BAB II

TINJAUAN PUSTAKA

2.1 Stroke

Menurut WHO stroke adalah adanya tanda-tanda klinik yang berkembang cepat akibat gangguan fungsi otak fokal (atau global) dengan gejala-gejala yang berlangsung selama 24 jam atau lebih yang menyebabkan kematian tanpa adanya penyebab lain yang jelas selain vaskuler (Susilo, 2000). Stroke dibedakan menjadi dua, yaitu:

a. Stroke *Ischemic*

   Stroke *ischemic* terjadi karena adanya sumbatan pembuluh darah oleh *thromboembolic* yang mengakibatkan daerah di bawah sumbatan tersebut mengalami *ischemic* (Atmaja, 2016).

b. Stroke Hemoragik

   Stroke hemoragik yang terjadi akibat adanya *mycroaneurisme* yang pecah (Atmaja, 2016). Sepertiga penyakit stroke disebabkan oleh stroke hemoragik.

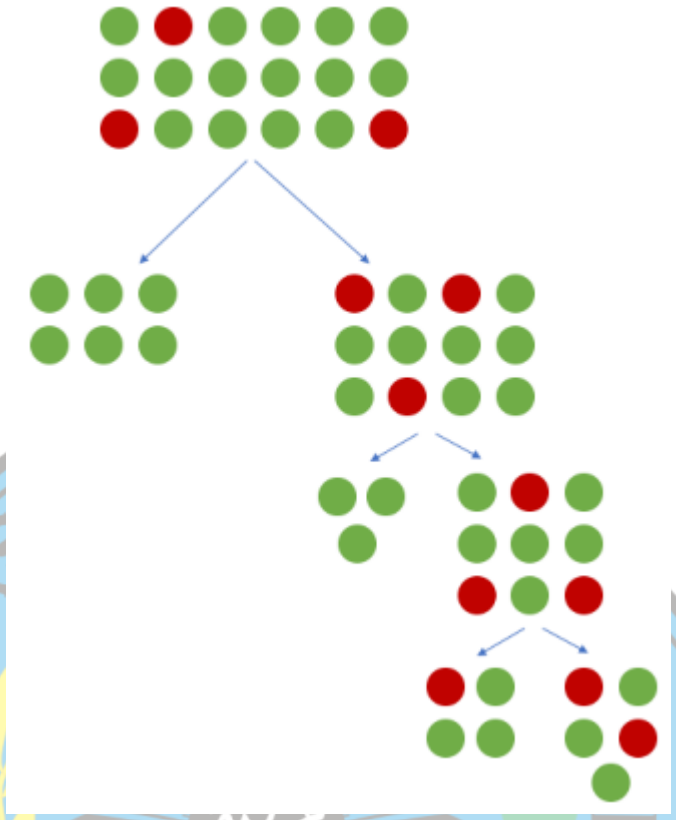
Faktor risiko stroke adalah kondisi atau penyakit atau kelainan yang terdapat pada seseorang yang memiliki potensi untuk memudahkan orang tersebut mengalami serangan stroke pada suatu saat (Atmaja, 2016). Faktor risiko penyakit stroke antara lain hipertensi, penyakit kardiovaskuler, kolesterol tinggi, obesitas, peningkatan hematokrit, diabetes, kontrasepsi oral, merokok, penyalahgunaan obat serta konsumsi alkohol.

2.2 *Preprocessing* Data

Preprocessing data bertujuan untuk mentransformasikan data input mentah ke dalam format yang sesuai untuk analisi selanjutnya (Pangastuti, 2017). Salah satu masalah umum yang diselesaikan pada tahap *preprocessing* data adalah menangani *missing value. Missing value* terjadi ketika terdapat beberapa informasi yang tidak tersedia untuk sebuah subyek (kasus) yang biasanya disebabkan karena kesalahan input data, informasi tentang subyek tidak diberikan atau tidak tersedia. *Missing value* bisa diatasi dengan mengisi *missing* data. Jika persentase data *missing value* melebihi 30%, maka data boleh dihapus sedangkan jika persentase data *missing value* kurang 30%, maka data *missing* diimputasi dengan nilai mean jika data kuantitatif dan modus jika data kualitatif (Hair, 1995).

2.3 Data Tidak Seimbang

Klasifikasi data tidak seimbang merupakan masalah dalam klasifikasi yang terjadi ketika distribusi kelas pada data training tidak seimbang (Brownlee, 2020). Data tidak seimbang terjadi ketika terdapat satu kelas yang memiliki jumlah respon yang kecil (minoritas) dibandingkan dengan kelas lainnya (mayoritas). Klasifikasi pada data kelas tidak seimbang menjadi masalah utama pada bidang machine learning dan data mining, misalnya pada masalah medis, klasifikasi tesks dan sosial media (Siringoringo, 2018). Ketidakseimbangan data ini dapat mengakibatkan kesalahan klasifikasi dalam memprediksi kelas minoritas. Ketidakseimbangan data dapat diatasi dengan menyeimbangkan data menggunakan metode *oversampling* dan *undersampling* (Mardiansyah, 2019).



Gambar 2. 1 Data Tidak Seimbang

Metode *undersampling* dilakukan pada kelas mayoritas dengan memilih secara acak kelas mayoritas agar jumlah kelas mayoritas sama dengan jumlah kelas minoritas. Kekurangan metode ini yaitu menghilangnya informasi yang terkait dengan penghapusan sampel dari data *training* (Seiffert, Khoshgoftaar, Hulse, & Napolitano, 2008). Sedangkan metode *oversampling* dilakukan pada data kelas minoritas dengan menduplikasi data pada kelas minoritas agar jumlah kelas minoritas sama dengan jumlah kelas mayoritas. Duplikasi data ini mengakibatkan model *overfitting* dan meningkatkan waktu pelatihan model (Seiffert, Khoshgoftaar, Hulse, & Napolitano, 2008).

2.4 *Syntethic Minority Oversampling Technique* (SMOTE)

SMOTE merupakan salah satu metode popular dalam mengatasi data tidak seimbang. Cara kerja SMOTE yaitu menambah jumlah sampel pada kelas minor agar setara dengan kelas mayor dengan cara membangkitkan data sintetik berdasarkan tetangga terdekat k-nearest neighbour dimana tetangga terdekat dipilih berdasarkan jarak Euclidean antara kedua data (Chawla, Lazarevic, Hall, & Bowyer, 2003).

Rumus jarak Euclidean secara umum sebagai berikut:

$$d(x,z) = \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2 + \cdots + (x_n - z_n)^2} \quad (2.1)$$

Persamaan pembangkit data sintetik sebagai berikut:

$$x_{syn} = x_i + (x_{knn} - x_i)\gamma \quad (2.2)$$

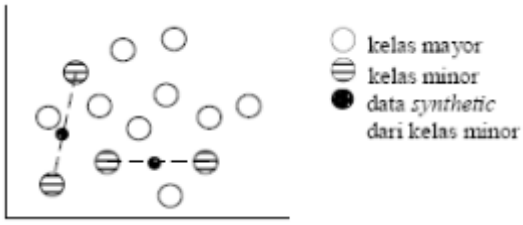
Dimana:

x_{syn} adalah data syntethic

x_i adalah data ke-i dari kelas minor

x_{knn} adalah data dari kelas minor yang memiliki jarak terdekat dari x_i

γ adalah bilangan random antara 0 dan 1



Gambar 2. 2 Ilustrasi Algoritma SMOTE

2.5 *eXtreme Gradient Boosting* (XGBoost)

XGBoost pertama kali diusulkan oleh Chen dan Guestrin tahun 2016 untuk mengatasi berbagai masalah pembelajaran yang bisa menghasilkan hasil yang efisien, cepat dan terukur dengan mengimplementasikan *Gradient Tree Boosting* (Shafila, 2020). *Gradien boosting* menggunakan ensamble dari decision tree untuk memprediksi nilai. *Gradient boosting* membuat simpel decision tree yang berkesinambungan dimana setiap tree baru yang dibuat merupakan perbaikan dari *error tree* sebelumnya. Metode XGBoost menggunakan data *training* dengan beberapa fitur X_i untuk memprediksi variabel target y_i. Nilai prediksi dapat memiliki interpretasi yang berbeda, tergantung pada tugasnya, yaitu regresi atau klasifikasi.

Pelatihan model bertujuan untuk menemukan parameter terbaik θ yang sesuai dengan data *training* X_i dan label Y_i. Prediksi dilakukan dengan menjumlahkan seluruh bobot yang ada di setiap pohon dan kemudian memasukkan nilai tersebut ke fungsi logistik. XGBoost akan meminimalkan fungsi objektif sebagai berikut:

$$L^{(t)} = \sum_{i=1}^{n} t(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (2.3)$$

Dimana L adalah fungsi training loss(kesalahan kuadrat rata-rata) yang mengukur seberapa prediktif model terkait dengan data *training* dan Ω adalah

regularization untuk mengontrol kompleksitas model agar menghindari *overfitting*. Nilai *gain* digunakan untuk splitting nodedihitung menggunakan rumus:

$$L_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i+\lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_L} h_R+\lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i+\lambda}\right] - \gamma \quad (2.4)$$

Sedangkan nilai *similarity* dihitung menggunakan rumus berikut:

$$Similarity = \frac{(\sum \hat{Y}_i)^2}{\sum[previous\ f_i(x).(1-previous f_i(x)]+\lambda} \quad (2.5)$$

Pada pohon klasifikasi biner nilai output pada leaf dapat dikonversi ke dalam bentuk nilai probabilitas dengan menggunakan fungsi sigmoid. Fungsi sigmoid adalah suatu fungsi matematika yang memiliki nilai antara 0 dan 1. Fungsi sigmoid disebut juga sebagai kurva sigmoidal atau fungsi logistik. Adapun rumus fungsi sigmoid adalah sebagai berikut.

$$\sigma(x) = \frac{1}{1+\exp(-x)} \quad (2.6)$$

2.6 Hyperparameter Tuning

Model machine learning memiliki nilai parameter yang perlu diatur untuk mendapatkan model dengan performa yang baik. Hyperparameter tuning digunakan untuk menentukan kombinasi nilai parameter yang tepat untuk memaksimalkan kinerja pemodelan. Pencarian hyperparameter dapat dilakukan secara manual atau otomatis. Pencarian hyperparameter secara manual dilakukan dengan cara mengatur dan menguji beberapa set hyperparameter secara manual. Sedangkan pencarian hyperparameter secara otomatis dilakukan dengan menggunakan

algoritma yang dapat mengotomatiskan dan mengoptimalkan proses untuk menemukan hyperparameter yang optimal.

Pada kasus data penelitian tidak seimbang menggunakan algoritma XGBoost digunakan parameter *scale_pos_weight* (Brownlee, 2020). Nilai parameter *scale_pos_weight* digunakan untuk memperbaiki kesalahan yang dihasilkan model selama pelatihan pada data kelas positif. Sehingga model memiliki kinerja yang baik dalam memprediksi kelas positif. Nilai parameter *scale_pos_weight* dihitung dari jumlah total kelas mayoritas dibagi dengan jumlah total kelas minoritas pada data penelitian.

Salah satu metode hyperparameter tuning yang dapat digunakan adalah grid search. Grid search merupakan metode alternatif yang digunakan untuk mencari parameter terbaik dalam model dengan cara mencoba satu per satu kombinasi parameter dan memvalidasi setiap kombinasinya. Grid search dapat diterapkan secara maksimum apabila batas atas dan batas bawah dari masin-masing parameter diketahui (Ramadhan, M, I.S, & Ghifari, 2017).

Parameter yang digunakan dalam proses hyperparameter tuning adalah sebagai berikut:

Tabel 2. 1 Parameter Hyperparameter Tuning

Parameter	Keterangan
learning_rate (eta)	Penyusutan ukuran yang digunakan untuk mencegah *overfitting*
max_depth	Kedalaman maksimum pada *tree*
min_child_weight	Jumlah bobot minimum pada *child node*

2.7 Kriteria Evaluasi Kinerja

Pengukuran kinerja algoritma klasifikasi dilakukan dengan cara membandingkan antara hasil prediksi algoritma klasifikasi dengan nilai target variabel data *testing* sebagai data sebenarnya (Wang, 2014). Hasil kinerja algoritma klasifikasi dapat diukur menggunakan confusion matrix.

2.7.1 Confusion Matrix

Confusion matrix mengandung informasi tentang kelas data yang aktual direpresentasikan pada baris matriks dan kelas data hasil prediksi pada kolom (Pangastuti, 2017).

Tabel 2. 2 *Confusion Matrix*

	Predictive Positive Class	Predictive Negative Class
Real Positif Class	True Positive (TP)	False Negative (FN)
Real Negatif Class	Flase Positive (FP)	True Negative (TN)

a) True Positive menunjukkan bahwa kelas hasil prediksi klasifikasi adalah positif dan kelas sebenarnya adalah positif.
b) True Negative menunjukkan bahwa kelas hasil prediksi klasifikasi adalah negatif dan kelas sebenarnya adalah negatif.
c) False Positif munjukkan bahwa kelas hasil prediksi klasifikasi adalah negatif dan kelas sebenarnya adalah positif
d) False Negative menunjukkan bahwa kelas hasil prediksi klasifikasi adalah positif dan kelas sebenarnya adalah negatif.

Berdasarkan nilai pada *confusion matrix* diperoleh nilai metrik yang digunakan untuk menghitung kinerja dari model klasifikasi, sebagai berikut:

a. Akurasi

Akurasi adalah tingkat akurasi prediksi keseluruhan dengan membandingkan data aktualnya.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (2.7)$$

b. Presisi

Presisi digunakan untuk mengevaluasi kepastian model dalam memprediksi suatu kelas dengan benar.

$$Presisi = \frac{TP}{TP+FP} \quad (2.8)$$

c. Recall/sensitivity

Recall/sensitivity digunakan untuk mengevaluasi seberapa besar coverage model dalam memprediksi kelas positif diklasifikasikan.

$$Recall/sensitivity = \frac{TP}{TP+FN} \quad (2.9)$$

d. Specificity

Specificity digunakan untuk mengevaluasi seberapa besar coverage model dalam memprediksi kelas negatif diklasifikasikan.

$$Specificity = \frac{TN}{TN+FP} \quad (2.10)$$

e. F1 Measure

Nilai F1 Measure menggabungkan nilai presisi dan recall untuk mengukur kebaikan model pada kelas

$$F1\ Measure = \frac{2(Recall \times Presisi)}{Recall + Presisi} \quad (2.11)$$

2.7.2 Area Under Curve

Area Under Curve (AUC) merupakan ukuran tunggal kinerja klasifikasi untuk evaluasi model mana yang lebih baik secarata rata-rata (Pangastuti, 2017). Nilai AUC diperoleh dengan menghitung nilai *true positive rate* (TPR) yaitu jumlah objek pada kelas positif yang diklasifikasikan dengan benar dan *false positive rate* (FPR) yaitu jumlah objek pada kelas positif yang salah diklasifikasikan.

$$TPR = \frac{TP}{TP + FN} \quad (2.12)$$

$$FPR = 1 - \frac{TN}{TN + FP} \quad (2.13)$$

$$AUC = \frac{1 + TPR - FPR}{2} \quad (2.14)$$

Interpretasi nilai AUC ditunjukkan oleh tabel 2.3 (Shafila, 2020).

Tabel 2. 3 Klasifikasi Nilai AUC

Nilai AUC	Keterangan
0,91 – 1,00	Klasifikasi sangat baik
0,81 – 0,90	Klasifikasi baik
0,71 – 0,80	Klasifikasi cukup
0,61 – 0,70	Klasifikasi buruk
≤ 0,60	Klasifikasi salah