BAB 2
LANDASAN TEORI

Berikut merupakan beberapa teori serta penelitian terdahulu yang digunakan sebagai dasar dalam penelitian untuk melakukan implementasi algoritma *Decision Tree* dalam mendeteksi penyakit strok pada pria.

2.1 Penelitian Terdahulu

Penelitian terdahulu yang digunakan sebagai dasar dalam penelitian untuk mengimplementasikan algoritma *Decision Tree* dalam mendeteksi penyakit strok pada pria dapat dilihat pada Tabel 2.1.

Tabel 2.1. Penelitian Terdahulu

Judul	Nama Penulis	Algoritma	Hasil Penelitian
Heart Disease Prediction Using Decision Tree and SVM	R. Vijaya Saraswathi, K. Gajavelly, A. Kousar Nikath, R. Vasavi, R. Reddy Anumasula	*Gaussian Naïve Bayes*, *Decision Tree*, *Support Vector Machine*, *Random Forest*	Akurasi algoritma *Decision Tree* adalah 89.6% dan lebih tinggi dibandingkan algoritma *Random Forest*, *Support Vector Machine*, dan *Gaussian Naïve Bayes*.
Classification of stroke patients using data mining with AdaBoost, Decision Tree, and Random Forest models	B. Imran, E. Wahyudi, A. Subki, Salman, A. Yani	*AdaBoost*, *Decision Tree*, *Random Forest*	Akurasi algoritma *Decision Tree* sebesar 95.3% dan lebih tinggi dibandingkan dengan algoritma *AdaBoost* dan algoritma *Random Forest*.
			Lanjut pada halaman berikutnya

Tabel 2.1 Penelitian Terdahulu (lanjutan)

Judul	Nama Penulis	Algoritma	Hasil Penelitian
Klasifikasi Penyakit *Stroke* dengan Metode *Support Vector Machine (SVM)*	S. Rahayu, Y. Yamasari	*Support Vector Machine*	Algoritma *Support Vector Machine* dengan *kernel polynomial* memiliki tingkat akurasi paling tinggi, yaitu sebesar 78.86% dengan rasio 80 : 20 dibandingkan *kernel linear*, *RBF*, dan *sigmoid*
Analysis of Stroke Classification using Random Forest Method	M. Firdaus Banjar, Irawati, F. Umar, L. Nur Hayati	*Random Forest*	Algoritma *Random Forest* dengan 100 *trees* memiliki akurasi tertinggi sebesar 86.82% dengan nilai presisi sebesar 15.76%, nilai *recall* sebesar 38.15%, dan *F1-Score* sebesar 22.30%.
Implementasi Algoritma *Logistic Regression* Untuk Klasifikasi Penyakit *Stroke*	Suhliyyah, H. Hikmayanti Handayani, K. Ahmad Baihaqi	*Logistic Regression*	Akurasi algoritma *Logistic Regression* sebesar 94% dengan menggunakan 3938 data *training* dan 996 data *testing*.

Pada penelitian pertama dengan judul "*Heart Disease Prediction Using Decision Tree and SVM*" oleh R. Vijaya Saraswati, K. Gajavelly, A. Kousar Nikath, R. Vasavi, dan R. Reddy Anumasula mengenai prediksi penyakit jantung dengan menggunakan *dataset* yang berasal dari UCI serta menggunakan algoritma *Decision Tree*, algoritma *Gaussian Naïve Bayes*, algoritma *Support Vector Machine*, dan algoritma *Random Forest*, didapatkan bahwa algoritma *Decision Tree* memiliki

tingkat akurasi paling tinggi, yaitu sebesar 89.6%. Sedangkan tingkat akurasi untuk algoritma *Gaussian Naïve Bayes* sebesar 85.6%, algoritma *Support Vector Machine* sebesar 87.6%, dan algoritma *Random Forest* sebesar 88.1%.

Pada penelitian kedua dengan judul "*Classification of stroke patients using data mining with AdaBoost, Decision Tree, and Random Forest models*" oleh B. Imran, E. Wahyudi, A. Subki, Salma, dan A. Yani mengenai klasifikasi penyakit strok dengan menggunakan *dataset* yang berasal dari *Kaggle* dengan pembagian *dataset* menjadi 80% data *training* dan 20% data *testing* serta menggunakan algoritma *AdaBoost*, *Decision Tree*, dan algoritma *Random Forest*, didapatkan bahwa algoritma *Decision Tree* memiliki tingkat akurasi paling tinggi, yaitu sebesar 95.3% dengan menggunakan *Folds* 5 dan *Folds* 10. Kemudian, algoritma *AdaBoost* dengan *Folds* 5 memiliki tingkat akurasi sebesar 91.7%, sedangkan dengan *Folds* 10 memiliki tingkat akurasi sebesar 91.5%. Lalu, algoritma *Random Forest* dengan *Folds* 5 memiliki tingkat akurasi sebesar 95%, sedangkan dengan *Folds* 10 memiliki tingkat akurasi sebesar 94.8%.

Pada penelitian ketiga dengan judul "Klasifikasi Penyakit *Stroke* dengan Metode *Support Vector Machine (SVM)*" oleh S. Rahayu dan Y. Yamasari mengenai klasifikasi penyakit strok dengan menggunakan *dataset Stroke Prediction Dataset* yang berasal dari *Kaggle* serta menggunakan algoritma *Support Vector Machine* dengan *kernel linear*, *kernel RBF*, *kernel polynomial*, dan *kernel sigmoid*, didapatkan bahwa pada *kernel linear* dengan rasio *dataset* 80:20, menghasil tingkat akurasi paling tinggi dibanding rasio lainnya, yaitu sebesar 75.73%, nilai *precision* sebesar 74.07%, dan nilai *recall* sebesar 56.16%. Pada algoritma *Support Vector Machine* menggunakan *kernel RBF* dengan rasio *dataset* 90:10 menghasilkan tingkat akurasi paling tinggi dibanding rasio lainnya, yaitu sebesar 73.38%, nilai *precision* sebesar 71.79%, serta nilai *recall* sebesar 55.85%. Kemudian, pada algoritma *Support Vector Machine* dengan *kernel polynomial* dengan rasio *dataset* 80:20 menghasilkan tingkat akurasi paling tinggi dibanding rasio lain, yaitu sebesar 78.86%, nilai *precision* sebesar 73.98%, dan nilai *recall* sebesar 56.75%. Pada *kernel* terakhir, yaitu *kernel sigmoid* dengan rasio 10:90 menghasilkan tingkat akurasi paling tinggi dibanding rasio lain, yaitu sebesar 46.14%, nilai *precision* sebesar 39.84%, dan 48.13%. Sehingga, pada penelitian ini *kernel polynomial* memiliki tingkat akurasi paling tinggi dibandingkan dengan seluruh *kernel* yang telah diuji.

Pada penelitian keempat dengan judul "*Analysis of Stroke Classification using Random Forest Method*" oleh M. Firdaus Banjar, Irawati, F. Umar, dan

L. Nur Hayati mengenai klasifikasi penyakit strok dengan *dataset* yang berasal dari *Kaggle* serta menggunakan algoritma *Random Forest*, didapatkan bahwa dengan menggunakan 100 *trees*, algoritma *Random Forest* dapat menghasilkan nilai akurasi paling tinggi dari jumlah *trees* lainnya, yaitu sebesar 86.62%, nilai *precision* sebesar 15.76%, nilai *recall* sebesar 38.15%, serta *f1-score* sebesar 22.30%. Sedangkan dengan menggunakan 50 *trees*, tingkat akurasi yang dihasilkan sebesar 86.49%, nilai *precision* sebesar 14.59%, nilai *recall* sebesar 35.52%, dan *f1-score* sebesar 20.68%. Kemudian, dengan menggunakan 200 *trees*, tingkat akurasi yang dihasilkan sebesar 86.30%, nilai *precision* sebesar 13.58%, nilai *recall* sebesar 32.89%, dan *f1-score* sebesar 19.23%. Terakhir, dengan menggunakan 500 *trees*, tingkat akurasi yang dihasilkan sebesar 86.49%, nilai *precision* sebesar 14.97%, nilai *recall* sebesar 36.84%, dan *f1-score* sebesar 21.29%.

Pada penelitian kelima dengan judul ”Implementasi Algoritma *Logistic Regression* Untuk Klasifikasi Penyakit *Stroke*” oleh Suhliyyah, H. Hikmayanti Handayani, dan K. Ahmad Baihaqi mengenai klasifikasi penyakit strok dengan *dataset Brain Stroke Dataset* dengan pembagian 3984 data *training* dan 996 data *testing* serta menggunakan algoritma *Logistic Regression*, didapatkan bahwa algoritma *Logistic Regression* menghasilkan akurasi sebesar 94%, nilai *precision* sebesar 94%, nilai *recall* sebesar 100%, dan *f1-score* sebesar 97%. Penelitian ini juga memperbaiki hasil akurasi dari algoritma *Support Vector Machine* yang memiliki tingkat akurasi sebesar 76%.

2.2 Penyakit Strok

Strok atau *Cerebrovascular disease* merupakan sebuah penyakit yang dikarenakan aliran darah dan oksigen yang terhambat, sehingga terjadi gangguan atau kerusakan pada otak. Menurut *World Stroke Organization* atau WSO, terdapat tiga jenis penyakit strok, yaitu *ischemia stroke* atau strok iskemik, *intracerebral haemorrhage*, dan *subarachnoid haemorrhage* atau strok hemoragik [10]. Strok iskemik terjadi karena aliran darah ke otak terhambat oleh gumpalan darah, sedangkan strok hemoragik terjadi karena pembuluh darah pada otak pecah [11].

Menurut Stephen dan David, sekitar 85% kasus strok adalah strok iskemik yang berasal dari 25% penyakit pada pembuluh darah kecil serebal, 25% kardioemboli, 20% penyakit pada arteri besar, dan sisanya dengan penyakit yang tidak diketahui. Sedangkan sekitar 15% kasus strok adalah strok hemoragik yang disebabkan oleh penyakit pembuluh darah kecil serebral dan angiopati amiloid

serebral [12].

Terdapat kelemahan anggota tubuh, kehilangan sensasi di wajah, bibir tidak simetris, kesulitan berbicara, dan penurunan kesadaran adalah beberapa tanda strok. Kemudian ada dua jenis faktor yang dapat diubah dan tidak dapat diubah dalam penyebab penyakit strok. Faktor yang dapat diubah termasuk obesitas, hipertensi, dan diabetes. Faktor yang tidak dapat diubah termasuk usia, jenis kelamin, genetik, ras, dan etnis [11].

2.3 Decision Tree

Decision Tree atau yang dikenal sebagai pohon keputusan merupakan sebuah metode klasifikasi yang *supervised* dan berbentuk pohon. *Decision tree* memiliki 4 bagian, yaitu *root node* atau simpul akar, cabang, *internal node* atau simpul internal, dan *leaf node* atau simpul daun [13].

Pada *decision tree*, simpul akar ditentukan dengan cara menghitung nilai gain dari setiap atribut. Gain merupakan sebuah ukuran seberapa efektif sebuah atribut dalam mengurangi ketidakpastian data setelah dibagi berdasarkan atribut tersebut [14]. Semakin tinggi nilai gain, maka atribut tersebut semakin baik dalam melakukan pemisahan data. Atribut yang memiliki nilai gain tertinggi akan menjadi akar pertama dari pohon keputusan tersebut [15]. Namun, untuk menghitung nilai gain dari atribut, diperlukan nilai entropi. Entropi merupakan teknik untuk mengukur ketidakpastian kelas dalam suatu *dataset* [14].

Algoritma *decision tree* dapat digunakan untuk melakukan prediksi, manipulasi data, hingga menangani data yang hilang dengan menjadikannya sebagai kategori yang terpisah [16]. *Decision tree* juga dapat meningkatkan kualitas keputusan dibandingkan dengan metode konvensional karena cukup fleksibel dalam memilih fitur dari berbagai *internal node*. [4].

2.4 SMOTEENN

SMOTEENN atau *Synthetic Minority Oversampling Techinque Edited Nearest Neighbors* merupakan sebuah metode *oversampling* dan *undersampling* yang merupakan kombinasi dari *SMOTE* dan *ENN* [17]. *SMOTEENN* dirancang untuk mengatasi ketidakseimbangan kelas dalam sebuah dataset dengan cara menghasilkan sampel sintetis dan menghilangkan data yang bersifat *noise*.

Menurut Bounab et al., pada metode *SMOTEENN* terdapat dua langkah

proses. Langkah pertama, metode *SMOTE* digunakan untuk memperbanyak representasi minoritas sehingga menghasilkan *instance* yang baru dengan interpolasi linier antara sampel kelas minoritas yang sudah ada dengan tetangga terdekatnya. Kemudian pada langkah kedua, *ENN* diterapkan untuk memurnikan *dataset* dengan menghilangkan *noise* atau redundansi dari *instance* yang telah dibuat oleh *SMOTE* dengan cara apabila terdapat kelas dari sampel berbeda dengan kelas mayoritas, maka sampel tersebut akan dihapus dari *dataset* [18].

2.5 Grid Search

Grid Search merupakan salah satu metode pencarian secara menyeluruh yang digunakan untuk menemukan kombinasi optimal dari *hyperparameter* suatu model [19]. Proses pada *Grid Search* melibatkan pengujian model pada setiap kombinasi dari *hyperparameter* dalam sebuah ruang diskrit. Dengan demikian, setiap kombinasi akan diuji, diukur, dan dievaluasi secara satu per satu untuk melakukan identifikasi set *parameter* yang menghasilkan performa terbaik.

Meskipun *Grid Search* mampu untuk melakukan identifikasi kombinasi optimal, namun *Grid Search* memiliki biaya komputasi yang tinggi karena setiap set *hyperparameter* harus dilakukan evaluasi sehingga kompleksitas meningkat seiring luasnya rentang nilai yang diuji [20].

2.6 Pengujian Model

Pengujian model merupakan sebuah teknik untuk melakukan evaluasi dan melakukan validasi kinerja dari sebuah model agar dapat memastikan keakuratan dari sebuah model. Pada umumnya, pengujian model dapat dilakukan dengan membagi sampel menjadi 3 bagian, yaitu data *training*, data *validation*, dan data *testing*.

Data *training* digunakan untuk melakukan pelatihan model [21]. Dalam hal ini, model akan menggunakan data untuk mengidentifikasi dan mempelajari mengenai pola-pola dan hubungan antara berbagai variabel *input* dengan variabel target atau variabel *output* untuk membuat prediksi ataupun keputusan serta model juga akan mempelajari mengenai aturan-aturan pemisahan data yang optimal antar kelas data sehingga dapat membuat prediksi yang akurat dan keputusan yang relevan.

Kemudian, data *validation* digunakan untuk melakukan evaluasi terhadap

model selama proses *training* secara *real-time*. Dalam hal ini, data *validation* berfungsi untuk membantu dalam menentukan *hyperparameter* yang optimal serta dapat mencegah *overfitting*.

Sedangkan data *testing* merupakan tolak ukur akhir untuk melakukan evaluasi kinerja dari keseluruhan model [22] setelah dilakukan tahap pelatihan dan tahap validasi. Data *testing* memiliki tujuan untuk mengukur kemampuan model secara objektif dengan cara model akan diberikan data yang belum pernah dilihat sebelumnya untuk melakukan evaluasi terhadap kemampuan dari sebuah model.

2.7 Confusion Matrix

Confusion matrix merupakan sebuah alat performa dalam bentuk tabel dan berfungsi untuk melakukan evaluasi kinerja dari model yang telah dibangun dalam bentuk dua kelas atau lebih [23] dengan cara membandingkan nilai aktual dan nilai hasil prediksi yang dihasilkan oleh model [24]. Nilai prediksi merupakan sebuah data dari model prediksi yang digunakan untuk memperkirakan suatu nilai, sedangkan nilai aktual merupakan nilai yang sebenarnya dan telah dibuktikan kebenarannya. Pada *confusion matrix*, terdapat empat istilah yang merupakan hasil representasi dari proses klasifikasi. Istilah tersebut antara lain sebagai berikut:

1. *True Positive (TP)* - Data diprediksi positif dan benar
2. *True Negative (TN)* - Data diprediksi negatif dan benar
3. *False Positive (FP)* - Data diprediksi positif dan salah
4. *False Negative (FN)* - Data diprediksi negatif dan salah

Selain itu, *confusion matrix* juga membantu dalam mencari nilai *accuracy*, nilai *precision*, serta nilai *recall* dari sebuah model. *Accuracy* merupakan salah satu metrik yang digunakan untuk mengukur seberapa sering sebuah model menghasilkan prediksi dengan benar dari keseluruhan data yang diuji. Rumus dari *accuracy* dapat dilihat pada Rumus 2.1.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Precision merupakan salah satu metrik yang digunakan untuk mengukur kualitas dari prediksi yang bersifat *true positive* dari suatu model dibandingkan

dengan seluruh hasil yang bersifat positif. Sehingga, *precision* dapat menggambarkan seberapa akurat prediksi prositif yang dibentuk oleh model. Rumus dari *precision* dapat dilihat pada Rumus 2.2.

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall merupakan salah satu metrik yang digunakan untuk mengukur kemampuan sebuah model dalam melakukan identifikasi seluruh kasus positif aktual dalam *dataset*. Rumus dari *recall* dapat dilihat pada Rumus 2.3.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

Selain *accuracy*, *precision*, serta *recall*, terdapat metrik *F1-Score*. *F1-Score* merupakan sebuah metrik yang menggabungkan antara *precision* dan *recall* dalam sebuah metrik untuk menyeimbangkan keduanya, terutama untuk data yang bersifat tidak seimbang atau *imbalanced* [13]. Rumus dari *F1-Score* dapat dilihat pada Rumus 2.4.

$$F1 - Score = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.4}$$