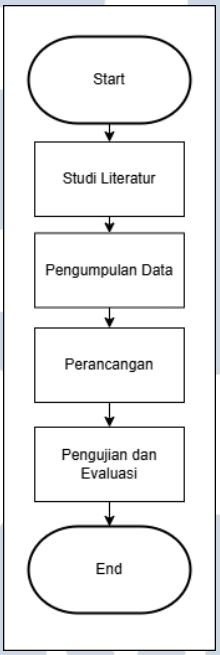
BAB 3
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian ini memberikan gambaran secara ringkas mengenai seluruh tahapan dari menyusun dan melakukan penelitian. Tahapan-tahapan tersebut adalah studi literatur, pengumpulan data, perancangan, pengujian dan evaluasi. Hal tersebut dapat dilihat pada tahapan pada Gambar 3.1.



Gambar 3.1. Tahapan Penelitian

A. Studi Literatur

Tahap studi literatur merupakan sebuah tahap yang digunakan untuk memberikan gambaran dan kerangka terhadap penelitian yang akan dilakukan. Tahap ini akan digunakan untuk mengidentifikasi dan menganalisis kesalahan atau keterbatasan yang sudah terjadi pada penelitian-penelitian terdahulu untuk menghindari pengulangan kesalahan yang sama. Selain itu, tahap studi literatur juga digunakan untuk membangun landasan dalam penelitian.

B. Pengumpulan Data

Pada penelitian ini, *dataset* yang digunakan berasal dari *Kaggle* dengan judul *Brain Stroke Dataset* yang berisikan 11 atribut dan 4981 baris *sample data*. Atribut yang terdapat pada *dataset* tersebut antara lain adalah *Gender*, *Age*, *Hypertensi*, *Heart Disease*, *Ever Married*, *Work Type*, *Residence Type*, *Average Glucose Level*, *BMI*, *Smoking Status*, dan *Stroke*.

C. Perancangan

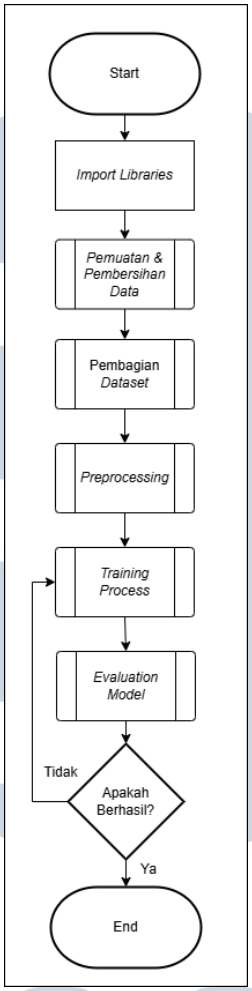
Pada tahap ini akan berfokus pada perancangan dan implementasi sistem yang akan dijadikan sebagai solusi yang efektif dari permasalahan yang telah diidentifikasi. Perancangan akan berfokus pada penggunaan algoritma *Decision Tree* untuk melakukan deteksi penyakit strok pada pria.

D. Pengujian dan Evaluasi

Pada tahap pengujian dan evaluasi ini, algoritma *Decision Tree* digunakan untuk memvalidasi kinerja model yang telah dikembangkan dalam mendeteksi penyakit strok pada pria serta mengukur akurasi model yang telah dikembangkan. Pengujian akan dilakukan dengan menggunakan *dataset* yang dibagi menjadi tiga bagian, yaitu data *training* untuk melatih model, data *testing* untuk menguji performa awal model, dan data *validation* untuk memastikan kemampuan model terhadap data baru. Setelah proses pengujian telah dilakukan, evaluasi akan dilakukan dengan menghitung nilai *accuracy*, *precision*, *recall*, dan *F1-Score* untuk mengetahui seberapa baik model berfungsi dalam mendeteksi penyakit strok pada pria.

3.2 Pembangunan Model

Pada penelitian ini dilakukan sebuah pembangunan model yang bertujuan untuk mendeteksi penyakit strok pada pria dengan menggunakan algoritma *Decision Tree*. Proses pembangunan model terdiri dari beberapa tahap, yaitu *import libraries*, pemuatan dan pembersihan data, pembagian *dataset*, *preprocessing*, *training process*, dan *evaluation model*. Hal tersebut dapat dilihat pada *flowchart* pada Gambar 3.2.



Gambar 3.2. *Flowchart* Pembangunan Model

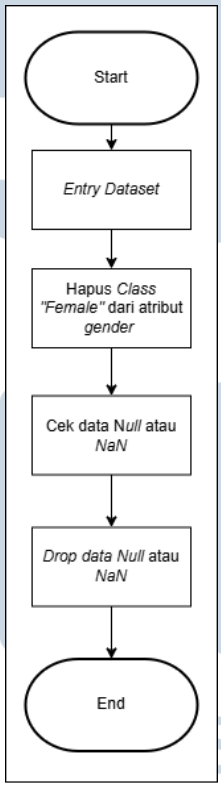
A. *Import Libraries*

Pada tahap ini, proses dimulai dengan melakukan *import library* yang digunakan untuk merancang kode, seperti *pandas* untuk melakukan pengolahan data dan manipulasi data, *scikit-learn* atau *sklearn* untuk membangun dan melatih model, *matplotlib* dan *seaborn* untuk melakukan visualisasi dari model yang telah dibangun, dan *imbalanced-learn* atau *imblearn* yang digunakan untuk menangani data yang tidak seimbang.

B. Pemuatan dan Pembersihan Data

Pada tahap ini, dilakukan persiapan *dataset* yang akan digunakan sebagai data *training*, *testing*, dan *validation*. *Dataset* yang digunakan berasal dari sumber yang sudah ada, yaitu berasal dari *Kaggle*. Dalam *dataset* tersebut terdapat 4981 data. Pada *dataset* tersebut, terdapat 2907 data dalam *class female*

dengan strok sebanyak 140 data yang dikategorikan sebagai 1 dan tidak strok sebanyak 2767 data yang dikategorikan sebagai 0. Sedangkan pada *class male*, terdapat 2074 data dalam *class male* dengan strok sebanyak 108 data yang dikategorikan sebagai 1 dan tidak strok sebanyak 1966 data yang dikategorikan sebagai 0. Setelah proses tersebut telah selesai, selanjutnya adalah proses menghapus data dengan *class female* dari atribut *gender*, sehingga *class* yang digunakan hanya *male*. Kemudian, tahap terakhir pada bagian pemuatan dan pembersihan data adalah pengecekan apakah terdapat nilai data yang kosong (*null*). Apabila terdapat data yang bersifat *null*, maka baris tersebut akan dihapus. Sehingga, jumlah data setelah dilakukan pemuatan dan pembersihan data adalah 2074 data. Hal tersebut dapat dilihat pada Gambar 3.3.

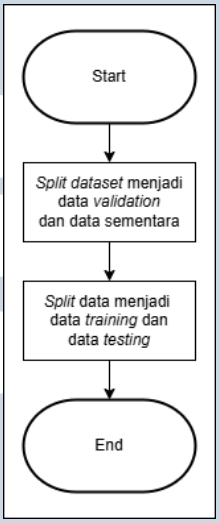


Gambar 3.3. *Flowchart* Pemuatan dan Pembersihan Data

C. Pembagian *Dataset*

Pada tahap ini, *dataset* dibagi menjadi 3, yaitu data *training*, data *validation*, dan data *testing*. Proses pembagian *dataset* dilakukan secara bertahap. Mulanya, sebanyak 10% dari *dataset* diambil untuk menjadi data *validation*.

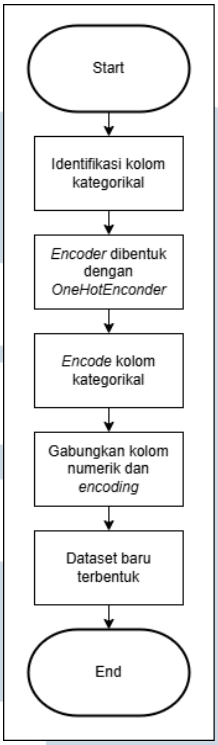
Kemudian, sebanyak 90% *dataset* yang tersisa dibagi menjadi menjadi data *training* dan data *testing* dengan rasio 80:20. Hal tersebut dapat dilihat pada Gambar 3.4.



Gambar 3.4. *Flowchart* Pembagian *Dataset*

D. *Preprocessing*

Pada tahap ini, fitur-fitur yang bersifat kategorikal akan diubah menjadi numerik dengan menggunakan *encoder*. Proses *preprocessing* dimulai dengan membentuk *encoder* dengan *OneHotEncoder* untuk melakukan identifikasi kategori unik pada data *training*. Kemudian, langkah selanjutnya adalah mengubah kolom kategorikal pada data *training*, data *validation*, dan data *testing* menjadi biner. Seluruh kategori unik akan diubah menjadi kolom baru yang berisi nilai 0 dan 1. Setelah diubah menjadi kolom baru, kolom tersebut digabung dengan kolom yang sudah bersifat numerik sehingga menghasilkan *dataset* baru. Hal tersebut dapat dilihat pada Gambar 3.5.

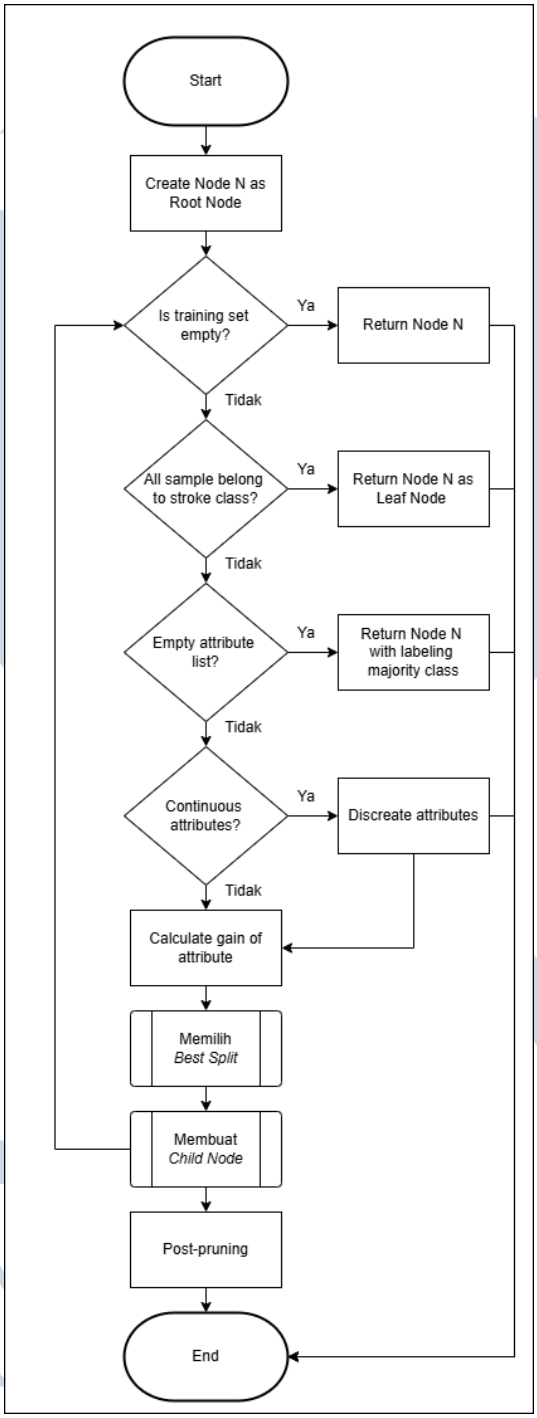


Gambar 3.5. *Flowchart Preprocessing*

E. *Training Process*

Pada tahap ini, dilakukan *training* terhadap model dengan menggunakan algoritma *Decision Tree* seperti yang dapat dilihat pada Gambar 3.6. Proses *training* dimulai dengan membuat *Node N* sebagai *Root Node*. Kemudian, seluruh *Node N* akan dilakukan evaluasi terhadap beberapa kondisi. Kondisi pertama adalah pengecekan apakah *training set* kosong. Apabila kosong, maka *Node N* dikembalikan. Kondisi kedua adalah pengecekan apakah seluruh sampel dalam *Node N* termasuk dalam *class* strok yang sama untuk menunjukkan kemurnian dari sebuah data. Apabila benar, *Node N* akan dikembalikan sebagai *Leaf Node* yang melakukan prediksi *class* tersebut. Kondisi ketiga adalah pengecekan daftar atribut. Apabila daftar atribut kosong, maka *Node N* akan dikembalikan sebagai *Leaf Node* yang melabeli kelas mayoritas dari sampel-sampel di *node* tersebut. Kondisi terakhir yaitu pengecekan apakah terdapat atribut numerik yang belum digunakan. Apabila terdapat atribut numerik yang belum digunakan, atribut-atribut tersebut akan diubah menjadi *discreate attributes*. Apabila seluruh kondisi tersebut tidak terpenuhi, model akan menghitung *gain* dari setiap atribut yang tersisa untuk mengukur potensi dari pemisahan data. Hasil dari menghitung *gain* tersebut

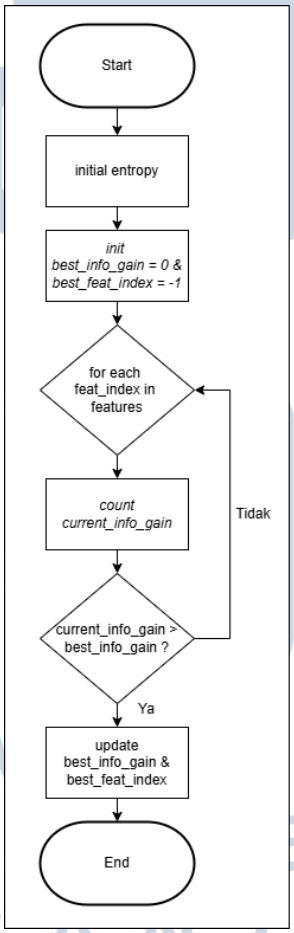
akan digunakan untuk memilih *best split* yang paling optimal untuk *Node N*.



Gambar 3.6. *Flowchart Training Process*

Proses pemilihan *best split* dimulai dengan melakukan inisialisasi *entropy* dari set data. Selanjutnya, inisialisasi *best_info_gain* dengan nilai 0

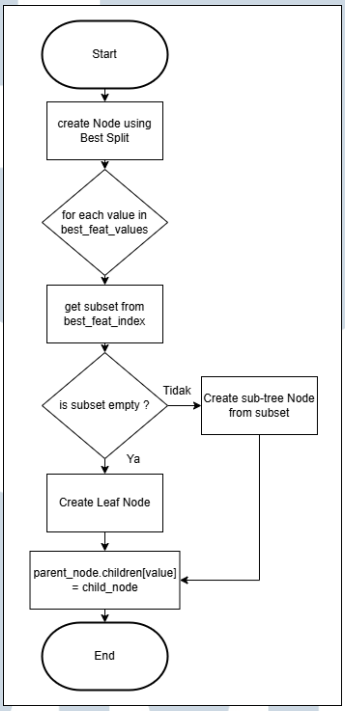
dan *best_feat_index* dengan nilai -1 yang akan digunakan untuk melacak *information gain* terbaik dan indeks fitur terbaik. Kemudian, algoritma akan melakukan iterasi. Dalam setiap iterasi, nilai *information gain* dihitung untuk fitur yang sedang dipertimbangkan. Nilai dari *information gain* tersebut akan dibandingkan dengan indeks fitur terbaik. Apabila *information gain* lebih besar, maka *best_info_gain* dan *best_feat_index* diperbarui dengan nilai saat ini. Proses iterasi akan berlanjut hingga seluruh fitur telah dievaluasi. Setelah seluruh fitur dihitung dan dibandingkan, model akan memiliki *best_info_gain* dan *best_feat_index* yang mewakili *information gain* tertinggi serta fitur yang menghasilkan *split* terbaik. Fitur yang memiliki *information gain* tertinggi akan dipilih sebagai *best split* untuk membagi *node*. Hal tersebut dapat dilihat pada Gambar 3.7



Gambar 3.7. *Flowchart* Memilih *Best Split*

Setelah *best split* sudah dipilih, selanjutnya membuat *child node*. Pembuatan *child node* dimulai dengan membuat *node* berdasarkan pada *best split*

yang telah diperoleh pada tahap sebelumnya. Kemudian, algoritma akan melakukan iterasi yang merepresentasikan setiap nilai atau kategori dari fitur yang terpilih sebagai *best split*. Untuk setiap nilai ini, dilakukan pengambilan *subset* data yang hanya memiliki sampel-sampel dengan nilai fitur tersebut. Selanjutnya dilakukan pengecekan apakah *subset* kosong. Apabila kosong, maka tidak ada data untuk cabang tersebut sehingga langkah selanjutnya adalah membuat *Leaf Node*. Namun, jika tidak kosong, maka algoritma akan membuat *sub-tree Node* dari *subset*. *Child node* yang telah terbentuk akan dihubungkan dengan *parent node*. Hal tersebut dapat dilihat pada Gambar 3.8



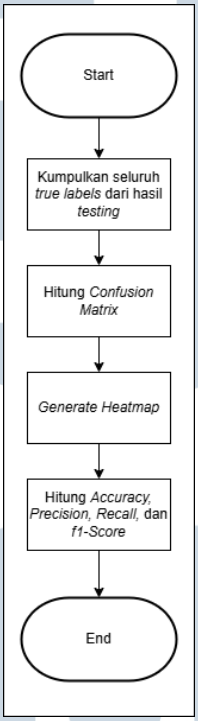
Gambar 3.8. *Flowchart* Membuat *Child Node*

Setelah *child node* terbentuk, proses pembangunan akan diulang secara rekursif untuk setiap *child node* yang baru dibuat. Proses ini terus diulang hingga seluruh cabang pohon mencapai salah satu kondisi berhenti. Setelah pohon dari *decision tree* selesai dibangun, tahap *post-pruning* dilakukan untuk meningkatkan generalisasi model dan mencegah *overfitting*.

F. *Evaluation Model*

Tahap *evaluation model* merupakan tahap terakhir dalam penelitian ini. Tahap ini bertujuan untuk mengukur, melakukan evaluasi dan analisa terhadap

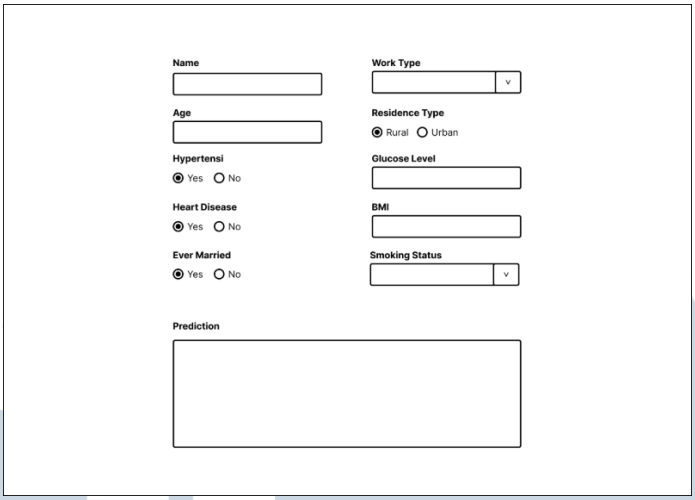
kinerja dari model yang telah dibangun. Untuk mengukur kinerja dari model tersebut, dilakukan dengan penghitungan *confusion matrix* dengan cara mengumpulkan *true labels* dari *dataset testing* yang telah dilakukan sebelumnya. *Confusion matrix* dapat memberikan gambaran mengenai prediksi yang benar dan prediksi yang salah. Kemudian, hasil dari *confusion matrix* dilakukan visualisasi dengan menggunakan *heatmap*. *Heatmap* dapat membantu dalam melakukan analisa performa model, serta dapat menunjukan letak kesalahan dari model dalam melakukan klasifikasi. Hal tersebut dapat dilihat pada Gambar 3.9.



Gambar 3.9. *Flowchart Evaluation Model*

3.3 Perancangan Wireframe

Pada penelitian ini dilakukan sebuah perancangan *wireframe* yang selanjutnya menjadi *website* yang bertujuan untuk melakukan prediksi risiko terkena strok berdasarkan data *input* dari pengguna. *Wireframe* dari *website* tersebut dapat dilihat pada Gambar 3.10.



Gambar 3.10. *Wireframe Website*

Pada *website* untuk deteksi penyakit strok, pengguna akan diminta untuk melakukan *input* data seperti *Name* untuk identitas, *Age*, *Hypertensi* dan *Heart Disease* dalam bentuk pilihan "*No*" atau "*Yes*". Pengguna dapat memilih *No* jika tidak hipertensi atau tidak sakit jantung. Apabila pengguna memiliki hipertensi atau penyakit jantung, maka dapat memilih *Yes*. Selain itu, pengguna juga diminta untuk memilih status pernikahan atau *Ever Married*, jenis pekerjaan atau *Work Type*, dan tempat tinggal atau *Residence Type* yang dibedakan antara pedesaan atau *Rural* dan perkotaan atau *Urban*. Kemudian, pengguna juga diminta untuk mengisi kadar glukosa atau *Average Glucose Level*, *BMI*, dan *Smoking Status*. Setelah seluruh data terisi, selanjutnya, pengguna akan mendapatkan *output* berupa hasil prediksi penyakit strok dari pengguna.