian nilai 0 pada data yang tidak memiliki nilai maupun bertanda strip (-).

1. Encoding Data

Pada tahapan encoding data ini, peneliti melakukan pengubahan bentuk data nama sekolah dari bentuk teks ke bentuk numerik dengan tujuan agar nama sekolah dapat digunakan dalam pemodelan. Pelaksanaan *encoding* data ini akan dijelaskan pada [**subbab 5.3.1**](#_5.3.1.__Encoding) dan hasil akan dijelaskan pada [**subbab 6.1**](#_6.1.__Hasil). Hasil akhir yang akan diperoleh dari tahapan *encoding* data adalah dataset baru yang sudah dapat digunakan dan dipelajari oleh model algoritma.

## Rancangan Normalisasi

Tahapan normalisasi ini masih termasuk dalam tahapan *preprocessing data*, namun peneliti memisahkan tahapan pada flow dikarenakan setelah tahapan *encoding* terdapat *output* dataset yang baru dimana pada dataset tersebut variabel nama sekolah sudah memiliki nilai dalam bentuk numerik yang dapat dipelajari *machine learning*.

Pada tahapan normalisasi ini, akan dilakukan dengan memasukkan dataset yang telah di *encoding* kemudian nilai-nilai di dalam dataset akan diubah ke dalam range nilai yang sama seperti yang dijelaskan pada [**subbab 3.3.4**](#_3.3.4_Normalisasi_Dataset).

## 4.4 Rancangan Implementasi Algoritma

Berikut ini adalah rancangan implementasi algoritma yang digunakan dalam penelitian.

### 4.4.1 Rancangan Implementasi Algoritma SVR

Gambar 4. 3 Flow algoritma *Support Vector Regression*



Berdasarkan **Gambar 4.3**diatas, langkah-langkah pengerjaan algoritma *Support Vector* *Regression* dilakukan dengan beberapa tahapan dengan rincian masing-masing tahapan adalah sebagai berikut :

* + - 1. Melakukan pembagian dataset

Dataset yang digunakan akan dilakukan pembagian untuk data latih dan data uji. Pada proses pembagian dataset ini digunakan menggunakan teknik *K-Fold Cross Validation*, yang mana untuk rentang nilai k yang digunakan yaitu mulai nilai k=2 sampai nilai k=10.

* + - 1. Melakukan inisialisasi kernel dan parameter

Kernel yang digunakan pada SVR terdiri dari 4 kernel yaitu linear, RBF, Sigmoid, dan polynomial. Sedangkan untuk inisialisasi Parameter SVR terdiri dari beberapa parameter ε (nilai epsilon untuk zona yang masih dapat diterima), C (cost), γ untuk masukan fungsi kernel yang digunakan dan jumlah iterasi maksimum. Untuk nilai parameter cost akan diinisialisasikan pada nilai 0,1, 1, 10, 100, kemudian parameter epsilon pada nilai 0,01, 0,1, 1 dan untuk parameter gamma berada pada nilai 0,001, 0,01, 0,1, dan 1 (Sepri & Fauzi, 2020).

* + - 1. Melakukan pencarian kernel dan parameter terbaik dengan menggunakan *Gridsearch*

*Gridsearch* akan digunakan untuk menemukan parameter dan kernel terbaik berdasarkan nilai yang telah diinisialisasikan untuk mendapatkan nilai error yang paling rendah.

1. Melakukan pemodelan dengan algoritma SVR

Kernel dan parameter terbaik yang telah didapatkan menggunakan GridSearch kemudian digunakan untuk melakukan pemodelan menggunakan algoritma SVR

1. Melakukan prediksi menggunakan SVR

Model terbaik telah didapatkan dan dapat digunakan untuk melakukan prediksi.

### 4.4.2 Rancangan Implementasi Algoritma ANN Backpropagation



Gambar 4. 4 Flow algoritma ANN *Backpropagation*

Berdasarkan Gambar di atas, langkah implementasi algoritma Backpropagation dilakukan dengan beberapa tahapan dengan rincian dari masing – masing tahapan adalah sebagai berikut:

1. Inisialisasi Nilai

Pada tahapan ini, untuk memulai tahapan pemodelan algoritma perlu dilakukan inisialisasi nilai *hyperparameter* seperti nilai *learning rate*, *epoch*, jumlah neuron *hidden layer*, dan fungsi aktivasi. Beberapa *hyperparameter*, seperti fungsi aktivasi, jumlah neuron pada *hidden layer*, dan *learning rate*, akan dilakukan pencarian nilai terbaik diantara beberapa nilai yang akan digunakan dengan menggunakan teknik *gridsearch*. Berikut adalah nilai-nilai *hyperparameter* yang diinisialisasikan untuk model.

Tabel 4. 1 Inisialisasi *Hyperparameter* ANN

|  |  |
| --- | --- |
| ***Hyperparameter*** | **Nilai** |
| Learning rate | 0.01, 0.001 |
| Jumlah neuron *hidden layer* | 12, 16, 24, 32, 64, 128 |
| Fungsi Aktivasi *hidden layer* | Sigmoid, Tanh, ReLu |

Jika dilihat pada **Tabel 4.1** di atas, bahwa *hyperparameter learning rate,* jumlah neuron pada *hidden layer*, dan fungsi aktivasi didefinisikan dengan beberapa nilai. *Learning* *rate* diinisialisasikan dengan nilai 0.01 dan 0.001 ditentukan dari nilai learning rate yang pernah digunakan pada penelitian sebelumnya (Mahyunis, 2022). Berlandaskan dengan jurnal yang sama dengan *learning rate*, penggunaan fungsi aktivasi yang diinisialisasi juga berdasarkan penelitian sebelumnya, dimana peneliti menggunakan percobaan pada tiga fungsi aktivasi seperti ReLU, sigmoid, dan tahn. Dari keempat bentuk fungsi aktivasi tersebut, peneliti mencoba mencari fungsi aktivasi yang optimal digunakan pada bentuk dataset yang akan dimodelkan. Inisialisasi *hyperparameter* selanjutnya yang diinisialisasi adalah jumlah neuron yang akan digunakan pada *hidden layer*. Nilai-nilai tersebut diambil dari penelitian yang sebelumnya (Rusdi et al., 2021) dan (Rizal, 2014) yang juga melakukan percobaan nilai epoch dengan nilai-nilai tersebut.

1. Pencarian nilai terbaik dengan *Gridsearch*

Pada tahapan ini, nilai-nilai yang sudah diinisialisasi pada tahapan sebelumnya akan diproses oleh library *GridSearch* untuk mencari nilai-nilai yang terbaik dari seluruh nilai yang telah diinisialisasi untuk setiap *hyperparameter*. Parameter yang akan diolah dengan teknik *gridsearch* adalah *learning rate,* jumlah neuron pada *hidden layer,* dan fungsi aktivasi.

1. Pembentukan Model

Pada tahapan ini dilakukan pembentukan model ANN yang mana model terdiri dari input layer, *hidden layer*, dan output layer. Dalam pembentukan model nilai-nilai *hyperparameter* yang telah diinisialisasikan pada tahapan sebelumnya dipanggil untuk digunakan dalam training model.

1. Pelatihan Model

Pada tahapan ini akan dilakukan pelatihan model sesuai dengan bentuk model yang dibuat dan nilai *hyperparameter* yang telah diinisialisasi. Di dalam pembentukan model, akan terjadi tahapan feedforward, perhitungan loss, dan backpropagation yang tidak terlihat secara langsung ketika model dilatih.

Pada tahapan feedforward nilai inputan data akan dimasukkan ke dalam input layer dan akan dihitung nilai output untuk setiap neuron dengan memperhitungankan bobot, bias, dan fungsi aktivasi. Nilai bobot dan bias pada proses ini akan dihasilkan secara otomatis oleh model. Kemudian setiap output akan disimpan untuk digunakan pada tahap berikutnya.

Pada tahapan perhitungan loss, nilai output yang dihasilkan pada tahapan feedforward akan dicari nilai *loss*-nya terhadap nilai target yang diinginkan. Nilai loss ini menggambarkan sejauh mana prediksi jaringan mendekati nilai target sebenarnya.

Pada tahapan backpropagation akan terjadi perhitungan gradien loss terhadap bobot dan bias yang dihasilkan langsung oleh model. Gradien ini mengindikasikan seberapa besar perubahan yang harus diterapkan pada bobot dan bias untuk mengurangi loss.

1. Kondisi Waktu Berhenti

Pada tahapan ini, selama proses train akan dilakukan pengecekan nilai *epoch* dimana ketika iterasi yang terjadi selama proses *training* sudah mencapai nilai *epoch* yang ditentukan maka proses *training* akan dihentikan dan ketika belum mencapai nilai *epoch* maksimum maka *training* akan terus dilakukan

## 4.5 Rancangan Aplikasi Del Predict

Del Predict merupakan aplikasi web yang digunakan untuk melakukan prediksi terhadap jumlah pendaftar calon mahasiswa baru. Pada aplikasi akan terdapat dua fitur yaitu fitur SVR untuk melakukan prediksi dengan algoritma *support vector regression* dan fitur ANN untuk melakukan prediksi dengan algoritma *artificial neural network*. Pembangunan aplikasi ini akan menggunakan tools HTML dan Bootstrap dalam membuat halaman tampilan dan akan menggunakan python dalam membuat server.

### 4.5.1 Use Case Diagram



Gambar 4. 5 Use Case Diagram Aplikasi

Pada **Gambar 4.5** merupakan gambaran fungsionalitas dari aplikasi web yang akan digunakan sebagai simulator sistem prediksi jumlah pendaftar calon mahasiswa baru. Tahapan yang terjadi pada use case diagram di atas akan dijelaskan di dalam tabel berikut berikut ini.

Tabel 4. 2 Deskripsi Use Case Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Use Case** | **Deskripsi** |
| 1 | Prediksi ANN | Use case ini menggambarkan bahwa aplikasi dapat melakukan prediksi dengan menggunakan algoritma ANN. |
| 2 | Prediksi SVR | Use case ini menggambarkan bahwa aplikasi dapat melakukan prediksi dengan menggunakan algoritma SVR. |
| 3 | Memilih nama sekolah | Use case ini menggambarkan bahwa aplikasi melakukan pemilihan nama sekolah yang akan diprediksi |
| 4 | Klik button **Lakukan Prediksi** | Use case ini menunjukkan bahwa setelah user memilih nama sekolah, user mengklik button *lakukan prediksi* untuk memulai proses prediksi. |
| 5 | Menampilkan Hasil Prediksi | Use case ini menggambarkan bawa aplikasi akan menampilkan hasil prediksi pada halaman fitur yang dipilih. |

### 4.5.2 Use Case Scenario

Pada subab ini akan menjelaskan mengenai prosedur dari sistem yang akan dibangun. Berikut ini adalah use case scenario dari aplikasi Del Predict.

Tabel 4. 3 Use Case Scenario Prediksi ANN

|  |  |  |
| --- | --- | --- |
| Use Case | : | Prediksi ANN |
| Deskripsi | : | Use case ini menggambarkan bahwa aplikasi dapat melakukan prediksi ANN |
| Aktor | : | User |
| Kondisi Awal | : | Sistem menampilkan halaman home Del Predict |
| Kondisi Akhir |  | Sistem menampilkan hasil prediksi |
| Skenario | | |
| Aksi Aktor | | Reaksi Sistem |
| 1.User memilih fitur **ANN** | |  |
|  | | 2.Menampilkan halaman prediksi ANN |
| 3.Memilih nama sekolah yang akan diprediksi | |  |
| 4. Mengklik button **Lakukan Prediksi** | |  |
|  | | 5.Menampilkan nilai hasil prediksi |

Tabel 4. 4 Use Case Scenario Prediksi SVR

|  |  |  |
| --- | --- | --- |
| Use Case | : | Prediksi SVR |
| Deskripsi | : | Use case ini menggambarkan bahwa aplikasi dapat melakukan prediksi SVR |
| Aktor | : | User |
| Kondisi Awal | : | Sistem menampilkan halaman home Del Predict |
| Kondisi Akhir | : | Sistem menampilkan hasil prediksi |
| Skenario | | |
| Aksi Aktor | | Reaksi Sistem |
| 1.User memilih fitur **SVR** | |  |
|  | | 2.Menampilkan halaman prediksi SVR |
| 3.Memilih nama sekolah yang akan diprediksi | |  |
| 4. Mengklik button **Lakukan Prediksi** | |  |
|  | | 5.Menampilkan nilai hasil prediksi |

### 4.5.3 Design Aplikasi

Berikut ini merupakan *design* tampilan dari aplikasi yang akan dijadikan sebagai simulator dari pemodelan algoritma.



Gambar 4. 6 Halaman Home



Gambar 4. 7 Halaman Prediksi SVR



Gambar 4. 8 Halaman Hasil Prediksi SVR



Gambar 4. 9 Halaman Prediksi ANN



Gambar 4. 10 Halaman Hasil Prediksi ANN

# BAB 5 IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai lingkungan implementasi, batasan, pengumpulan data, preprocessing data, pembentukan model untuk prediksi, dan pembuatan aplikasi web sebagai simulator prediksi.

## 5.1 Lingkungan Implementasi

Sub-bab ini akan menjelaskan mengenai spesifikasi *hardware* (perangkat keras) dan *software* ( perangkat lunak) yang digunakan dalam proses implementasi selama masa penelitian.

* 1. Hardware

Spesifikasi *hardware* yang digunakan antara lalin :

Tabel 5. 1 Spesifikasi Hardware

|  |  |  |
| --- | --- | --- |
| **No** | **Hardware** | **Spesifikasi** |
| 1 | Laptop | Asus Laptop X507UF |
| 2 | *Processor* | Intel(R) Core(TM) i7-8550U CPU @1.80GHz(8CPU) |
| 3 | RAM | 8 GB |

* 1. Software

Spesifikasi *software* yang digunakan antara lain :

Tabel 5. 2 Spesifikasi Software

|  |  |  |
| --- | --- | --- |
| **No** | **Software** | **Spesifikasi** |
| 1 | Sistem Operasi | Windows 10 |
| 2 | Development Tools | Google Colaboratory, Jupyter Notebook |
| 3 | Bahasa Pemograman | Python |
| 4 | Web Browser | Google Chrome |
| 5 | Dokumentasi | Ms. Word, Google docs. |

## 5.2 Batasan Implementasi

Batasan implementasi yang ditetapkan dalam implementasi Tugas Akhir ini adalah :

* + - 1. Bahasa pemograman yang digunakan dalam melakukan implementasi arsitektur dan pembentukan model prediksi jumlah pendaftar adalah *python*.
      2. Implementasi yang dilakukan oleh peneliti sampai pada tahap pembuatan aplikasi web untuk melakukan prediksi jumlah pendaftar calon mahasiswa IT Del yang akan mendaftar pada tahun depan.
      3. Implementasi yang dilakukan berfokus pada pembangunan model dengan algoritma *Support Vector Regression* dan *Artificial Neural Network*. Dari kedua algoritma akan dilakukan penentuan algoritma dengan akurasi terbaik yang akan digunakan pada aplikasi web yang disiapkan.
      4. Implementasi dilakukan mulai dari tahapan *encoding data*, dikarenakan tahapan *preprocessing* lain seperti data integrasi dan data cleaning sudah dilakukan secara manual dengan cara yang telah dijelaskan pada[**subbab 3.3.1**](#_3.3.1_Data_Integration) dan [**subbab 3.3.2**](#_3.2.2__Data).

## 5.3 Implementasi Preprocessing

Tahapan implementasi *preprocessing* data yang dilakukan adalah *encoding* data dan normalisasi data dimana data yang digunakan merupakan data yang telah melalui tahapan integrasi data dan *cleaning* data.

### 5.3.1 Pengecekan Outlier

Sebelum dilakukan tahapan *pre-processing* data untuk mencegah adanya terdeteksi nilai yang tidak wajar yang dapat mengakibatkan kesalahan pada analisis data dan prediksi. Outlier terjadi ketika nilai berada di luar range nilai pada umumnya. Berdasarkan implementasi yang telah dilakukan untuk pengecekan outlier, diperoleh hasil sebagai berikut:

|  |
| --- |
| Total outliers pada tahun 2016: 168  Total outliers pada tahun 2017: 172  Total outliers pada tahun 2018: 163  Total outliers pada tahun 2019: 169  Total outliers pada tahun 2020: 171  Total outliers pada tahun 2021: 170  Total outliers pada tahun 2022: 166 |

Berdasarkan hasil outlier yang diterima dari hasil pengecekan, bahwa sebagian besar variabel mengalami oulier. Hal ini disebabkan karena nilai pendaftar pada setiap sekolah berbeda-beda dan tidak adanya kemungkinan jumlah pendaftar antara sekolah A, sekolah B, dan sekolah C berada pada *range* nilai yang sama. Oleh karena, sifat data jumlah pendaftar antara sekolah yang satu dengan yang lain dapat berada pada rentang yang jauh, maka peneliti tidak melakukan tindakan untuk mengatasi *outlier* yang terjadi.

### 5.3.2 Encoding

Sesuai dengan analisis data yang telah dilakukan pada [**subbab 3.3.3**](#_3.3.3_Data_Encoding) bahwa pada dataset yang akan digunakan dalam pemodelan, terdapat variabel ”nama sekolah” yang perlu diubah ke dalam bentuk numerik**.** Hal ini digunakan agar nilai ”nama sekolah” dapat dipelajari, dikarenakan *machine* *learning* hanya dapat mengolah data dengan bentuk numerik. Berikut ini adalah bentuk implementasi encoding data yang dilakukan.

|  |  |  |
| --- | --- | --- |
| Input | : | Dataset file .csv yang telah dilakukan data integrasi dan data cleaning. |
| Output | : | Dataset baru .csv yang telah dilakukan *encoding* data. |
| Library | : | Labelencoder yang digunakan untuk melakukan *encoding* |
| Tahapan | : | 1.Lakukan import dataset dan library yang akan digunakan.  2.Memisahkan variabel yang akan di encoding dengan variabel lain.  3.Lakukan *encoding* pada variabel yang telah dipisahkan |

### 5.3.3 Normalisasi

Seperti yang sudah dijelaskan pada [**subbab 2.12**](#_2.12_Normalisasi) mengenai normalisasi, pada dataset yang akan digunakan pada penelitian ini juga dilakukan tahapan normalisasi untuk menyamakan range nilai pada data. Tahapan normalisasi ini dilakukan untuk mempermudah pemrosesan data dan meningkatkan performa dari model yang dibuat.

|  |  |  |
| --- | --- | --- |
| Input | : | Dataset file .csv yang telah melewati tahapan *encoding* |
| Output | : | Data yang telah dinormalisasikan |
| Library | : | MinMaxScaler digunakan untuk melakukan normalisasi |
| Tahapan | : | 1.Lakukan import dataset dan library yang akan digunakan.  2.Menentukan variabel feature dan target  3. Melakukan normalisasi pada variabel feature |

## Implementasi Algoritma Support Vector Regression

### 5.4.1 Pencarian nilai parameter terbaik

Pada tahapan implementasi algoritma *support vector regression* dilakukan dengan menggunakan data yang normalisasi pada tahapan sebelumnya. Sebelum dilakukan pembentukan model, peneliti melakukan pencarian *parameter* yang akan digunakan dalam pembentukan model. Pencarian parameter dilakukan dengan menggunakan teknik *gridsearch*. Sesuai dengan penjelasan pada[**subbab 2.8**](#_2.8__Grid), penggunaan teknik GridSearch akan menghasilkan parameter terbaik dari nilai-nilai yang telah diinisialisasi. Untuk nilai dari kernel dan parameter-parameter diinisialisasi sebagaimana sudah dijelaskan pada [**subbab 4.4.1**](#_4.4.1_Rancangan_Implementasi) dengan nilai-nilai berikut.

Tabel 5. 3 Nilai parameter algoritma support vector regression

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Kernel | Linear, RBF. Sigmoid, Poly |
| C | 0.1, 1, 10, 100 |
| Epsilon | 0.01, 0.1, 1 |
| Gamma | 0.01, 0.1, 1 |

Dengan menggunakan GridSearch nilai-nilai dari setiap kernel dan parameter yang telah diinisialisasikan akan didapatkan nilai kernel dan parameter terbaik. *Grid Search* akan mencari kombinasi parameter satu per satu dan membandingkan nilai galat terkecil pada parameter dan kernel tersebut (Saputra et al., 2019). Berikut ini adalah tahapan implementasi *gridsearch* yang dilakukan.

|  |  |  |
| --- | --- | --- |
| Input | : | Data yang telah dinormalisasikan |
| Output | : | Nilai Kernel dan parameter terbaik menggunakan *Grid Search* |
| Library | : | * Pandas digunakan untuk memuat dataset dan membuat dataframe. * KFold digunakan untuk mengatur pembagian data * GridSearchCV digunakan untuk menemukan parameter dan kernel terbaik |
| Tahapan | : | 1.Melakukan import dataset dan library yang akan digunakan  2. Menentukan variabel input dan output  3. Melakukan normalisasi pada variabel input  4. Melakukan inisialisasi nilai parameter dan kernel  5. Melakukan pemodelan menggunakan algoritma SVR  6. Melakuka fungsi objek pada *Grid Search*  7. Menampilkan output yaitu kernel dan parameter terbaik pada algoritma SVR berdasarkan nilai yang telah diinisialisasi. |

### 5.4.2 Percobaan variasi nilai k-fold Cross Validation

Setelah mendapatkan nilai kernel dan parameter terbaik menggunakan *Grid Search* proses selanjutnya yang dilakukan yaitu melakukan penyetelan algoritma SVR menggunakan parameter dan kernel terbaik yang telah didapatkan dengan variasi nilai k-fold cross validation yang digunakan. Kemudian melakukan pengecekan nilai RMSE berdasarkan nilai dari k-fold cross validation. Untuk rentang yang diberikan yaitu mulai dari nilai k = 2 sampai dengan nilai k = 10.

Adapun langkah-langkah yang dilakukan adalah sebagai berikut.

|  |  |  |
| --- | --- | --- |
| Input | : | Kernel dan parameter terbaik dan nilai k=2 sampai 10 |
| Output | : | Evaluasi model menggunakan RMSE |
| Library | : | * Pandas digunakan untuk membuat dataset dan membuat dataframe * Stats digunakan untuk membantu menghitung statistik deskriptif. * Numpy digunakan untuk melakukan komputasi ilmiah atau scientific computing * Label Encoder digunakan untuk melakukan encoding terhadap dataset * K-Fold digunakan untuk melakukan pembagian data ke dalam bentuk data latih dan data uji * SVR digunakan untuk implementasi algoritma suppor vector regression. * mean\_squared\_error digunakan untuk menemukan evaluasi model menggunakan RMSE |
| Tahapan | : | 1.Melakukan import dataset dan library yang akan digunakan.  2. Menentukan variabel input dan output  3. Melakukan normalisasi pada variabel input  4. Melakukan inisialisasi nilai kernel dan parameter dan dicari nilai terbaik dari masing – masing kernel dan parameter tersebut.  5. Melakukan inisialisasi nilai k untuk penggunaan K-Fold.  6. Melakukan pemodelan menggunakan algoritma SVR menggunakan nilai kernel dan parameter terbaik  7. Melakukan evaluasi model yang telah dibagun menggunakan RMSE  8. Melakukan prediksi jumlah pendaftar |

Berdasarkan nilai RMSE yang didapatkan pada setiap nilai k, nilai RMSE yang paling kecil yang akan digunakan untuk melakukan pemodelan algoritma dan digunakan untuk melakukan prediksi jumlah pendaftar

## 5.5 Implementasi Algoritma Artificial Neural Network

Pada implementasi algoritma *Artificial Neural Network* data yang digunakan adalah data yang telah menyelesaikan tahapan normalisasi. Sebelum dilakukan implementasi algoritma, peneliti melakukan pencarian *hyparameter* sama seperti yang dilakukan algoritma *support vector regression* yang dijelaskan pada [**subbab 5.4**](#_5.4__Implementasi) yaitu dengan menggunakan teknik *gridsearch*. Agar *gridsearch* dapat mempelajari nilai *hyperparameter* pada model yang akan digunakan, maka perlu dilakukan pembentukan model terlebih dahulu. Pembentukan model harus disesuaikan dengan bentuk model yang akan dilakukan pada saat training data. Setelah model dibentuk maka dilakukan inisialisasi nilai *hyperparameter* seperti pada **Tabel 4.1.**

Setelah dilakukan inisialisasi *hyperparameter* selanjutnya akan dideklarasikan fungsi *gridsearch* untuk menjalankan setiap nilai *hyperparameter* ke dalam model dan untuk menghasilkan *hyperparameter* terbaik. Implementasi teknik ini digambarkan pada langkah berikut :

|  |  |  |
| --- | --- | --- |
| Input | : | Data yang telah dinormalisasikan |
| Output | : | Nilai *hyperparameter* terbaik yang dihasilkan oleh teknik *gridsearch*. |
| Library | : | * Pandas digunakan untuk memuat dataset dan membuat dataframe. * Tensorflow digunakan untuk membantu dalam proses pelatihan *neural network*. * GridSearch digunakan untuk mencari nilai parameter terbaik yang menghasilkan performa model yang optimal. * Keras digunakan untuk membuat bentuk pemodean *deep learning*. |
| Tahapan | : | 1.Lakukan import dataset dan library yang akan digunakan.  2.Menentukan variabel input dan output.  3.Melakukan normalisasi pada variabel input.  4.Buat struktur model ANN yang digunakan.  5.Inisialisasi nilai-nilai *hyperparameter* yang akan digunakan.  6. Inisialisasi objek Gridsearch dengan memasukkan *hyparameter* yang akan dijalankan.  7. Menjalankan fungsi objek gridsearch  8. Menampilkan output *best\_params* sebagai nilai *hyperparameter* yang lebih optimal. |

Nilai-nilai *hyperparameter* yang telah diterima sebagai nilai yang paling optimal, maka selanjutnya nilai-nilai tersebut akan digunakan di dalam model algoritma *Artificial Neural Network* yang akan dilatih. Berikut adalah implementasi dari pemodelan *Artificial Neural Network*.

|  |  |  |
| --- | --- | --- |
| Input | : | Data yang telah dinormalisasikan |
| Output | : | Nilai hasil evaluasi model seperti RMSE dan Akurasi |
| Library | : | * Pandas digunakan untuk memuat dataset dan membuat dataframe. * Tensorflow digunakan untuk membantu dalam proses pelatihan *neural network*. * K-Fold digunakan untuk melakukan pembagian data ke dalam bentuk data latih dan data uji. * Keras digunakan untuk membuat bentuk pemodean *deep learning*. * Numpy digunakan untuk melakukan komputasi ilmiah atau *scientific computing*. * Matplotlib digunakan untuk visualisasi data. |
| Tahapan | : | 1.Lakukan import dataset dan library yang akan digunakan.  2.Menentukan variabel input dan output.  3.Melakukan normalisasi pada variabel input.  4.Menginisialisasikan nilai *hyperparameter* yang akan dicari nilai terbaiknya.  5.Menginisialisasikan nilai *k* untuk penggunaan *K-Fold*.  6.Melakukan train data sebanyak nilai iterasi (*epoch*) kali dengan membagi variabel X dan y ke dalam *X\_train, X\_test, y\_train, y\_test* sebanyak *k* bagian. Pada saat train juga dilakukan pembentukan model ANN dan pemanggilan *hyperparameter* yang akan digunakan. Setelah dilakukan pembentukan model, selanjutnya model yang dijalankan , kemudian model kita latih, dan kita lakukan evaluasi model.  7. Menampilkan RMSE data latih dan data uji  8. Menampilkan nilai RMSE dari hasil pembentukan model  9. Menampilkan visualisasi dari nilai RMSE model. |

Pada pembentukan model algoritma, peneliti membentuk model dengan 1 input layer, 2 *hidden layer*, dan 1 output layer sehingga banyak layer yang dibentuk adalah 4 layer. Setiap layer memiliki jumlah neuron yang berbeda-beda seperti yang dijelaskan pada [**subbab 4.4.2**](#_4.4.2_Rancangan_Implementasi) . Nilai bias dan bobot pada setiap layer akan diinisialisasikan secara otomatis dengan menggunakan fungsi *kernel\_initializer=uniform* seperti yang dapat dilihat pada [**Source Code Pemodelan Algoritma SVR**](#_Source_Code_5.) . Untuk nilai aktivasi yang digunakan pada layer juga berbeda dan disesuaikan dengan fungsi aktivasi yang lebih optimal sesuai dengan hasil pada *gridsearch*. Setelah dilakukan pembentukan model, dilakukan konfigurasi proses pelatihan model. Pelatihan model ini melibatkan *loss function*, optimasi, dan metric evaluasi yang digunakan. *Loss function* ini digunakan untuk melihat perbedaan antara nilai prediksi dan nilai sebenarnya. Optimasi digunakan untuk mengatur mengoptimalkan parameter model selama pelatihan. Pengoptimalan yang dimaksud adalah perbaikan nilai bobot saat melakukan pelatihan dengan menggunakan nilai laju pembelajaran yang digunakan. Metric evaluasi digunakan untuk memantau kinerja model selama pelatihan model. Setelah model dikonfigurasi, model siap untuk dilatih dengan menggunakan perintah *model.fit()*. Dalam proses ini, model akan dilatih sesuai dengan nilai *epoch* yang telah diinisialisasikan. Proses pelatihan akan dilakukan selama nilai iterasi *epoch* belum terpenuhi. Tahapan terakhir dari proses pelatihan adalah evaluasi model. Pada tahapan ini model yang telah dilatih akan dilakukan evaluasi, dimana dari hasil evaluasi akan menghasilkan nilai RMSE ketika proses pelatihan, RMSE ketika proses validasi, dan nilai RMSE dari metric evaluasi model yang dilakukan. Pada pelaksanaan implementasi algoritma, peneliti melakukan percobaan untuk mencari nilai *k* dan nilai *epoch* terbaik. Dalam percobaan ini nilai-nilai *epoch* akan digunakan pada setiap nilai *k*, sehingga dengan itu peneliti dapat melihat bagaimana nilai RMSE pada setiap *epoch* dengan pembagian  *k* data. Selain melihat nilai RMSE, peneliti juga melihat hasil pelatihan apakah hasil pelatihan mengalami *overfitting*, *underfitting*, atau *good fit*. Pada hasil yang diperoleh hanya akan menerima hasil pelatihan yang *good fit* dan memiliki nilai RMSE terendah. Dalam implementasi algoritma ANN ada beberapa percobaan yang dilakukan. Percobaan ini akan dijelaskan pada subbab berikut.

### 5.5.1 Pencarian Nilai Epoch

Untuk melakukan pencarian nilai epoch yang akan digunakan dalam pemodelan, peneliti menggunakan teknik *early stopping*.Percobaan ini dilakukan peneliti untuk melihat nilai epoch yang optimal digunakan pada model yang telah dirancang. Pada percobaan ini pertama kali dilakukan inisialisasi nilai epoch yang dipilih secara acak. Nilai epoch yang digunakan dalam percobaan ini adalah 50, 100, 250, 500, dan 1000. Pada bagian implementasi, teknik *early stopping* digunakan dengan penambahan fungsi *early\_stop* yang diambil dari library *Keras* dan ditambahkan seperti pada potongan kode berikut :

|  |
| --- |
| early\_stop = tf.keras.callbacks.EarlyStopping (monitor='val\_loss', patience=20)  model.fit(X\_train, y\_train, epochs=num\_epochs, batch\_size=batch\_size,validation\_data=(X\_test,y\_test), callbacks=[early\_stop]) |

Seperti yang sudah dijelaskan pada [**subbab 2.15**](#_2.15_Early_Stopping), bahwa proses pelatihan akan berhenti ketika sudah mulai akan terjadi *overfitting*, sehingga dari hasil percobaan ini akan dihasilkan nilai-nilai epoch yang optimal pada nilai *k* dan nilai *epoch* tertentu. Berdasarkan data hasil percobaan pada lampiran [**Hasil Implementasi Early Stopping**](#_B.1._Hasil_Implementasi) yang merupakan hasil implementasi yang dilakukan, nilai *epoch* optimal berada di antara nilai 25 sampai dengan 100. Berlandaskan dari *range* nilai *epoch* tersebut maka peneliti melakukan percobaan kedua dengan membuat epoch baru dalam dalam rentang nilai tersebut dan nilai *epoch*  yang ditentukan memiliki nilai range yang sama. Penentuan nilai *epoch* dengan range yang sama memiliki tujuan untuk melihat perbedaan hasil evaluasi model dengan jarak perbedaan nilai *epoch* yang sama.

### 5.5.2 Percobaan Variasi Nilai Epoch

Seperti yang telah dijelaskan pada [**subbab 5.5.1**](#_5.5.1_Percobaan_Early) di atas, bahwa akan dilakukan percobaan kedua untuk melihat nilai *epoch* terbaik dari antara nilai variasi nilai *epoch* lain yang diambil dari rentang nilai 25 sampai 100. Nilai *epoch* yang diuji adalah 25, 50, 75, 100, 125. Dari nilai-nilai *epoch* tersebut maka akan dicari nilai *epoch* yang menghasilkan nilai RMSE yang terbaik, dan nilai *epoch* yang menghasilkan RMSE terbaik akan digunakan sebagai nilai *epoch* pada model yang akan digunakan di dalam simulator. Variasi nilai *epoch* ini akan dikombinasi dengan nilai *k*.

Pada saat melakukan percobaan variasi *epoch*, terdapat kemungkinan nilai RMSE menurun ketika nilai epoch semakin tinggi. Dikarenakan kondisi tersebut, peneliti menambah kembali nilai *epoch* sampai 300 *epoch*, sehingga nilai *epoch* yang digunakan dalam percobaan ini adalah 25, 50, 75, 100, 125, 150, 175, 200, 225, 250, 275, dan 300.

# BAB 6 HASIL DAN PEMBAHASAN

Pada bab ini akan dilakukan penjelasan mengenai hasil implementasi yang telah dilakukan.

## 6.1. Hasil dan Pembahasan Preprocessing Dataset

Pada sub-bab ini akan menjelaskan hasil pre-processing yang didapatkan. Adapun tahapan pre-processing yang dilakukan yaitu dengan menggunakan Label Encoder dan mendapatkan hasil sebagai berikut:



Gambar 6. 1 Dataset yang sudah dilakukan pre-processing

Berdasarkan hasil pre-processing dataset yang didapatkan yang merujuk pada bagian **6.1,** variabel ”nama sekolah” yang sebelumnya bertipe kategorikal tidak dapat dimasukkan kedalam pemodelan, kemudian melalui hal itu diperlukan tindakan konversi untuk mengubah tipe dari variabel ”nama sekolah” agar berbentuk numerik. Teknik yang dilakukan yaitu Encoding. Adapun encoding yang digunakan yaitu Label Encoder yang mana didapatkan kolom atau variabel baru dengan nama ”nama\_sekolah\_encoded”. Setelah melakukan proses encoding menggunakan Label Encoder, dilakukan normalisasi untuk menghilangkan perbedaan skala antara fitur atau atribut data, dan digunakan normalisasi min-max, dan kemudian hasil nya digunakan untuk melakukan pemodelan.

## 6.2. Hasil dan Pembahasan Algoritma Support Vector Regression

Setelah membangun model prediksi jumlah pendaftar dengan menggunakan algoritma Support Vector Regression, maka didapatkan hasil sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| **kernel** | **gamma** | **cost** | **Epsilon** |
| Linear | 0,001 | 100 | 0,01 |

Setelah proses *pre-processing* dataset dilakukan seperti pada Gambar 6.1, kemudian dilakukan pemodelan menggunakan dataset yang sudah siap pakai, dan hal pertama yang dilakukan yaitu menemukan kernel dan parameter terbaik berdasarkan inisialisai kernel dan parameter yang diberikan pada [**subbab 4.4.1**](#_4.4.1_Rancangan_Implementasi). Setelah melakukan pemilihan kernel dan parameter terbaik menggunakan grid search, grid search akan bekerja dengan cara melakukan kombinasi terhadap nilai kernel dan parameter yang telah diinisialisasikan untuk mendapatkan nilai error yang paling rendah. Melalui cara kerja tersebut, didapatkan hasil bahwa kernel terbaik yaitu kernel linear, hasil ini menunjukkan bahwa dataset yang digunakan dalam penelitian lebih mudah dipisahkan secara linier dengan menggunakan hyperplane. Kemudian setelah mendapatkan nilai dari kernel yang terbaik, dilanjutkan dengan mencari nilai terbaik dari setiap parameter yaitu parameter gamma dengan nilai 0,001, parameter cost dengan nilai 100, parameter epsilon dengan nilai 0,01. Untuk parameter gamma menghasilkan nilai 0,001 menunjukkan bahwa radius kesamaan yang besar yang menghasilkan lebih banyak titik yang dikelompokkan bersama, namun dikarenakan kernel terbaik yang didapatkan adalah kernel liner, maka untuk parameter gamma tidak terlalu berpengaruh terhadap nilai error dan juga hasil prediksi. Kemudian parameter cost sendiri menghasilkan nilai 100 menunjukkan bahwa model memiliki deviasi yang kecil dari paraemeter epsilon yang masih dapat ditoleransi. Dan untuk parameter epsilon menghasilkan nilai terbaik pada nilai 0,01 mengindikasikan bahwa toleransi kesalahan yang lebih rendah ataupun ketat memberikan hasil yang lebih baik dalam hal performa berdasarkan metrik evaluasi yang digunakan.

Setelah mendapatkan nilai terbaik dari setiap kernel dan parameter, setiap nilai tersebut digunakan dalam pemodelan dan dilakukan evaluasi model dengan menggunakan evaluasi model Root Mean Squared Error, dengan menggunakan kombinasi pada rentang nilai k pada k-fold cross validation mulai dari rentang nilai 2 sampai dengan 10, setelah dilakukan analisis, didapatkan kesimpulan bahwa nilai k pada k-fold cross validation berpengaruh pada nilai error model, kemudian didapatkan nilai RMSE terendah pada nilai k=5, dengan nilai 0,00455. Adapun hasil yang didapatkan adalah sebagai berikut:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **k** | **kernel** | **gamma** | **Cost** | **Epsilon** | **RMSE** |
| 2 | Linear | 0,001 | 100 | 0,01 | 0.00839 |
| 3 | Linear | 0,001 | 100 | 0,01 | 0.00561 |
| 4 | Linear | 0,001 | 100 | 0,01 | 0.00580 |
| 5 | Linear | 0,001 | 100 | 0,01 | 0.00455 |
| 6 | Linear | 0,001 | 100 | 0,01 | 0.00507 |
| 7 | Linear | 0,001 | 100 | 0,01 | 0.00535 |
| 8 | Linear | 0,001 | 100 | 0,01 | 0.00480 |
| 9 | Linear | 0,001 | 100 | 0,01 | 0.00478 |
| 10 | Linear | 0,001 | 100 | 0,01 | 0.00457 |

## 6.3. Hasil dan Pembahasan Algoritma Artificial Neural Network

Pada subbab ini akan membahas tentang hasil yang diperoleh setelah melakukan percobaan. Setelah melakukan percobaan mulai dari tahapan pencarian nilai *hyperparameter* dengan menggunakan *gridsearch* hingga melakukan pemodelan algoritma. Berdasarkan percobaan yang dilakukan untuk mencari nilai *hyperparameter* yang optimal digunakan, diperoleh nilai-nilai *hyperparameter* seperti berikut :

Jumlah neuron pada *hidden layer* yang digunakan adalah 12 neuron.

Fungsi aktivasi yang digunakan pada *hidden layer* adalah ReLU.

Nilai *learning rate* yang digunakan adalah 0.01.

Nilai-nilai *hyperparameter* yang telah diperoleh tidak akan mempengaruhi nilai-nilai data yang diproses. Nilai-nilai ini hanya berperan dalam mengatur bagaimana model belajar agar dapat bekerja secara optimal. Penggunaan nilai *hyperparameter* pada model memiliki dampak yang cukup berpengaruh pada kinerja model yang sedang dikembangkan. Untuk menilai kinerja model digunakan metrik RMSE untuk melihat ukuran kinerja model. Pemilihan metrik RMSE digunakan sebagai tolak ukur kinerja model dikarenakan tujuan model yang dibuat adalah melakukan prediksi nilai-nilai di masa depan (*continues target*). Metrik ini hanya akan mengukur besar kesalahan prediksi yang mungkin terjadi pada model. Semakin kecil nilai RMSE yang diperoleh maka akan semakin kecil kemungkinan kesalahan yang dilakukan oleh model, sehingga dapat dikatakan bahwa model tersebut akan semakin baik.

Setelah memperoleh nilai *hyperparameter* seperti di atas, nilai-nilai tersebut selanjutnya digunakan ke dalam model algoritma untuk dilakukan pembentukan model. Seperti yang telah disampaikan pada [**subbab 5.5**](#_5.5__Implementasi) bahwa pada saat melakukan pemodelan dilakukan percobaan juga terkait nilai *k* dan nilai *epoch* yang menghasilkan nilai RMSE terbaik. Dari hasil percobaan yang telah dilakukan, diperoleh nilai *k*  dan nilai *epoch* terbaik adalah *k* = 5 dan *epoch* = 300 dengan nilai RMSE yang dihasilkan adalah 0.000528855104879. Berdasarkan hasil percobaan yang dilakukan bahwa kombinasi nilai ­*k* dan nilai *epoch* juga dapat mempengaruhi kinerja model yang dibangun. Oleh karena itu, selain nilai *learning rate*, fungsi aktivasi, dan jumlah neuron, pemilihan nilai *k* dan nilai *epoch* juga perlu diperhatikan untuk menghasilkan kinerja model yang lebih baik.

## 6.4. Pembahasan Evaluasi Model

Berdasarkan hasil percobaan dan evaluasi model yang telah diperoleh dari kedua algoritma maka didapatkan bahwa bentuk pemodelan terbaik dihasilkan pada algoritma *artificial neural network*  dengan nilai RMSE yang diperoleh adalah 0,000528855104879. Berdasarkan hasil evaluasi model ini, terbukti hipotesis **H0** yang disampaikan pada [**Subbab 3.7**](#_3.7_Hipotesis_Penelitian) bahwa kinerja model ANN lebih baik dibandingkan dengan model SVR dalam melakukan prediksi.

## 6.5. Hasil dan Pembahasan Prototype

Hasil dari pembangunan aplikasi yang telah dibangun sesuai dengan design prototype yang telah dibuat, aplikasi sudah dapat berjalan dengan baik dan sudah dapat menghasilkan hasil prediksi.

# BAB 7 KESIMPULAN DAN SARAN

## 7.1. Kesimpulan

Berdasarkan hasil penelitian dan implementasi dapat diberi kesimpulan sebagai berikut:

1. Untuk melakukan pengolahan data mentah ke dalam bentuk dataset yang akan diolah ke dalam model, dimulai dari bentuk dataset yang ingin diolah baik dalam bentuk dataset time series ataupun dataset tabular. Selain itu juga perlu diperhatikan tahapan *pre-processing* yang perlu dilakukan untuk menyiapkan dataset.
2. Untuk melakukan pemodelan algoritma *Artificial Neural Network* dan *Support Vector Regression* yang optimal, perlu dilakukan pencarian nilai *hyperparameter* yang dapat membantu kinerja model agar bekerja dengan baik. Dengan menghasilkan bentuk pemodelan yang optimal, maka kemungkinan kesalahan prediksi akan semakin kecil.
3. Berdasarkan hasil percobaan yang telah dilakukan pada kedua model algoritma yaitu algoritma *Artificial Neural Network* dan *Support Vector Regression*, model yang memiliki nilai RMSE terbaik dengan nilai terendah model terbaik ditunjukkan pada model algoritma *Artificial Neural Network* dengan nilai RMSE = 0,000528855104879 .

## 7.2. Saran

Berdasarkan hasil penelitian yang telah dilakukan ada beberapa saran yang perlu dilakukan pada penelitian selanjutnya:

* + - 1. Pada penelitian selanjutnya dapat digunakan dengan bentuk dataset yang lain dengan mempertimbangkan variabel-variabel yang dapat mempengaruhi jumlah pendaftar calon mahasiswa baru Institut Teknologi Del.

# LAMPIRAN

## Lampiran A Source Code

### Source Code 1. Encoding Data

|  |
| --- |
| #import library yang akan digunakan  import pandas as pd  import numpy as np  from sklearn.preprocessing import LabelEncoder  import matplotlib.pyplot as plt  # Load dataset  data = pd.read\_excel('data\_breakdown.xlsx')  # Membuat objek label encoder  encoder = LabelEncoder()  # Melakukan encoding pada variabel 'School Name'  data['nama\_sekolah\_encoded'] = encoder.fit\_transform(data['Nama Sekolah'])  #Mengurutkan posisi kolom agar lebih mudah digunakan  coloum\_order=['Nama Sekolah', 'nama\_sekolah\_encoded',2016,2017,2018,2019,2020,2021,2022]  #import hasil dataset yang telah di encoding  data\_prep.to\_csv('data-prep.csv', index=0) |

### Source Code 2. *Grid Search* ANN

|  |
| --- |
| #import library  import numpy as np  import pandas as pd  import tensorflow as tf  from keras.layers import Dense  from keras.models import Sequential  from sklearn.preprocessing import StandardScaler  from keras.wrappers.scikit\_learn import KerasRegressor  from sklearn.model\_selection import KFold  from sklearn.model\_selection import GridSearchCV  from keras.wrappers.scikit\_learn import KerasClassifier  # Memuat dataset yang telah dilakukan preprocessing encoding  from google.colab import drive  drive.mount('/content/gdrive')  %cd /content/gdrive/MyDrive/ANN/Code/Data  data = pd.read\_csv('data-prep.csv')  # Penentuan Variabel Input(X) dan Variabel Output(y)  X = data.iloc[:, 1:].values  y = data.iloc[:, -1].values.reshape(-1, 1)  # Menormalisasikan variabel input dengan menggunakan normalisasi min-max  scaler = StandardScaler()  X= scaler.fit\_transform(X)  # Fungsi untuk membuat model  def create\_model(num\_neurons\_hidden\_layer1, num\_neurons\_hidden\_layer2, activation1, activation2, learning\_rate):  model = Sequential()  model.add(Dense(X.shape[1],kernel\_initializer='uniform', input\_dim=X.shape[1]))  model.add(Dense(num\_neurons\_hidden\_layer1,kernel\_initializer='uniform', activation=activation1))  model.add(Dense(num\_neurons\_hidden\_layer2,kernel\_initializer='uniform', activation=activation2))  model.add(Dense(1, activation='linear'))  model.compile(loss='mean\_squared\_error', optimizer=tf.keras.optimizers.Adam(learning\_rate), metrics=['accuracy'])    return model  # Membungkus objek model ke dalam KerasClassifier  model = KerasClassifier(build\_fn=create\_model)  # Definisikan *hyperparameter* yang akan dioptimasi  param\_grid = {'num\_neurons\_hidden\_layer1': [12,16,24, 64, 128],  'num\_neurons\_hidden\_layer2': [12,16,24, 32, 64,128],  'batch\_size':[2,16,32,42],  'activation1':['relu', 'sigmoid','linear'],  'activation2':['relu','sigmoid','linear'],  'learning\_rate': [0.01, 0.001]}  # Inisialisasi objek GridSearchCV  grid\_search = GridSearchCV(estimator=model, param\_grid=param\_grid, cv=2)  # Fit the *Grid Search* object to the data  grid\_search.fit(X, y)  # Print the best parameters and the corresponding MSE score  print("Best Parameters: ", grid\_search.best\_params\_) |

### Source Code 3. *Grid Search* SVR

|  |
| --- |
| from google.colab import drive  # Mengimport modul GD pada Gcolab ke /content/drive  drive.mount('/content/drive')  import pandas as pd  import numpy as np  from sklearn.preprocessing import LabelEncoder,MinMaxScaler  from sklearn.svm import SVR  from sklearn.model\_selection import train\_test\_split, GridSearchCV, KFold  from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score  import matplotlib.pyplot as plt  # Melakukan Load data  data = pd.read\_csv("/content/drive/My Drive/Tugas\_Akhir/data-prep.csv")  # Penentuan Variabel Input(X) dan Variabel Output(y)  X = data.iloc[:, 1:].values  y = data.iloc[:, -1].values.reshape(-1, 1)  # Menormalisasikan variabel input dengan menggunakan normalisasi min-max  scaler = StandardScaler()  X= scaler.fit\_transform(X)  # Inisialisasi Parameter  parameters = {  "kernel" <- linear, rbf, sigmoid, polynomial  "C" <- 0.1, 1, 10, 100  "epsilon" <- 0.01, 0.1, 1  "gamma" <- 0.01, 0.1, 1  }  # Melakukan pemodelan menggunakan SVR  model = SVR()  cv = KFold(n\_splits= 2 to 10 , shuffle=True, random\_state=42)  grid\_search = GridSearchCV(model, parameters, cv=cv, scoring="neg\_root\_mean\_squared\_error")  grid\_search.fit(X, y)  print("Parameter terbaik pada model SVR:", grid\_search.best\_params\_) |

### Source Code 4. Pemodelan Algoritma ANN

|  |
| --- |
| #import library  from keras.models import model\_from\_json  import numpy as np  import pandas as pd  import tensorflow as tf  from keras.layers import Dense  import matplotlib.pyplot as plt  from keras.models import Sequential  from sklearn.preprocessing import MinMaxScaler  from keras.wrappers.scikit\_learn import KerasRegressor  from sklearn.model\_selection import KFold  # Memuat dataset yang telah dilakukan preprocessing encoding  from google.colab import drive  drive.mount('/content/gdrive')  %cd /content/gdrive/MyDrive/ANN/Code/Data  data = pd.read\_csv('data-prep.csv')  # Penentuan Variabel Input(X) dan Variabel Output(y)  X = data.iloc[:, 1:].values  y = data.iloc[:, -1].values.reshape(-1, 1)  # Menormalisasikan variabel input dengan menggunakan normalisasi min-max  scaler = MinMaxScaler()  X= scaler.fit\_transform(X)  **PEMODELAN**  # Define the *hyperparameter*s  learning\_rate = 0.01  num\_epochs = 250  batch\_size = 2  num\_neurons\_hidden\_layer\_1 = 32  num\_neurons\_hidden\_layer\_2 = 24  num\_neurons\_output = 1  # Define k-fold cross validation  n\_folds = 8  kf = KFold(n\_splits=n\_folds, shuffle=True, random\_state=16)  #Membuat variabel untuk menyimpan score MSE dan acc  rmse\_scores = []  acc\_scores = []  #TRAIN  for train\_idx, test\_idx in kf.split(X, y):  # Split the data into training set and testing set  X\_train, X\_test = X[train\_idx], X[test\_idx]  y\_train, y\_test = y[train\_idx], y[test\_idx]  # Membangun bentuk arsitektur ANN  model = Sequential()  model.add(Dense(X.shape[1],kernel\_initializer='uniform', input\_dim=X.shape[1]))  model.add(Dense(num\_neurons\_hidden\_layer1,kernel\_initializer='uniform', activation='linear'))  model.add(Dense(num\_neurons\_hidden\_layer2,kernel\_initializer='uniform', activation='sigmoid'))  model.add(Dense(num\_neurons\_output , activation='linear'))  # Menjalankan model yang telah dibentuk  model.compile(loss='mean\_squared\_error', optimizer=tf.keras.optimizers.Adam(learning\_rate), metrics=['acc'])  # Train the model  history = model.fit(X\_train, y\_train, epochs=num\_epochs, batch\_size=batch\_size, validation\_data=(X\_test,y\_test))  # Evaluate the model on the training set  train\_loss, train\_acc = model.evaluate(X\_train, y\_train, verbose=0)  # Evaluate the model on the test set  test\_loss, test\_acc = model.evaluate(X\_test, y\_test, verbose=0)    # Menghitung nilai RMSE dan akurasi pada data testing setiap fold  rmse\_scores.append(np.sqrt(test\_loss))  acc\_scores.append(test\_acc)    print("\n")  #Menampilkan akurasi data test dan data training  print("Training Accuracy: {:.2f}%".format(train\_acc \* 100))  print("Test Accuracy: {:.2f}%".format(test\_acc \* 100))  #Print rata-rata MSE dan akurasi dari k-fold cross-validation  print(f'Average RMSE score: {np.mean(rmse\_scores):.4f}')  print(f'Average acc score: {np.mean(acc\_scores)\*100:.2f} %')  #Menampilkan plot untuk melihat grafik akurasi dan loss  # Plot accuracy  plt.plot(history.history['acc'])  plt.plot(history.history['val\_acc'])  plt.title('Model Accuracy')  plt.ylabel('Accuracy')  plt.xlabel('Epoch')  plt.legend(['train', 'test'], loc='upper left')  plt.show()  # Plot loss  plt.plot(history.history['loss'])  plt.plot(history.history['val\_loss'])  plt.title('Model Loss')  plt.ylabel('Loss')  plt.xlabel('Epoch')  plt.legend(['train', 'test'], loc='upper left')  plt.show()  #Melakukan prediksi dengan menggunakan data X\_test  forecasting = model.predict(X\_test)  print(forecasting)  # Evaluasi Hasil Prediksi dengan RMSE  from sklearn.metrics import mean\_squared\_error  from math import sqrt  mse = mean\_squared\_error(forecasting, y\_test)  rmse = sqrt(mse)  print('RMSE: %f'% rmse) |

### Source Code 5. Pemodelan Algoritma SVR

|  |
| --- |
| library(pandas)  library(numpy)  library(matplotlib)  # Load dataset  Data = rad\_csv(”/content/drive/My Drive/Tugas\_Akhir/data-final-ta.csv”)  # Checking outliers  data\_check=data.drop(columns='Nama Sekolah')  mean = np.mean(data\_check)  std\_dev = np.std(data\_check)  print('Mean :', mean)  print('Standar deviasi : ', std\_dev)  # count z-score  z\_score = (data\_check - mean) / std\_dev  print('Nilai z-score : ', z\_score)  # identify outliers depends z-score value  outliers <- data\_check[(z\_score < batas\_bawah) | (z\_score > batas\_atas)]  print("Jumlah outliers: ", len(outliers))  # Normalization  scaler = MinMaxScaler()  X= scaler.fit\_transform(X)  # svr implementation  # initialization parameter  Kernel <- linear, polynomial, sigmoid, poly  C <- 0.1, 1, 10, 100  Epsilon <- 0.01, 0.1, 1  Gamma <- 0.01, 0.1, 1  # K-Fold Implementation  cv <- KFold(n\_splits= 2 to 10, shuffle=True, random\_state=42)  # *Grid Search* Implementation  # Implementasi GridSearch  grid\_search <- GridSearchCV(model, parameters, cv=cv, scoring="neg\_root\_mean\_squared\_error")  grid\_search.fit(X, y)  print("Best score:", -grid\_search.best\_score\_)  print("Parameter terbaik pada model SVR:", grid\_search.best\_params\_)  # model evaluation  rmse\_scores = []  for train\_index, test\_index in cv.split(X, y):  X\_train, X\_test = X[train\_index], X[test\_index]  y\_train, y\_test = y[train\_index], y[test\_index]  # Melakukan training model pada setiap fold  best\_model.fit(X\_train, y\_train)  # Melakukan prediksi pada data testing  y\_pred = best\_model.predict(X\_test)  # Menghitung RMSE  rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))  rmse\_scores.append(rmse)  # Menghitung rata-rata nilai RMSE  avg\_rmse = np.mean(rmse\_scores)  print("RMSE:", avg\_rmse)  # predicting total registrans |

## 

## Lampiran B Hasil Implementasi

### B.1. Hasil Implementasi Early Stopping

|  |  |  |  |
| --- | --- | --- | --- |
| **K** | **Epoch** | **RMSE** | **Grafik Loss** |
| 2 | 50 | 1.7287 |  |
|  | 100 | 2.1539 |  |
|  | 250 | 1.9023 |  |
|  | 500 | 1.7718 |  |
|  | 1000 | 1.4075 |  |
| 3 | 50 | 1.6082 |  |
|  | 100 | 2.5202 |  |
|  | 250 | 1.4228 |  |
|  | 500 | 1.2878 |  |
|  | 1000 | 1.4240 |  |
| 4 | 50 | 1.5200 |  |
|  | 100 | 1.2978 |  |
|  | 250 | 0.5457 |  |
|  | 500 | 1.0361 |  |
|  | 1000 | 1.4044 |  |
| 5 | 50 | 1.7034 |  |
|  | 100 | 1.0921 |  |
|  | 250 | 1.3027 |  |
|  | 500 | 1.8213 |  |
|  | 1000 | 1.0513 |  |
| 6 | 50 | 1.3160 |  |
|  | 100 | 1.4356 |  |
|  | 250 | 1.6127 |  |
|  | 500 | 1.4892 |  |
|  | 1000 | 1.1542 |  |
| 7 | 50 | 0.9618 |  |
|  | 100 | 0.8367 |  |
|  | 250 | 1.0022 |  |
|  | 500 | 1.3330 |  |
|  | 1000 | 0.8742 |  |
| 8 | 50 | 1.1765 |  |
|  | 100 | 0.8401 |  |
|  | 250 | 0.8828 |  |
|  | 500 | 1.0371 |  |
|  | 1000 | 1.2442 |  |
| 9 | 50 | 1.1127 |  |
|  | 100 | 0.6499 |  |
|  | 250 | 0.6821 |  |
|  | 500 | 1.1198 |  |
|  | 1000 | 1.2872 |  |
| 10 | 50 | 1.2626 |  |
|  | 100 | 1.1056 |  |
|  | 250 | 0.8009 |  |
|  | 500 | 1.0929 |  |
|  | 1000 | 1.2441 |  |

### B.2. Hasil Implementasi Artificial Neural Network

|  |  |  |  |
| --- | --- | --- | --- |
| **K** | **Epoch** | **RMSE** | **Grafik Loss** |
| 2 | 25 | 3.438939862412025 |  |
|  | 50 | 2.7875840508632947 |  |
|  | 75 | 2.4038626077529193 |  |
|  | 100 | 1.1869217909250027 |  |
|  | 125 | 1.0331602513146225 |  |
|  | 150 | 1.1377781555163482 |  |
|  | 175 | 0.25589813206110995 |  |
|  | 200 | 0.1886967953753623 |  |
|  | 225 | 0.03111090274051176 |  |
|  | 250 | 0.045496688367639725 |  |
|  | 275 | 0.1085616003228827 |  |
|  | 300 | 0.013954165317098628 |  |
| 3 | 25 | 3.193403363245757 |  |
|  | 50 | 1.7118233262635967 |  |
|  | 75 | 1.0494064961980243 |  |
|  | 100 | 0.5599950451307182 |  |
|  | 125 | 0.1253463991525267 |  |
|  | 150 | 0.06279257127432451 |  |
|  | 175 | 0.05141818888072767 |  |
|  | 200 | 0.013826568297501002 |  |
|  | 225 | 0.01184645778536645 |  |
|  | 250 | 0.15693683173040993 |  |
|  | 275 | 0.16653101481879015 |  |
|  | 300 | 0.16458265098389085 |  |
| 4 | 25 | 2.685105147293489 |  |
|  | 50 | 1.852713374453132 |  |
|  | 75 | 0.6480644469280247 |  |
|  | 100 | 0.28097460568256766 |  |
|  | 125 | 0.20697836182190454 |  |
|  | 150 | 0.09386506459686453 |  |
|  | 175 | 0.026401936020311368 |  |
|  | 200 | 0.019344500647970435 |  |
|  | 225 | 0.004786774357240431 |  |
|  | 250 | 0.008128318027280049 |  |
|  | 275 | 0.005664892640118424 |  |
|  | 300 | 0.0019863496762552403 |  |
| 5 | 25 | 2.669052191467496 |  |
|  | 50 | 1.637964475135055 |  |
|  | 75 | 0.3737569216197687 |  |
|  | 100 | 0.26794388034247774 |  |
|  | 125 | 0.2245488528542809 |  |
|  | 150 | 0.2758472155179349 |  |
|  | 175 | 0.15730655644502475 |  |
|  | 200 | 0.008489350195782348 |  |
|  | 225 | 0.03856728363401609 |  |
|  | 250 | 0.0031995331039631402 |  |
|  | 275 | 0.04705094715382409 |  |
|  | 300 | 0.0005288551048793737 |  |
| 6 | 25 | 2.5831638839628495 |  |
|  | 50 | 1.3119160736166355 |  |
|  | 75 | 0.42193106606642045 |  |
|  | 100 | 0.19686680238591817 |  |
|  | 125 | 0.04751489562484102 |  |
|  | 150 | 0.026375547006644285 |  |
|  | 175 | 0.023040734270126317 |  |
|  | 200 | 0.01002567742955106 |  |
|  | 225 | 0.003075995549444306 |  |
|  | 250 | 0.0018702185577078297 |  |
|  | 275 | 0.0007863470756262829 |  |
|  | 300 | 0.001026175674282013 |  |
| 7 | 25 | 2.4304509152552605 |  |
|  | 50 | 1.1888226453662412 |  |
|  | 75 | 0.34764898653149817 |  |
|  | 100 | 0.12345263917058868 |  |
|  | 125 | 0.035926542089134705 |  |
|  | 150 | 0.031202343141708753 |  |
|  | 175 | 0.00958314378582385 |  |
|  | 200 | 0.002689401346946824 |  |
|  | 225 | 0.0014933016950617625 |  |
|  | 250 | 0.0020926379458555116 |  |
|  | 275 | 0.0006432096139618035 |  |
|  | 300 | 0.0009482358529844376 |  |
| 8 | 25 | 2.3984831769205415 |  |
|  | 50 | 1.1133163921923497 |  |
|  | 75 | 0.42656457507250034 |  |
|  | 100 | 0.13923480159094404 |  |
|  | 125 | 0.056923458319583786 |  |
|  | 150 | 0.01719775109948827 |  |
|  | 175 | 0.008425436282929612 |  |
|  | 200 | 0.002894983383658573 |  |
|  | 225 | 0.0021489424496988156 |  |
|  | 250 | 0.0011222786413965726 |  |
|  | 275 | 0.002417377452086527 |  |
|  | 300 | 0.0009735199407929115 |  |
| 9 | 25 | 2.488338691319856 |  |
|  | 50 | 0.9594285521097008 |  |
|  | 75 | 0.34357327013424827 |  |
|  | 100 | 0.08292645987458835 |  |
|  | 125 | 0.04339699332557359 |  |
|  | 150 | 0.026140180961522527 |  |
|  | 175 | 0.009253973333603095 |  |
|  | 200 | 0.0024792691871124255 |  |
|  | 225 | 0.011958175391564829 |  |
|  | 250 | 0.0005823397206967445 |  |
|  | 275 | 0.0012123310788996933 |  |
|  | 300 | 0.005005012948727575 |  |
| 10 | 25 | 2.5778362169643976 |  |
|  | 50 | 1.46112172057286 |  |
|  | 75 | 0.5454319593502512 |  |
|  | 100 | 0.1998487278026349 |  |
|  | 125 | 0.08955237729489159 |  |
|  | 150 | 0.04601806153578511 |  |
|  | 175 | 0.01667731119193255 |  |
|  | 200 | 0.006720867116259743 |  |
|  | 225 | 0.0023062993750202405 |  |
|  | 250 | 0.018958301209324926 |  |
|  | 275 | 0.0017862247656611097 |  |
|  | 300 | 0.0023661251087691982 |  |

### B.3. Hasil Implementasi Support Vector Regression



## Lampiran C Aplikasi Web Sistem Prediksi Del Predict

# DAFTAR REFERENSI

Abdel-Sattar, M., Aboukarima, A. M., & Alnahdi, B. M. (2021). Application of artificial neural network and support vector regression in predicting mass of ber fruits (Ziziphus mauritiana Lamk.) based on fruit axial dimensions. *PLoS ONE*, *16*(1 January), 1–15. https://doi.org/10.1371/journal.pone.0245228

Andreas, F., Mikhael, & Enri, U. (2022). *Perbandingan Algoritma Backpropagation Neural Network dan Long Short-Term Memory dalam Prediksi Harga Bitcoin*. *8*(12), 547–558.

Andriyani, W. (2021). *Apa perbedaan Root Mean Square Error (RMSE) dan Standard Deviation? Bagaimana cara penggunaannya?* Quora. https://id.quora.com/Apa-perbedaan-Root-Mean-Square-Error-RMSE-dan-Standard-Deviation-Bagaimana-cara-penggunaannya#:~:text=RMSE digunakan untuk mengukur tingkat,nilai Profit berdasarkan nilai Sales.

Barus, T. E., & Simamora, N. S. P. (2020). Analisis Persebaran Pendaftar di Institut Teknologi Del Berdasarkan Asal Wilayah. *Prosiding Industrial Research …*, 26–27. https://jurnal.polban.ac.id/proceeding/article/view/2193

*Early Stopping*. (2020). DeepLearning4J.

Gunawan, M. I., Sugiarto, D., & Mardianto, I. (2020). Peningkatan Kinerja Akurasi Prediksi Penyakit Diabetes Mellitus Menggunakan Metode Grid Seacrh pada Algoritma Logistic Regression. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *6*(3), 280. https://doi.org/10.26418/jp.v6i3.40718

Ishukatiyar. (2023). *Backpropagation in Data Mining*. GeeksForGeeks. https://www.geeksforgeeks.org/backpropagation-in-data-mining

Itdel. (2022). *Pendaftaran Mahasiswa Baru IT Del*. http://spmb.del.ac.id/

Jabbar, H. W. (2022). *Penerapan Metode Backpropagation Pada Data Klasifikasi Kualitas Air*. RPubs by RStudio. https://rpubs.com/Husein\_Wisnu/PenerapanBackpropagation

Khoirudin, K., Nurdiyah, D., & Wakhidah, N. (2019). Prediksi Penerimaan Mahasiswa Baru Dengan Multi Layer Perceptron. *Jurnal Pengembangan Rekayasa Dan Teknologi*, *14*(1), 1. https://doi.org/10.26623/jprt.v14i1.1212

Lembong, R. I. (2022). MACHINE LEARNING MENGGUNAKAN PERBANDINGAN DUA ALGORITMA ANTARA SUPPORT VECTOR REGRESSION DAN DECISION TREE UNTUK MEMPREDIKSI HARGA SAHAM AGRO. In *upj.ac.id* (Vol. 33, Issue 1).

Mahyunis, R. V. (2022). Penerapan Artificial Neural Network (ANN) Menggunakan Algoritma Backpropagation dengan Membandingkan Empat Fungsi Aktivitas dalam Memprediksi Harga Emas. *Braz Dent J.*, *33*(1), 1–12.

Maryana, T., Kusrini, K., & Fatta, H. Al. (2019). Analisis Perbandingan Predisksi Obat Dengan Menggunakan Metode Abc Analisys Dan Svr Pada Aplikasi “Morbis.” *Jurnal Teknologi Informasi*, *3*(2), 174. https://doi.org/10.36294/jurti.v3i2.1016

Muhadzdzab, H., Asfi, M., & Putri, T. E. (2020). Sistem Prediksi untuk Menentukan Jumlah Pendaftaran Mahasiswa Baru pada Unversitas Catur Insan Cendekia Menggunakan Metode Least Square. *Jurnal Informatika Universitas Pamulang*, *5*(3), 350. https://doi.org/10.32493/informatika.v5i3.6598

Mustafeez, A. Z. (2023). *What is Early Stopping*. Educative.Io.

Nafi’iyah, N. (2016). Perbandingan Regresi Linear , Backpropagation Dan Fuzzy Mamdani Dalam Prediksi Harga Emas. *Seminar Nasional Inovasi Dan Aplikasi Teknologi Di Industri*, 291–296.

Nainggolan, K. S., & Lumbanraja, B. (2018). *PREDIKSI CURAH HUJAN DI DELI SERDANG DENGAN BACKPROPAGATION NEURAL NETWORK , DEEP BELIEF NETWORK DAN SUPPORT VECTOR REGRESSION TUGAS AKHIR Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik 11S14008 KWATRI SABATTYAN NAINGGOLAN 11S140*.

Nasution, D. A., Khotimah, H. H., & Chamidah, N. (2019). Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma K-NN. *Computer Engineering, Science and System Journal*, *4*(1), 78. https://doi.org/10.24114/cess.v4i1.11458

Primartha, R. (2018). *Belajar Machine Learning Teori dan Praktik*.

Putra, H., & Ulfa Walmi, N. (2020). Penerapan Prediksi Produksi Padi Menggunakan Artificial Neural Network Algoritma Backpropagation. *Jurnal Nasional Teknologi Dan Sistem Informasi*, *6*(2), 100–107. https://doi.org/10.25077/teknosi.v6i2.2020.100-107

Rais, Z. (2022). Analisis Support Vector Regression (Svr) Dengan Kernel Radial Basis Function (Rbf) Untuk Memprediksi Laju Inflasi Di Indonesia. *VARIANSI: Journal of Statistics and Its Application on Teaching and Research*, *4*(1), 30–38. https://doi.org/10.35580/variansiunm13

Ridwan, D. I., Setianingsih, C., & Murti, M. A. (2021). *Prediksi Penggunaan Energi Listrik Support Vector Regression Prediction of Electricity Using Support Vector Regression Method*. *8*(6), 12135–12144.

Rizal, N. A. (2014). *Implementasi Metode Hybrid Jst-Som Pada Prediksi Churn Pelanggan Seluler : Studi Kasus Pt . Telekomunikasi Seluler Implementation Hybrid Nn-Som Method for Prediction Customer Churn : Study Case Pt . Telekomunikasi Seluler*.

Rusdi, Z., Lubis, C., & Tjandra, V. G. (2021). *Prediksi Kurs Mata Uang dengan Metode Long Short Term Memory Berbasis Attention*.

Saadah, S., Z, F. Z., & Z, H. H. (2021). Support Vector Regression (SVR) Dalam Memprediksi Harga Minyak Kelapa Sawit di Indonesia dan Nilai Tukar Mata Uang EUR/USD. *Journal of Computer Science and Informatics Engineering (J-Cosine)*, *5*(1), 85–92. https://doi.org/10.29303/jcosine.v5i1.403

Santoso, H., & Putri, R. A. (2023). *Deteksi Komentar Cyberbullying pada Media Sosial Instagram Menggunakan Algoritma Random Forest Cyberbullying Comment Detection on Instagram Social Media Using Random Forest Algorithm*. *13*(April), 62–72.

Saputra, G. H., Wigena, A. H., & Sartono, B. (2019). Penggunaan Support Vector Regression Dalam Pemodelan Indeks Saham Syariah Indonesia Dengan Algoritme *Grid Search*. *Indonesian Journal of Statistics and Its Applications*, *3*(2), 148–160. https://doi.org/10.29244/ijsa.v3i2.172

Sepri, D., & Fauzi, A. (2020). Prediksi Harga Cabai Merah Menggunakan Support Vector Regression. *Computer Based Information System Journal*, *8*(2), 1–5. https://doi.org/10.33884/cbis.v8i2.1921

Setyoningrum, N. R., Rahimma, P. J., Teknologi, S. T., Tanjungpinang, I., & Tanjungpinang, K. (2022). Implementasi Algoritma Regresi Linear Dalam Sistem Prediksi Pendaftar Mahasiswa Baru Sekolah Tinggi Teknologi Indonesia Tanjungpinang. *Prosiding Seminar Nasional Ilmu Sosial Dan Teknologi (SNISTEK)*, *4*, 13–18. https://ejournal.upbatam.ac.id/index.php/prosiding/article/view/5200

Situmorang, Y. S. (2022). Analisis Pengaruh Holt Winter Exponential Smoothing sebagai Metode Optimasi Pemulusan Data Pada Model Peramalan Backpropagation Neural Network. *IT Del*, *1*.

Somya, R. (2018). Perancangan Aplikasi Chatting Berbasis Web di PT. Pura Barutama Kudus menggunakan Socket.IO dan Framework Foundation. *Khazanah Informatika : Jurnal Ilmu Komputer Dan Informatika*, *4*(1), 8–15. https://doi.org/10.23917/khif.v4i1.5979

Sugara, B., & Subekti, A. (2019). Penerapan Support Vector Machine (Svm) Pada Small Dataset Untuk Deteksi Dini Gangguan Autisme. *Jurnal Pilar Nusa Mandiri*, *15*(2), 177–182. https://doi.org/10.33480/pilar.v15i2.649

Syahputra, M. E., Bahri, S., & Rambe, M. F. (2020). Pengaruh Kepemimpinan, Disiplin dan Motivasi Terhadap Kinerja Pegawai Dinas Tarukim Labura. *Pamator Journal*, *13*(1), 110–117. https://doi.org/10.21107/pamator.v13i1.7017

Trivusi. (2022). *Normalisasi*. Trivusi. https://www.trivusi.web.id/2022/09/normalisasi-data.html

Yuniarti, D., & Gunawan, B. (2021). *PENERAPAN DATA MINING UNTUK MEMPREDIKSI JUMLAH SISWA DIKTUKPA (PENDIDIKAN PEMBENTUKAN PERWIRA) TNI AL DENGAN METODE ARTIFICIAL NEURAL NETWORK DAN ALGORITMA BACKPROPAGATION*. *5*(4), 938–954. https://doi.org/10.52362/jisamar.v5i4.574

Abdel-Sattar, M., Aboukarima, A. M., & Alnahdi, B. M. (2021). Application of artificial neural network and support vector regression in predicting mass of ber fruits (Ziziphus mauritiana Lamk.) based on fruit axial dimensions. *PLoS ONE*, *16*(1 January), 1–15. https://doi.org/10.1371/journal.pone.0245228

Andreas, F., Mikhael, & Enri, U. (2022). *Perbandingan Algoritma Backpropagation Neural Network dan Long Short-Term Memory dalam Prediksi Harga Bitcoin*. *8*(12), 547–558.

Andriyani, W. (2021). *Apa perbedaan Root Mean Square Error (RMSE) dan Standard Deviation? Bagaimana cara penggunaannya?* Quora. https://id.quora.com/Apa-perbedaan-Root-Mean-Square-Error-RMSE-dan-Standard-Deviation-Bagaimana-cara-penggunaannya#:~:text=RMSE digunakan untuk mengukur tingkat,nilai Profit berdasarkan nilai Sales.

Barus, T. E., & Simamora, N. S. P. (2020). Analisis Persebaran Pendaftar di Institut Teknologi Del Berdasarkan Asal Wilayah. *Prosiding Industrial Research …*, 26–27. https://jurnal.polban.ac.id/proceeding/article/view/2193

*Early Stopping*. (2020). DeepLearning4J.

Gunawan, M. I., Sugiarto, D., & Mardianto, I. (2020). Peningkatan Kinerja Akurasi Prediksi Penyakit Diabetes Mellitus Menggunakan Metode Grid Seacrh pada Algoritma Logistic Regression. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *6*(3), 280. https://doi.org/10.26418/jp.v6i3.40718

Ishukatiyar. (2023). *Backpropagation in Data Mining*. GeeksForGeeks. https://www.geeksforgeeks.org/backpropagation-in-data-mining

Itdel. (2022). *Pendaftaran Mahasiswa Baru IT Del*. http://spmb.del.ac.id/

Jabbar, H. W. (2022). *Penerapan Metode Backpropagation Pada Data Klasifikasi Kualitas Air*. RPubs by RStudio. https://rpubs.com/Husein\_Wisnu/PenerapanBackpropagation

Khoirudin, K., Nurdiyah, D., & Wakhidah, N. (2019). Prediksi Penerimaan Mahasiswa Baru Dengan Multi Layer Perceptron. *Jurnal Pengembangan Rekayasa Dan Teknologi*, *14*(1), 1. https://doi.org/10.26623/jprt.v14i1.1212

Lembong, R. I. (2022). MACHINE LEARNING MENGGUNAKAN PERBANDINGAN DUA ALGORITMA ANTARA SUPPORT VECTOR REGRESSION DAN DECISION TREE UNTUK MEMPREDIKSI HARGA SAHAM AGRO. In *upj.ac.id* (Vol. 33, Issue 1).

Mahyunis, R. V. (2022). Penerapan Artificial Neural Network (ANN) Menggunakan Algoritma Backpropagation dengan Membandingkan Empat Fungsi Aktivitas dalam Memprediksi Harga Emas. *Braz Dent J.*, *33*(1), 1–12.

Maryana, T., Kusrini, K., & Fatta, H. Al. (2019). Analisis Perbandingan Predisksi Obat Dengan Menggunakan Metode Abc Analisys Dan Svr Pada Aplikasi “Morbis.” *Jurnal Teknologi Informasi*, *3*(2), 174. https://doi.org/10.36294/jurti.v3i2.1016

Muhadzdzab, H., Asfi, M., & Putri, T. E. (2020). Sistem Prediksi untuk Menentukan Jumlah Pendaftaran Mahasiswa Baru pada Unversitas Catur Insan Cendekia Menggunakan Metode Least Square. *Jurnal Informatika Universitas Pamulang*, *5*(3), 350. https://doi.org/10.32493/informatika.v5i3.6598

Muhathir, M., Santoso, M. H., & Larasati, D. A. (2021). Wayang Image Classification Using SVM Method and GLCM Feature Extraction. *Journal of Informatics and Telecommunication Engineering*, *4*(2), 373–382. https://doi.org/10.31289/jite.v4i2.4524

Mustafeez, A. Z. (2023). *What is Early Stopping*. Educative.Io.

Primartha, R. (2018). *Belajar Machine Learning Teori dan Praktik*.

Putra, H., & Ulfa Walmi, N. (2020). Penerapan Prediksi Produksi Padi Menggunakan Artificial Neural Network Algoritma Backpropagation. *Jurnal Nasional Teknologi Dan Sistem Informasi*, *6*(2), 100–107. https://doi.org/10.25077/teknosi.v6i2.2020.100-107

Rais, Z. (2022). Analisis Support Vector Regression (Svr) Dengan Kernel Radial Basis Function (Rbf) Untuk Memprediksi Laju Inflasi Di Indonesia. *VARIANSI: Journal of Statistics and Its Application on Teaching and Research*, *4*(1), 30–38. https://doi.org/10.35580/variansiunm13

Ridwan, D. I., Setianingsih, C., & Murti, M. A. (2021). *Prediksi Penggunaan Energi Listrik Support Vector Regression Prediction of Electricity Using Support Vector Regression Method*. *8*(6), 12135–12144.

Rizal, N. A. (2014). *Implementasi Metode Hybrid Jst-Som Pada Prediksi Churn Pelanggan Seluler : Studi Kasus Pt . Telekomunikasi Seluler Implementation Hybrid Nn-Som Method for Prediction Customer Churn : Study Case Pt . Telekomunikasi Seluler*.

Rusdi, Z., Lubis, C., & Tjandra, V. G. (2021). *Prediksi Kurs Mata Uang dengan Metode Long Short Term Memory Berbasis Attention*.

Saadah, S., Z, F. Z., & Z, H. H. (2021). Support Vector Regression (SVR) Dalam Memprediksi Harga Minyak Kelapa Sawit di Indonesia dan Nilai Tukar Mata Uang EUR/USD. *Journal of Computer Science and Informatics Engineering (J-Cosine)*, *5*(1), 85–92. https://doi.org/10.29303/jcosine.v5i1.403

Samsudiney. (2019). *Penjelasan Sederhana tentang Apa Itu SVM?* Medium. https://medium.com/@samsudiney/penjelasan-sederhana-tentang-apa-itu-svm-149fec72bd02

Santoso, H., & Putri, R. A. (2023). *Deteksi Komentar Cyberbullying pada Media Sosial Instagram Menggunakan Algoritma Random Forest Cyberbullying Comment Detection on Instagram Social Media Using Random Forest Algorithm*. *13*(April), 62–72.

Trivusi. (2022). *Normalisasi*. Trivusi. https://www.trivusi.web.id/2022/09/normalisasi-data.html

Handayani, A., Jamal, A., & Septiandri, A. A. (2017). *Evaluasi Tiga Jenis Algoritme Berbasis Pembelajaran Mesin untuk Klasifikasi Jenis Tumor Payudara*. *6*(4), 394–403.