Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*

SKRIPSI

Disusun Sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Komputer
pada Departemen Ilmu Komputer/ Informatika

Disusun Oleh:
Dhiyaussalam
24010316130061

DEPARTEMEN ILMU KOMPUTER/ INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
2020

HALAMAN PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan di bawah ini:

Nama : Dhiyaussalam

NIM : 24010316130061

Judul : Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*

Dengan ini saya menyatakan bahwa dalam skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang tertulis diacu dalam naskah ini dan disebutkan di dalam daftar pustaka.

Semarang, __ Juni 2020

Dhiyaussalam
24010316130061

HALAMAN PENGESAHAN

Judul : Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*
Nama : Dhiyaussalam
NIM : 24010316130061

Telah diujikan pada sidang skripsi pada tanggal 30 Juni 2020 dan dinyatakan lulus pada tanggal __ Juni 2020

Semarang, __ Juli 2020

Mengetahui,	
Ketua Departemen Ilmu Komputer/ Informatika	Panitia Penguji Skripsi
FSM UNDIP	Ketua,
Dr. Retno Kusumaningrum, S.Si, M. Kom	Nama Ketua Penguji
NIP. 198104202005012001	NIP.

HALAMAN PENGESAHAN

Judul : Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*

Nama : Dhiyaussalam

NIM : 24010316130061

Telah diujikan pada sidang skripsi pada tanggal 30 Juni 2020

Semarang, __ Juli 2020

Pembimbing

Dr. Eng. Adi Wibowo, S.Si., M.Kom
NIP. 198203092006041002

ABSTRAK

Sakit kepala merupakan salah satu penyakit yang sangat sering dijumpai. Sebanyak 50% penduduk dunia pernah mengalami sakit kepala. Sakit kepala primer sendiri memiliki beberapa jenis yaitu *migrain, tension, cluster,* dan *medication overuse*. Diagnosis menggunakan komputer dapat mempermudah penderita dalam mengetahui jenis sakit kepala yang diderita tanpa perlu bertemu dengan dokter. Algoritma *Random Forest* digunakan dalam penelitian ini untuk menghasilkan model yang dapat diandalkan untuk melakukan klasifikasi terhadap jenis sakit kepala yang diderita. Dalam pembangunan model *Random Forest* yang digunakan, dilakukan pengaturan secara manual terhadap beberapa *parameter* untuk menghasilkan model *Random Forest* terbaik. *Dataset* yang digunakan untuk membangun model *Random Forest* dalam penelitian ini adalah *Migbase Dataset* yang kemudian dilakukan pra-pemrosesan terhadap *dataset* tersebut. Hasil yang diperoleh adalah sebuah model *Random Forest* dengan kinerja terbaik memiliki nilai akurasi sebesar 99,12%. Untuk mencapai model dengan akurasi 99,12% perlu dilakukan pengaturan secara manual terhadap beberapa *parameter*. Dengan demikian, model *Random Forest* terbaik yang dihasilkan dapat digunakan oleh dokter untuk melakukan klasifikasi jika dokter tidak dapat melakukan klasifikasi dengan pasti. Model *Random Forest* terbaik yang dihasilkan juga dapat digunakan oleh masyarakat umum untuk melakukan diagnosis dini agar dapat melakukan penanganan yang tepat dan efisien.

Kata Kunci : Sakit Kepala Primer, Klasifikasi, *Machine Learning*, *Random Forest*

ABSTRACT

Headache disorder is one of the most often disorder. At least 50% of the world's population have experience headache. Primary headaches have several types, namely migraine, tension, cluster, and medication overuse. Diagnosis using computer could help sufferers find the type of the headache without the need to meet doctor. The Random Forest algorithm was used in this study to produce a reliable model for classifying the types of headaches. In the development of the Random Forest model that will be used, a number of parameters are tunned manually to produce the best Random Forest model. The dataset used to develop the Random Forest model in this study is the Migbase Dataset which is pre-processed before. The results obtained are the Random Forest model with the best accuracy reaching 99,12%. In order to reach the model with 99,12% accuracy, it is necessery to manually tunned the value of a few parameters. Thus, the best Random Forest model produced can be used by doctors to classify if doctors can't classify with certainly. The best Random Forest model can also be used by general public to carry out an early diagnosis in or der to make an appropriate and efficient treatment.

Keyword : Primary Headache, Classification, Machine Learning, Random Forest

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah Subhanahu wa ta'ala, karena atas Rahmat dan Ridho-Nya, penulis dapat menyelesaikan laporan skripsi yang berjudul "Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*" dengan baik dan lancar. Laporan skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar sarjana strata satu pada Departemen Ilmu Komputer/ Informatika Fakultas Sains dan Matematika Universitas Diponegoro Semarang.

Penulis menyadari bahwa sangat sulit bagi saya untuk menyelesaikan Karya Tulis Ilmiah ini tanpa bantuan, dukungan, dan bimbingan dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati, penulis menyampaikan terimakasih kepada:

1. Prof. Dr. Widowati, M.Si, selaku Dekan FSM UNDIP.
2. Dr. Retno Kusumaningrum, S.Si, M.Kom, selaku Ketua Departemen Ilmu Komputer/ Informatika.
3. Panji Wisnu Wirawan, S.T, M.T, selaku Koordinator Skripsi.
4. Dr. Eng. Adi Wibowo, S.Si, M.Kom, selaku dosen pembimbing yang telah menyediakan waktu, tenaga dan pikiran untuk membimbing penulis.
5. Kedua Orangtua saya, Bapak Mukhyar dan Ibu Ellya Roliantinie yang telah membimbing, mendukung, dan senantiasa mendoakan saya.

Penulis menyadari bahwa dalam laporan ini masih banyak kekurangan baik dari penyampaian materi maupun isi dari materi itu sendiri. Hal ini dikarenakan keterbatasan kemampuan dan pengetahuan dari penulis. Oleh karena itu, kritik dan saran yang bersifat membangun sangat penulis harapkan. Semoga laporan skripsi ini dapat bermanfaat bagi penulis dan juga pembaca pada umumnya.

Semarang, 30 Juni 2020

Dhiyaussalam

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Diponegoro, saya yang bertanda tangan di bawah ini:

Nama : Dhiyaussalam
NIM : 24010316130061
Program Studi : Ilmu Komputer/ Informatika
Jenis Karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*)** kepada Universitas Diponegoro atas karya ilmiah saya yang berjudul:

Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*

Beserta perangkat yang ada (jika diperlukan). Dengan Hak bebas Royalti Noneksklusif ini Universitas Diponegoro berhak menyimpan, mengalihmedia/ formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/ pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya

Semarang, __ Juni 2020

Yang Menyatakan

Dhiyaussalam

24010316130061

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

Bab ini membahas mengenai latar belakang, rumusan masalah, manfaat dan tujuan, ruang lingkup, dan sistematika penulisan pada skripsi yang berjudul klasifikasi jenis sakit kepala menggunakan algoritma *Random Forest*.

1.1. Latar Belakang

Sakit kepala merupakan gajala gangguan sistem saraf yang terjadi pada bagian kepala. Sakit kepala dapat dialami oleh segala rentang usia, ras, dan status ekonomi dan lebih sering terjadi pada wanita. Sakit kepala terjadi terhadap hampir 50% populasi dan merupakan gejala tertinggi ketiga di dunia. Sakit kepala primer memiliki 4 jenis yaitu *migrain, tension, cluster*, dan *medication-overuse headache*. Secara statistik jenis sakit kepala primer yang paling sering terjadi adalah *tension* dengan angka sebesar 60-80%, diikuti oleh *migrain* sebesar 15% dan *cluster* sebesar 0,1%. *Medication-overuse headache* terjadi bersamaan dengan sakit kepala primer lainnya (Ahmed, 2012).

Kurangnya pengetahuan tentang gejala sakit kepala primer merupakan hambatan dalam memilih pengobatan yang tepat. Di seluruh dunia, rata-rata, hanya 4 jam didedikasikan untuk penanganan sakit kepala dan hanya sebesar 50% dari sakit kepala didiagnosis dan ditangani. Gejala sakit kepala dianggap masyarakat sebagai masalah yang tidak serius. Setengah dari penderita sakit kepala memilih untuk mengobati sendiri dan jumlah orang yang menyadari bahwa ada pengobatan yang efektif begitu rendah. Jika sakit kepala tidak diobati dapat menyebabkan penderita mengalami kegelisahan hingga depresi (WHO, 2016).

Peran komputer dalam melakukan diagnosis terhadap suatu penyakit dapat meningkatkan mutu dari pelayanan kesehatan. Diagnosis yang dilakukan sedini mungkin dapat mempermudah penanganan terhadap penyakit. Penggunaan komputer sebagai diagnosis dini dapat mempermudah penderita penyakit karena mempermudah proses diagnosis. Penderita tidak perlu bertemu langsung dengan dokter untuk melakukan diagnosis dan cukup menggunakan komputer saja sehingga proses diagnosis lebih efektif dan cepat (Patra et al., 2010). Komputer dapat digunakan sebagai alat bantu dokter dalam melakukan diagnosis sakit kepala primer dengan tepat. Dengan adanya peran komputer untuk

melakukan diagnosis sakit kepala primer akan mempermudah pekerjaan dokter yang hanya perlu melakukan anamnesis. Bahkan penderita bisa melakukan diagnosis sakit kepala primer secara mandiri agar bisa melakukan penanganan yang tepat dengan menggunakan komputer (Krawczyk et al., 2013).

Dalam dekade terakhir, sejumlah algoritma klasifikasi dalam pembelajaran mesin digunakan untuk mengklasifikasikan jenis sakit kepala primer. Vandewiele et al., (2018) menggunakan beberapa algoritma untuk menghasilkan model dan menyarankan untuk menggunakan *Decision Tree* sebagai model yang digunakan untuk melakukan klasifikasi. Krawczyk et al., (2013) melakukan klasifikasi dengan beberapa algoritma dan menggunakan *feature selection algorithms* dan nilai akurasi tertinggi dihasilkan menggunakan algoritma *random forest*. Aljaaf et al., (2015) melakukan penelitian dengan menggunakan sejumlah algoritma dan *dataset* buatan berdasarkan *The International Classification of Headache Disorders* (ICHD-2) dan menyarankan *Decision Tree* sebagai metode yang memilki akurasi tertinggi.

Salah satu metode pembelajaran mesin terbaik untuk klasifikasi jenis sakit kepala primer adalah *Random Forest*. *Random Forest* memiliki kinerja yang lebih baik dibandingkan dengan sebagian besar model klasifikasi tunggal. Dengan *Random Forest* dimungkinkan untuk menghasilkan sistem yang dapat melakukan diagnosis sakit kepala primer secara otomatis yang akurasinya sebanding atau bahkan lebih baik dari dokter (Krawczyk et al., 2013).

Random Forest merupakan algoritma yang sangat berhasil dalam melakukan metode klasifikasi dan regresi. *Random Forest* juga memiliki tingkat efisiensi yang tinggi untuk permasalahan regresi dan kasifikasi. *Random Forest* memiliki kinerja yang sangat bagus dengan jumlah data yang besar. Dengan melakukan pengaturan dalam beberapa nilai akan meningkatkan kinerja dari algoritma ini (Breiman, 2001).

Pada penelitian ini akan digunakan algoritma *Random Forest* karena tidak rentan terhadap *overfitting*. *Overfitting* sendiri rentan terjadi terhadap *dataset* yang memiliki fitur dengan data yang bervariasi (Raschka & Mirjalili, 2017). Dengan *dataset* kesehatan yang digunakan memiliki variasi yang cukup tinggi sehingga diperlukan model terbaik agar klasifikasi jenis sakit kepala dapat melakukan klasifikasi dengan akurasi yang tinggi. Untuk

menghasilkan model *Random Forest* terbaik perlu dilakukan pengaturan *hyper-parameter* dan pada peneitian ini beberapa parameter diatur secara manual.

1.2. Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan suatu masalah yaitu bagaimana mendapatkan model terbaik untuk melakukan klasifikasi jenis sakit kepala menggunakan algoritma *Random Forest*.

1.3. Manfaat dan Tujuan

Tujuan dari skripsi ini adalah untuk menghasilkan model terbaik untuk melakukan klasifikasi jenis sakit kepala menggunakan algoritma *Random Forest*.

Manfaat yang diharapkan dari skripsi ini adalah model yang dihasilkan dapat melakukan klasifikasi jenis sakit kepala dengan akurat

1.4. Ruang Lingkup

Ruang lingkup mempunyai tujuan memberikan batasan dari skripsi ini agar tidak menyimpang dan sesuai dengan target yang diinginkan. Ruang lingkup dalam skripsi ini adalah sebagai berikut :

1. Data yang digunakan adalah *Migbase Dataset*.
2. Model *Random Forest* yang dibangun berbasis pemrograman *Python* dengan memanfaatkan *library Scikit-learn*.
3. Pelatihan dan pengujian model *Random Forest* dilakukan pada lingkungan *Jupyter Notebook.*
4. Beberapa parameter akan dioptimasi secara manual.
5. Akurasi digunakan untuk mengevaluasi kinerja model *Random Forest*.

1.5. Sistematika Penulisan

Sistematika penulisan memberikan gambaran laporan dari skripsi ini secara urut dan jelas. Berikut adalah sistematika penulisan dari skripsi ini ini :

BAB I PENDAHULUAN

Bab ini membahas mengenai latar belakang, rumusan masalah, manfaat dan tujuan, ruang lingkup, dan sistematika penulisan dari skripsi ini.

BAB II TINJAUAN PUSTAKA

Bab ini membahas teori-teori yang berhubungan dan mendukung topik atau masalah yang dibahas pada skripsi ini.

BAB III METODOLOGI PENELITIAN

Bab ini menyajikan informasi mengenai metodologi penelitian yang digunakan pada penelitian klasifikasi jenis sakit kepala menggunakan algoritma *Random Forest*.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menyajikan informasi tentang lingkungan dan perangkat yang digunakan untuk penelitian ini serta menyajikan analisis hasil penelitian tentang pengaturan secara manual *hyper-parameter* terhadap model yang dihasilkan.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan dari bab-bab yang telah dibahas dan berisi saran sebagai bahan masukan untuk pengembangan atau penelitian selanjutnya.

BAB II
TINJAUAN PUSTAKA

Bab ini akan membahas tentang segala dasar teori yang berhubungan dengan permasalahan atau topik pada skripsi ini. Pada bab ini akan terdiri dari beberapa sub bab diantaranya mengenai penyakit kepala, *Random Forest*, *confusion matrix*, *library*, dan *state-of-the-art*.

2.1. Sakit Kepala

Sakit kepala merupakan salah satu jenis penyakit yang sangat sering ditemui pada bidang neurologi. Sakit kepala merupakan gangguan pada saraf di bagian kepala yang menyebabkan rasa sakit dan dapat mengakibatkan kelumpuhan sementara atau kecil. Ditemukan sejak jaman dulu oleh *Aretaeus of Cappadocia* (Rizzoli & Mullally, 2018).

Terdapat 3 jenis sakit kepala primer utama dengan jenis terbanyak adalah *migrain* dan *tension*. *Cluster* merupakan jenis terakhir yang sangat jarang dialami tetapi sering terjadi kesalahan diagnosis dan penanganannya. Sakit kepala primer lainnya yaitu *medication-overuse headache (MOH)* yang terjadi bersamaan dengan 3 jenis sakit kepala primer utama (Ahmed, 2012).

Untuk melakukan klasifikasi terhadap 3 jenis utama sakit kepala primer diperlukan beberapa kriteria yang terpenuhi untuk setiap jenisnya. Kriteria utama yang digunakan adalah durasi dari sakit kepala sendiri (Olesen, 2018).

2.1.1. *Migrain*

Migrain merupakan salah satu jenis sakit kepala primer yang sering terjadi. Banyak pakar epidemologis yang medokumentasikan prevalensi tinggi dan berdampak pada sosial-ekonomi seseorang. *Migrain* memiliki beberapa karakteristik pada gejala yang dihasilkan, yaitu (Olesen, 2018):

a. Sedikitnya terjadi 5 kali.
b. Sakit kepala terjadi selama 4 – 72 jam saat tidak dilakukan penanganan.
c. Sakit kepala memiliki sedikitnya 2 dari 4 gejala berikut:
    1) Terjadi pada 1 lokasi
    2) Memiliki karakter berdenyut

3) Memiliki intensitas rasa sakit yang ringan maupun sedang
4) Merasa berat disebabkan atau melakukan suatu aktivitas

d. Saat terjadi sakit kepala sedikitnya memiliki salah satu gejala berikut:
1) Mual dan/ atau muntah
2) Sensitif terhadap cahaya dan suara

2.1.2. *Tension*

Tension merupakan salah satu jenis sakit kepala primer yang sering terjadi. *Tension* memiliki dampak yang tinggi dalam kehidupan sosial-ekonomi. Berikut merupakan beberapa karakteristik dari sakit kepala primer jenis *tension* (Olesen, 2018):

a. Terjadi paling sedikit 10 kali (<1 hari per bulan dengan rata-rata <12 hari per tahun).
b. Sakit kepala terjadi selama 30 menit hingga 7 hari.
c. Sakit kepala memiliki paling sedikit 2 dari gejala berikut:
1) Terjadi pada kedua sisi kepala
2) Memiliki karakter menekan atau diikat dengan ketat
3) Memiliki intensitas rasa sakit yang sedang maupun ringan
4) Tidak disebabkan oleh suatu aktivitas

d. Memiliki kedua gejala berikut:
1) Tidak mual atau muntah
2) Hanya memiliki salah satu gejala sensitif terhadap cahaya atau suara

2.1.3. *Cluster*

Cluster merupakan jenis sakit kepala primer yang jarang terjadi. *Cluster* memiliki karakterisitik sebagai berikut (Olesen, 2018):

a. Terjadi paling sedikit 5 kali.
b. Memiliki intensitas rasa sakit yang tinggi atau sangat tinggi pada satu bagian mata, diatas mata dan/ atau sakit dengan durasi 15 – 180 menit jika tidak dilakukan penanganan.
c. Memiliki dua gejala berikut:
1) Paling sedikit memiliki salah satu gejala berikut:
a) Mata merah dan/ atau mata berair
b) Pilek
c) Kelopak mata membengkak
d) Berkeringat pada bagian dahi dan wajah

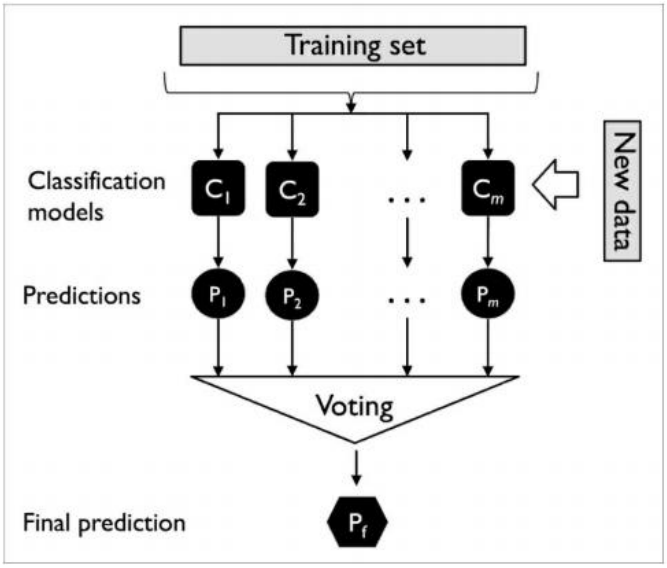
e) Pupil mengecil dan/ atau kelopak mata berat

2) Merasa resah atau gelisah

d. Terjadi dengan frekuensi satu setiap dua hari dan delapan jam per hari.

2.2. *Random Forest*

Random Forest adalah salah satu algoritma pembelajaran mesin yang menggunakan kombinasi dari banyak *Decision Tree* untuk mendapatkan hasil klasifikasi dan regresi. *Random Forest* memiliki kemajuan hasil akurasi klasifikasi yang signifikan berkat sekumpulan *Decision Tree* yang melakukan klasifikasi secara individu dan melakukan *voting* untuk mendapatkan hasil klasifikasi (Breiman, 2001). Hasil dari *Random Forest* lebih baik dan tidak rentan terhadap *overfitting* dibandingkan dengan *Decision Tree*. *Overfitting* merupakan masalah yang sering terjadi pada pembelajaran mesin dimana kinerja pada saat pengujian menggunakan data uji tidak sebagus kinerja pada saat pelatihan data menggunakan data latih. *Overfitting* sendiri terjadi dikarenakan tingginya variasi data dan/ atau banyaknya fitur yang digunakan pada saat pelatihan sehingga menghasilkan sebuah model yang kompleks (Raschka & Mirjalili, 2017). *Random Forest* sendiri termasuk dalam *ensemble learning*. *Ensemble learning* merupakan penggabungan beberapa model yang digunakan untuk membuat sebuah model tunggal yang lebih baik dari model tunggal biasa. Pada dasarnya, model-model tadi digunakan untuk melakukan pelatihan data dan melakukan *voting* sebagai hasil akhirnya (Frank, 2017). Ilustrasi dari *ensemble learning* dapat dilihat pada Gambar 2.1.



Gambar 2.1. Ilustrasi dari *Ensemble Learning*
(Raschka & Mirjalili, 2017)

2.2.1. Algoritma *Random Forest*

Random Forest merupakan *ensemble* dari *Decision Tree*. Gagasan utama dari *Random Forest* adalah hasil tertinggi dari sekumpulan *Decision Tree* untuk membangun model yang lebih baik dan tidak rentan terhadap *overfitting*. Berikut adalah 4 langkah utama dalam algoritma *Random Forest* (Raschka & Mirjalili, 2017):

1. Pilih secara acak sebanyak *n* sampel dari data latih dengan pengganti.
2. Latih data sampel tersebut menggunakan algoritma *Decision Tree* dan untuk setiap *node* lakukan:
   a. Pilih secara acak *d* fitur tanpa pengganti.
   b. Lakukan *split node* pada fitur yang menghasilkan nilai terbaik berdasarkan fungsi objektif.
3. Ulangi langkah 1 – 2 sebanyak *k* kali.
4. Label kelas prediksi dihasilkan dari pengambilan suara terbanyak dari masing-masing pohon.

Pada skripsi ini algoritma *Random Forest* diimplementasikan menggunakan *library Scikit-learn*. Berikut adalah persamaan-persamaan yang digunakan (Pedregosa et al., 2011):

1. Jumlah sampel yang digunakan adalah sebanyak jumlah baris data asli

   $$n = n_{baris\ data\ asli} \quad (2.1)$$

2. Nilai *d* ditentukan dengan persamaan berikut:

   $$d_{fitur} = \sqrt{jumlah\ fitur} \quad (2.2)$$

3. *Decision Tree* adalah algoritma *Cart Tree* (*Classification and Regression Trees*) yang menghasilkan pohon biner berdasarkan fitur dan nilai *threshold* dari nilai *information gain* tertinggi untuk setiap *node*.
   a. *Split data*:

   $$split\ \theta = (j, t_m) \quad (2.3)$$

   $$Q_{left}(\theta) = (x, y) \mid x_j \leq t_m \quad (2.4)$$

   $$Q_{right}(\theta) = (x, y) \mid x_j > t_m \quad (2.5)$$

   Keterangan:

   $split\ \theta$: *split* data
   j : fitur
   t_m : nilai *threshold* yang ditentukan pada *node m*

$Q_{left}(\theta)$: partisi data pada anak sebelah kiri dimana nilai x pada fitur *j* kurang dari atau sama dengan nilai *threshold* pada *node m*

$Q_{right}(\theta)$: partisi data pada anak sebelah kanan dimana nilai x pada fitur *j* lebih dari nilai *threshold* pada *node m*

x : data latih

y : label kelas

b. Nilai *impurity* dari *child node m* dihitung menggunakan *gini index* dengan persamaan berikut:

$$G_{child}(Q,\theta) = \frac{n_{left}}{N_m} H\left(Q_{left}(\theta)\right) + (\frac{n_{right}}{N_m} H\left(Q_{right}(\theta)\right) \quad (2.6)$$

$$H(X_m) = \sum_k P_{mk}(1 - P_{mk}) \quad (2.7)$$

$$P_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I\ (y_i = 1) \quad (2.8)$$

Keterangan:

$G_{child}(Q,\theta)$: *information gain child* atau *impurity node m*

n_{left} : jumlah partisi data pada anak sebelah kiri

n_{right} : jumlah partisi data pada anak sebelah kanan

N_m : banyaknya data pada *node m*

$Q_{left}(\theta)$: partisi data pada anak sebelah kiri dimana nilai x pada fitur *j* kurang dari atau sama dengan nilai *threshold* pada *node m*

$Q_{right}(\theta)$: partisi data pada anak sebelah kanan dimana nilai x pada fitur *j* lebih dari nilai *threshold* pada *node m*

$H(X_m)$: fungsi untuk mencari nilai *impurity*

k : nilai kelas label

P_{mk} : proporsi kelas label k pada *node m*

x : data latih

y : label kelas

I : banyak data yang nilai kelas label *y* sama dengan kelas *k*

i : region baris pada node m [0,1,2, ..., N_m-1]

Split data dilakukan dengan cara memilih parameter fitur dan *threshold node m* yang menghasilkan nilai *Informatin Gain Child* terkecil.

$$\theta^* = argmin_\theta G_{child}(Q,\theta) \quad (2.9)$$

Keterangan:

θ^* : nilai *information gain child* terkecil *node m*

$G_{child}(Q,\theta)$: *impurity child node m*

c. *Information Gain*:

$$IG(Q,\theta) = \frac{N_m}{N}\Big(H\big(Q(\theta)\big) - G_{child}(Q,\theta)\Big) \quad (2.10)$$

Keterangan:

$IG(Q,\theta)$: *information gain node m*

N_m : jumlah data pada *node m*

N : jumlah data

$H\big(Q(\theta)\big)$: *impurity parent* pada *node m*

$G_{child}(Q,\theta)$: *information gain child* pada node *m*

2.2.2. Fungsi *Feature Importance*

Feature importance dapat dihitung dari rata-rata pengurangan *impurity* dari seluruh *Decision Tree* yang ada pada sebuah *Random Forest* tanpa berasumsi tentang apakah data yang digunakan terpisah secara linear atau tidak (Raschka & Mirjalili, 2017). Untuk mencari nilai *feature importance* dapat dilihat pada persaman berikut ini (Ronaghan, 2018):

1. Menghitung nilai *importance* pada setiap *Decision Tree*:

$$FI_i = \frac{\sum_j IG_j}{\sum_k IG_k} \quad (2.11)$$

Keterangan:

FI_i : nilai *importance* untuk fitur *i* dalam *Decision Tree*

IG_j : nilai *imformation gain* dari fitur terpenting pada *node j*

j : *node* dalam *Decision Tree*

i : indeks fitur ke-i

k : semua *node* yang ada pada *Decision Tree*

2. Menghitung nilai *feature importance* pada *Random Forest*:

$$RFFI_i = \frac{\sum_j FI_{ij}}{T} \quad (2.12)$$

Keterangan:

$RFFI_i$: nilai *importance* untuk fitur *i* dalam *Random Forest*

FI_{ij} : fitur terpenting i dalam *Decision Tree j*

j : *Decision Tree* pada *Random Forest*

i : indeks fitur ke-i

T : jumlah *Decision Tree* pada *Random Forest*

2.3. *Confusion Matrix*

Confusion matrix merupakan sebuah matriks yang menjabarkan kinerja dari sebuah algoritma pembelajaran mesin (Raschka & Mirjalili, 2017). Ilustrasi dari *confusion matrix* dengan kelas multi dapat dilihat pada Gambar 2.2.

		Predicted Number			
		Class 1	Class 2	…	Class *n*
Actual Number	Class 1	x_{11}	x_{12}	…	x_{1n}
	Class 2	x_{21}	x_{22}	…	x_{2n}
	. . .	. . .	. . .	. . .	. . .
	Class *n*	x_{n1}	x_{n2}	…	x_{nn}

Gambar 2.2. *Cofusion Matrix* dengan Kelas Multi (Manliguez, 2016)

Dari *confusion matrix* dapat diperoleh nilai dari *false negative* (*TFN*), *false positive* (*TFP*), dan *true negative* (*TTN*) dengan persamaan sebagai berikut (Manliguez, 2016):

$$TFN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij} \quad \text{..... (2.13)}$$

$$TFP_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ji} \quad \text{..... (2.14)}$$

$$TTN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{\substack{k=1 \\ k \neq i}}^{n} x_{jk} \quad \text{..... (2.15)}$$

$$TTP_{all} = \sum_{j=1}^{n} x_{ji} \quad \text{..... (2.16)}$$

Dari persamaan diatas dapat ditentukan nilai akurasi untuk mengetahui kinerja dari algoritma. Akurasi merupakan rata-rata kinerja secara keseluruhan (Sokolova & Lapalme, 2009). Berikut adalah persamaan dari akurasi:

$$akurasi = \frac{TTP_{all}}{Total\ number\ of\ testing\ entries} \quad \text{..... (2.17)}$$

2.4. *Tools* dan *Library*

Tools dan *library* yang digunakan untuk membantu dan memudahkan dalam proses pengerjaan penelitian ini. Pada penelitian ini bahasa pemrograman yang digunakan adalah *Python* karena dikenal memiliki banyak *library.* Penelitian ini dijalankan dalam lingkungan *Jupyter Notebook* dengan menjalankan bahasa pemrograman *Python* versi 3.7.7. *Library* yang digunakan pada penelitian ini antara lain adalah *NumPy, Pandas,* dan *Scikit-Learn.*

2.4.1. *NumPy*

NumPy adalah sebuah *library* untuk komputasi ilmiah pada *Python* (Numpy, 2020). *NumPy* memberikan paket fungsi yang mendukung pemrosesan *array*, pengaturan multi-dimensi dan matriks, dan banyak operasi matematis maupun operasi logis. *Array* disimpan dengan cara yang lebih efisien daripada fungsi *list* dan *operasi* pada *Python*, selain itu juga lebih cepat dilakukan.

2.4.2. *Pandas*

Pandas adalah *library* untuk menyediakan struktur data yang cepat, fleksibel, dan ekspresif yang dirancang untuk membuat pekerjaan menjadi lebih terstruktur (Pandas, 2020). Tujuannya untuk menjadi dasar dari pembangunan data untuk melakukan analisis data dunia nyata yang praktis dengan *Python*. Selain itu, *pandas* memiliki tujuan yang lebih luas untuk menjadi alat analisis/ manipulasi data open source yang paling kuat dan fleksibel yang tersedia.

2.4.3. *Scikit-learn*

Scikit-learn adalah modul *Python* untuk pembelajaran mesin yang dibangun di atas *SciPy* (Scikit-learn, 2020). *Scikit-learn* menyediakan berbagai klasifikasi, regresi dan algoritma pengelompokan termasuk *Support Vector Machine, Random Forest, Gradient Boosting, K-Means* and *DBSCAN*, dan dirancang untuk beroperasi dengan *library NumPy* dan *SciPy*.

2.4.4. *Jupyter Notebook*

Jupyter Notebook merupakan aplikasi berbasis *web* untuk membuat, mengedit, dan menjalankan kode pemrograman yang mengandung pengkodingan secara langsung, persamaan, tampilan visual, dan teks narasi (Jupyter, 2020). *Jupyter Notebook* dapat

digunakan untuk melakukan pembersihan dan transformasi data, simulasi numerik, *statistical modeling*, penampilan visual, pembelajaran mesin, dan sebagainya.

2.5. *State-Of-The-Art*

Klasifikasi jenis sakit kepala sudah dilakukan oleh beberapa peneliti. Pada penelitian yang dilakukan oleh Krawczyk et al., (2013) menghasilkan nilai akurasi tertinggi dibandingkan algoritma lain dengan nilai akurasi 79,97% dengan menggunakan algoritma *Random Forest*. Penelitian Aljaaf et al., (2015) menghasilkan 3 algoritma terbaik dengan nilai akurasi 96,67%. Namun penelitian tersebut hanya bisa melakukan klasifikasi terhadap 2 label kelas. Terakhir, penelitian Vandewiele et al., (2018) menghasilkan 3 algoritma *terbaik* dengan kisaran nilai akurasi dari 98,11% - 98,35%. Untuk lebih lengkapnya dapat dilihat pada Tabel 2.1.

Tabel 2.1. Penelitian Tentang Klasifikasi Jenis Sakit Kepala Menggunakan Pembelajaran Mesin

No.	Penulis	Judul	*Dataset*	Algoritma Klasifikasi	Hasil
1	Krawczyk et al., (2013)	"*Automatic diagnosis of primary headaches by machine learning methods*"	*Dataset* yang dikumpulkan oleh penulis.	*Naïïve Bayes, C4.5, Support Vector Machine,* dan *Random Forest.*	Beberapa buah model klasifikasi dengan nilai akurasi rata-rata tertinggi diperoleh oleh *Random Forest* dengan nilai 79,97%.
2	Aljaaf et al., (2015)	"*A Systematic Comparison and Evaluation of Supervised Machine Learning Classifiers Using Headache Dataset*"	*Dataset* sintetis yang dihasilkan dari *International Classification of Headache Disorder* (*ICHD*).	*Naïïve Bayes, Neural Network, Decision Tree, Zero R Classifier, Support Vector Machine* (*SVM*), *k-Nearest*	Beberapa buah model klasifikasi dengan nilai akurasi tertinggi diperoleh oleh *Neural Network, Decision Tree,* SVM, dan *kNN* dengan nilai 96,67%. Model yang dihasilkan hanya dapat melakukan

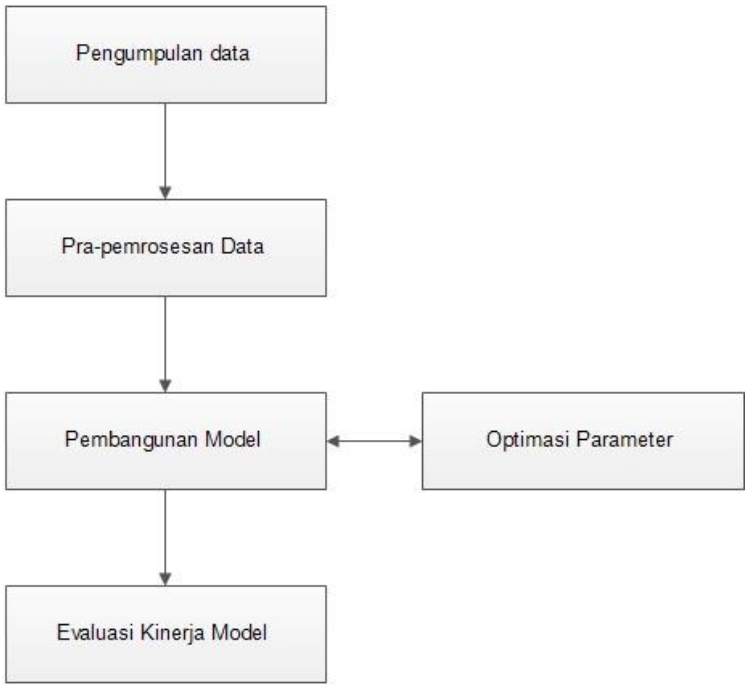
				Neighbor (*kNN*), dan *Logistic Regression.*	klasifikasi terhadap 2 kelas *output.*
3	Vandewiele et al., (2018)	"*A decision support system to follow up and diagnose primary headache patients using semantically enriched data*"	*Migbase dataset.*	*Neural Network, Support Vector Machine, Logistic Regression, kNN, CART, C4.5, GENESIM, Random Forest,* dan *eXtreme Gradient Boosting*	Beberapa buah model klasifikasi dengan nilai akurasi teritinggi oleh *GENESIM* diikuti oleh *C4.5* dan *Random Forest* dengan kisaran nilai akurasi dari 98,11% sampai 98,35%. Model yang dihasilkan dapat melakukan klasifikasi 3 jenis sakit kepala.

BAB III
METODOLOGI PENELITIAN

Bab ini menyajikan tentang garis besar penyelesaian masalah mengenai klasifikasi jenis sakit kepala menggunakan alogritma *Random Forest*, penjelasan singkat dari langkah-langkah yang akan dilakukan dan perhitungan manual dari algoritma yang akan digunakan. Terdapat beberapa proses yang akan dilakukan untuk menyelesaikan permasalahan ini yaitu garis besar penyelesaian masalah, pengumpulan data, pra-pemrosesan data, pembangunan model, dan evaluasi model.

3.1. Garis Besar Penyelesaian Masalah

Klasifikasi jenis sakit kepala menggunakan algoritma *Random Forest* dilakukan dengan beberapa tahapan. Tahapan tersebut digambarkan pada diagram garis yang dapat dilihat pada Gambar 3.1.



Gambar 3.1. Diagram Garis Besar Penyelesaian Masalah

3.2. Pengumpulan Data

Dataset yang digunakan pada penelitian ini bersumber dari *Migbase dataset* yang bisa diunduh lewat http://www.migbase.com/migbase_dataset.xls. Jenis data dari *dataset* yang digunakan terdiri dari 850 data dengan 39 fitur. *Migbase dataset* terdiri dari 3 label kelas yaitu *migrain*, *tension*, dan *cluster* dengan proporsi 71,73%, 21,67%, dan 6,60% secara

berturut-turut. Data pada fitur setiap fitur memiliki jenis yang berbeda-beda. Fitur dan jenis data untuk setiap fitur dapat dilihat pada Tabel 3.1 (Vandewiele et al., 2018).

Tabel 3.1. Jenis Data Pada Setiap Fitur

No.	Fitur	Jenis Data
1	*headache_days*	- <1 - 1 – 14 - 7 – 365 - *None*
2	*durationGroup*	- A: 0 – 4 detik - B: 5 – 199 detik - C: 120 – 239 detik - D: 240 – 899 detik - E: 900 – 1799 detik - F: 1800 – 10799 detik - G: 10800 – 14399 detik - H: 14400 – 259199 detik - I: 259200 – 604799 detik - J: 604800+ detik
3	*location*	- *Unilateral* - *Orbital* - *Bilateral*
4	*severity*	- *Mild* - *Moderate* - *Severe*
5	*characterisation*	- *Pressing* - *Pulsating* - *Stabbing*
6	*nausea, vomitting, photophobia, aggravation, pericranial, conjunctival_injection, lacrimation, nasal_congestion, rhinorrhoea, eyelid_oedema, sweating, miosis, ptosis agitation, motor_weakness, speech_disturbance, visual_sympthomps, sensory_simptomps, homonymous_symptomps, dysarthria, vertigo, tinnitus, hypacusia, diplopia, ataxia, decreased_consciousness,*	- *Yes* - *No*

	nasal_visual_symptomps, paraesthesias, aura_development, headache_with_aura, hemiplegic	
7	*aura_duration*	- *None* - *Hour* - *Day*
8	*previous_attacks*	- 2 – 4 - 5 – 9 - 10 – 19 - 20+

3.3. Pra-pemrosesan Data

Tahapan setelah pengumpulan dataset adalah melakukan pra-pemrosesan data. Pada tahap ini semua data diubah menjadi numerik agar mempermudah pemrosesan data. Hasil numerik dari pra-pemrosesan data dapat dilihat pada Tabel 3.2.

Tabel 3.2. Hasil dari Pra-pemrosesan Data

No.	Fitur	Data Sebelum Pra-pemrosesan	Data Sesudah Pra-pemrosesan
1	*nausea, vomitting, photophobia, aggravation, pericranial, conjunctival_injection, lacrimation, nasal_congestion, rhinorrhoea, eyelid_oedema, sweating, miosis, ptosis agitation, motor_weakness, speech_disturbance, visual_sympthomps, sensory_simptomps, homonymous_symptomps, dysarthria, vertigo, tinnitus, hypacusia, diplopia, ataxia, decreased_consciousness, nasal_visual_symptomps, paraesthesias, aura_development, headache_with_aura, hemiplegic*	*Yes*	1
		No	0
2	*headache_days*	*None*	0
		<1	1
		1 – 14	2
		7 – 365	3
		>14	4
		>365	5
3	*durationGroup*	A	1

		B	2
		C	3
		D	4
		E	5
		F	6
		G	7
		H	8
		I	9
		J	10
4	*location*	*Unilateral*	1
		Bilateral	2
		Unilateral	3
5	*Characterisation*	*Pressing*	1
		Pulsating	2
		Stabbing	3
6	*severity*	*Moderate*	1
		Mild	2
		Severe	3
7	*aura_duration*	*None*	0
		Hour	1
		Day	2
8	*previous_attacks*	2 – 4	1
		5 – 9	2
		10 – 19	3
		20+	4

Setelah melakukan pengubahan data untuk tiap fitur, penulis melakukan *drop fitur* atau penghapusan fitur karena memiiki isi nilai yang sama di setiap data pada fiturnya. Fitur-fitur yang akan dihapus memiliki nilai yang sama pada setiap datanya di masing-masing fitur sehingga tidak berpengaruh terhadap hasil pemodelan. Contohnya pada fitur *motor_weakness* yang pada memiliki nilai "*No*" atau "0" pada setiap datanya. Fitur-fitur yang dihapus dapat dilihat pada Tabel 3.3.

Tabel 3.3. Fitur-fitur Yang Dihapus

No.	Fitur	Nilai Setiap Data
1	*motor_weakness*	0
2	*tinnitus*	0
3	*hypacusia*	0
4	*paraesthesias*	0

5	*decreased_consciousness*	0
6	*nasal_visual_symptomps*	0

Selain metode pengecekan secara manual seperti pada Tabel 3.3 dapat juga dilakukan pengecekan menggunakan *correlation matrix* terhadap label target. Untuk fitur dengan nilai korelasi lebih besar dari sama dengan -0,5 hingga lebih kecil dari 0,5 akan dilakukan *drop fitur*. Fitur dengan nilai korelasi dengan nilai yang akan di-*drop* dapat dilihat pada Tabel 3.4.

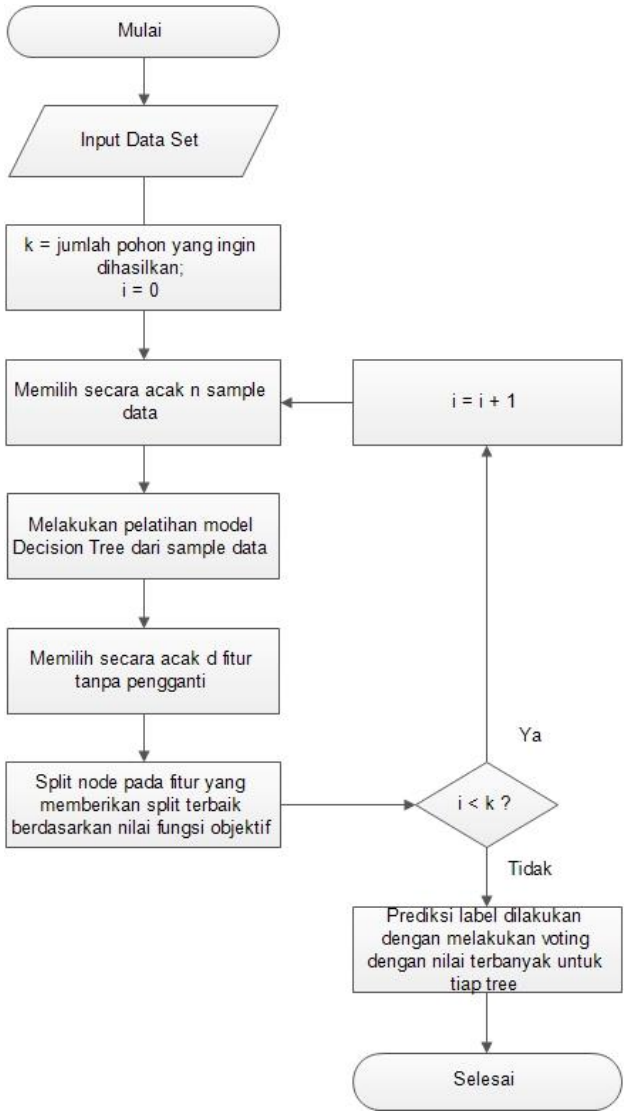
Tabel 3.4. Fitur-fitur yang Akan Di-*drop*

No.	Fitur	Nilai Setiap Data
1	*headache_days*	0,1
2	*severity*	-0,1
3	*vomitting*	-0,2
4	*pericranial*	0,2
5	*agitation*	0,1
6	*speech_disturbance*	-0,07
7	*visual_symptomps*	-0,1
8	*sensory_ symptomps*	-0,1
9	*homonymous_ symptomps*	-0,04
10	*dysanthria*	0,04
11	*vertigo*	0,04
12	*diplopia*	0,04
13	*ataxia*	0,02
14	*aura_development*	-0,1
15	*headache_with_aura*	-0,1
16	*aura_duration*	-0,1
17	*hemiplegic*	-0,2
18	*previous_attacks*	0,02
19	*ptosis*	0,4

Seteah dilakukan *drop fitur* terhadap beberapa fitur, fitur-fitur yang tersisa berjumlah 14 fitur yaitu *durationGroup, location, characterisation, nausea, photophobia, phonophobia, aggravation, conjunctival_injection, lacramation, nasal_congestion, rhinorroea, eyeid_oedema, sweeting,* dan *miosis.*

3.4. Pembagunan Model

Model klasifikasi menggunakan algoritma *Random Forest* pada penelitian ini dibangun berdasarkan dokumentasi *library Scikit-learn*. Dalam membangun model *Random Forest* terdapat beberapa langkah yang dapat dilihat pada Gambar 3.2 yang menunjukkan diagram alir algoritma *Random Forest*.



Gambar 3.2. Diagram Alir Dari Algoritma *Random Forest*

Untuk proses perhitungan manual model klasifikasi dengan algoritma *Random Forest* akan dibangun 2 pohon (*k = 2*). Kemudian dataset yang digunakan hanya berisi 10 baris data (*n = 10*) dengan menggunakan 6 fitur yang berpengaruh pada proses klasifikasi, yaitu: *durationGroup*, *location*, *severity*, *characterisation*, *nausea*, dan *photophobia* (McGeeney, 2009).

Berikut adalah penjabaran proses perhitungan manual dari klasifikasi algoritma *Random Forest*:

1. Langkah 1 – Memilih secara acak *n* sampel data dari data asli pelatihan dengan penggantian. Sampel terpilih dapat dilihat pada Tabel 3.5.

Tabel 3.5. *Dataset* untuk Membangun Model

No.	Fitur (x)						Label (y)
	Duration	*Location*	*Severity*	*Characterisation*	*Nausea*	*Photophobia*	*Class*
1	8	1	2	3	1	1	1
2	8	1	3	1	1	1	1
3	8	1	3	3	1	1	1
4	8	1	3	3	1	1	1
5	7	2	2	1	0	0	2
6	6	2	2	1	0	0	2
7	6	1	2	1	0	0	2
8	7	2	1	1	0	0	2
9	6	3	3	2	0	1	3
10	6	3	3	2	0	1	3

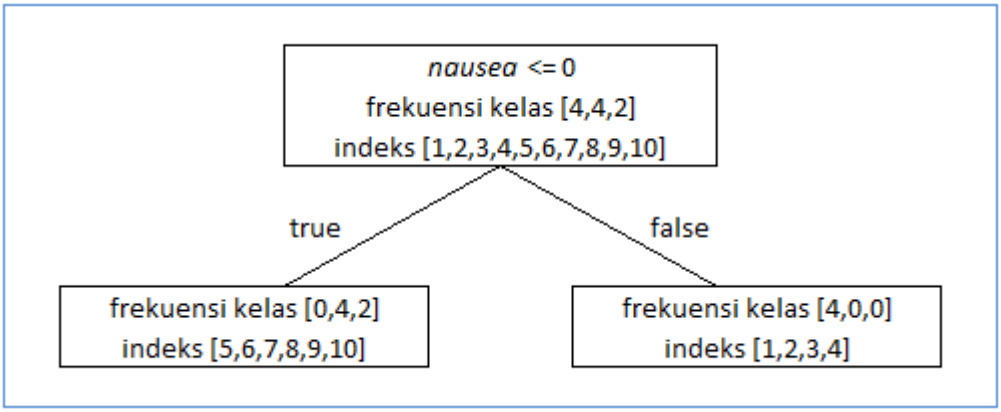
2. Langkah 2 – Melakukan pelatihan model *Decision Tree*. Untuk setiap *node*:
   a. Langkah 2.1 – Memilih fitur sebanyak *d* secara acak tanpa penggantian, sehingga diperoleh *d* fitur. Nilai *d* didapat dari akar kuadrat dari jumlah fitur berdasarkan persamaan 2.2.

$$d = \sqrt{6}$$
$$d = 2.449$$
$$d \approx 2$$

   Fitur yang diperoleh dari dataset pada Tabel 3.5 adalah fitur *nausea* dan *severity*.
   b. Langkah 2.2 – melakukan *split node* fitur yang diberikan nilai terbaik berdasarkan fungsi objektif, yakni nilai maksimal dari *information gain* fitur *d*. Pada perhitungan manual ini perhitungan hanya akan dilakukan pada *information gain child*.

Berikut adalah langkah-langkah *split node*:

1) Langkah 2.2.1 – melakukan *split node* dengan memilih fitur dan *threshold* seperti Gambar 3.3 dengan fitur *nausea* dan *threshold* = 0.



Gambar 3.3. *Split Data* Untuk Fitur *Nausea* Dengan Nilai Threshold 0

2) Langkah 2.2.2 – menghitung *information gain child* berdasarkan persamaan 2.6, 2.7 dan 2.8 untuk fitur ***nausea*** dengan nilai ***threshold* 0**. Berdasarkan Gambar 3.3 diperoleh jumlah anak sebelah kiri adalah 6 dengan komposisi kelas 1 sebanyak 0, kelas 2 sebanyak 4, dan kelas 3 sebanyak 2. Sedangkan untuk kelas sebelah kanan diperoleh anak sebanyak 4 dengan komposisi kelas 1 sebanyak 4, kelas 2 sebanyak 0, dan kelas 3 sebanyak 0. Berikut perhitungan *information gain child*-nya:

$$G_{child}(Q,\theta) = \frac{n_{left}}{N_m} H\left(Q_{left}(\theta)\right) + (\frac{n_{right}}{N_m} H\left(Q_{right}(\theta)\right)$$

$$= \frac{6}{10}(0.444) + \frac{4}{10}(0)$$

$$= 0.2667 + 0$$

$$= 0.2667$$

$$H\left(Q_{left}(\theta)\right) = H(X_m)$$

$$= \sum_k P_{mk}(1 - P_{mk})$$

$$= P_{m1}(1 - P_{m1}) + P_{m2}(1 - P_{m2}) + P_{m3}(1 - P_{m3})$$

$$= \frac{0}{6}\left(1 - \frac{0}{6}\right) + \frac{4}{6}\left(1 - \frac{4}{6}\right) + \frac{2}{6}\left(1 - \frac{2}{6}\right)$$

$$= 0 + \frac{4}{6}\left(\frac{2}{6}\right) + \frac{2}{6}\left(\frac{4}{6}\right)$$

$$= 0 + \frac{8}{36} + \frac{8}{36}$$

$$= 0.444$$

$$H\left(Q_{right}(\theta)\right) = H(X_m)$$

$$= \sum_k P_{mk}(1 - P_{mk})$$

$$= P_{m1}(1 - P_{m1}) + P_{m2}(1 - P_{m2}) + P_{m3}(1 - P_{m3})$$

$$= \frac{4}{4}\left(1 - \frac{4}{4}\right) + \frac{0}{4}\left(1 - \frac{0}{4}\right) + \frac{0}{4}\left(1 - \frac{0}{4}\right)$$

$$= \frac{4}{4}(0) + \frac{0}{4}(1) + \frac{0}{4}(1)$$

$$= 0 + 0 + 0$$

$$= 0$$

P_{mk} anak untuk sebelah kiri:

$$P_{m1} = \frac{1}{N_m} \sum_{x_i \in R_m} I\ (y_i = 1)$$

$$= \frac{1}{6}(0)$$

$$= 0$$

$$P_{m2} = \frac{1}{N_m} \sum_{x_i \in R_m} I\ (y_i = 2)$$

$$= \frac{1}{6}(4)$$

$$= \frac{4}{6}$$

$$P_{m3} = \frac{1}{N_m} \sum_{x_i \in R_m} I\ (y_i = 3)$$

$$= \frac{1}{6}(2)$$

$$= \frac{2}{6}$$

P_{mk} anak untuk sebelah kanan:

$$P_{m1} = \frac{1}{N_m} \sum_{x_i \in R_m} I\ (y_i = 1)$$

$$= \frac{1}{4}(4)$$

$$= 1$$

$$P_{m2} = \frac{1}{N_m} \sum_{x_i \in R_m} I\,(y_i = 2)$$

$$= \frac{1}{4}(0)$$

$$= 0$$

$$P_{m3} = \frac{1}{N_m} \sum_{x_i \in R_m} I\,(y_i = 3)$$

$$= \frac{1}{4}(0)$$

$$= 0$$

Berdasarkan perhitungan diatas, diketahui bahwa ***information gain child*** untuk **fitur *nausea*** dengan ***threshold* 0** adalah **0.2667**.

3) Langkah 2.2.3 – mengulangi Langkah 2.2.1 dan Langkah 2.2.2 secara terus menerus untuk tiap nilai *threshold* pada semua fitur. Hasil lengkap perhitungan yang dihasilkan dapat dilihat pada Tabel 3.6.

Tabel 3.6. Hasil Perhitungan *Information Gain Child*

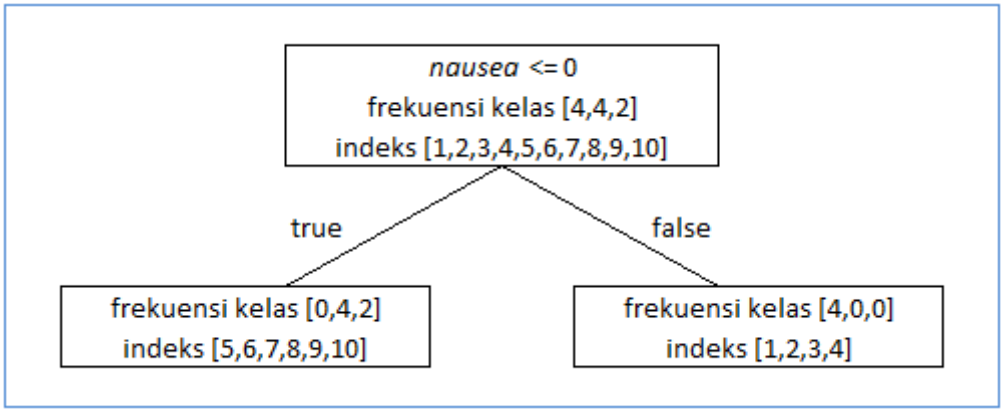
No.	Fitur	*Threshold*	*Information Gain Child*
1	*Severity*	1	0.5778
2	*Severity*	2	0.4000
3	***Nausea***	**0**	**0.2667**

4) Langkah 2.2.4 – mencari nilai terkecil dari *information gain child* berdasarkan persamaan 2.9 untuk setiap nilai *threshold* pada fitur (*d* = 2) terpilih dilangkah sebelumnya, yaitu: *nausea* dan *severity*.

Berdasarkan Tabel 3.6 diperoleh nilai *information gain child* terkecil dimiliki oleh fitur *nausea* dengan nilai *threshold 0*.

5) Langkah 2.2.5 – mencari nilai *information gain* berdasarkan persamaan 2.10 setelah nilai *information gain child* diketahui. Nilai *information gain* akan digunakan untuk mencari nilai *feature importance*.

Hasil *split data* dari fitur *nausea* dengan nilai *threshold 0* menghasilkan anak sebelah kiri sebanyak 6 dengan komposisi kelas 1 sebanyak 0, kelas 2 sebanyak 4, dan kelas 3 sebanyak 2. Untuk anak sebelah kanan dihasilkan sebanyak 4 dengan komposisi kelas 1 sebanyak 4, kelas 2 sebanyak 0, dan kelas 3 sebanyak 0. Ilustrasi hasil *split data* dapat dilihat pada Gambar 3.4.



Gambar 3.4. Hasil S*plit Data* dari Fitur *Nausea* dengan Nilai Threshold 0

Proses perhitungan nilai *information gain*:

$$IG(Q,\theta) = \frac{N_m}{N}\left(H\big(Q(\theta)\big) - G_{child}(Q,\theta)\right)$$
$$= \frac{10}{10}(0.640 - 0.2667)$$
$$= 1(0.3733)$$
$$= 0.3733$$

$$H\big(Q(\theta)\big) = H(X_m)$$
$$= \sum_k P_{mk}(1 - P_{mk})$$
$$= P_{m1}(1 - P_{m1}) + P_{m2}(1 - P_{m2}) + P_{m3}(1 - P_{m3})$$
$$= \frac{4}{10}\left(1 - \frac{4}{10}\right) + \frac{4}{10}\left(1 - \frac{4}{10}\right) + \frac{2}{10}\left(1 - \frac{2}{10}\right)$$
$$= \frac{4}{10}\left(\frac{6}{10}\right) + \frac{4}{10}\left(\frac{6}{10}\right) + \frac{2}{10}\left(\frac{8}{10}\right)$$
$$= \frac{24}{100} + \frac{24}{100} + \frac{16}{100}$$
$$= 0.640$$

P_{mk} orang tua/ *parent*:

$$P_{m1} = \frac{1}{N_m}\sum_{x_i \in R_m} I\,(y_i = 1)$$

$$= \frac{1}{10}(4)$$

$$= \frac{4}{10}$$

$$P_{m2} = \frac{1}{N_m} \sum_{x_i \in R_m} I\,(y_i = 2)$$

$$= \frac{1}{10}(4)$$

$$= \frac{4}{10}$$

$$P_{m3} = \frac{1}{N_m} \sum_{x_i \in R_m} I\,(y_i = 3)$$

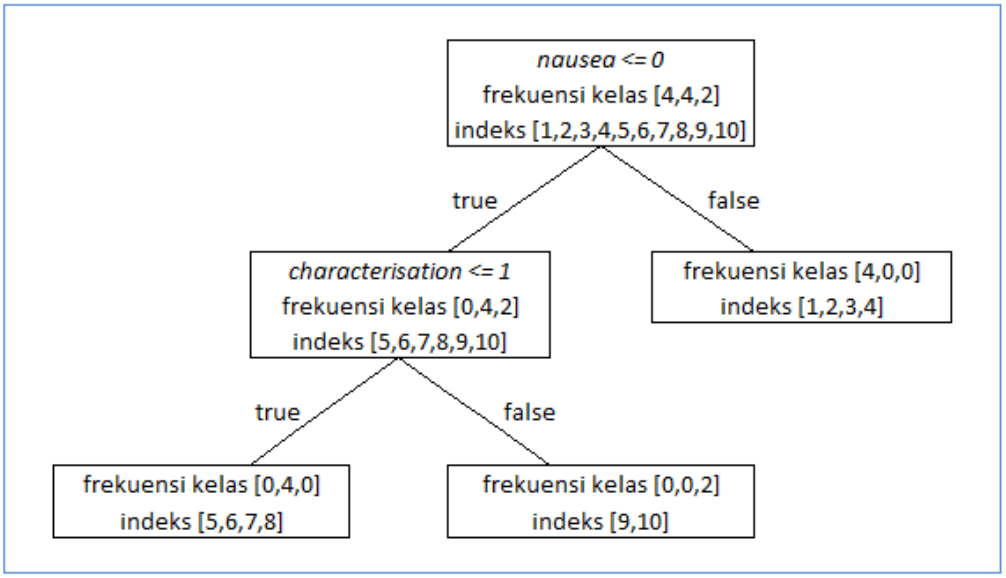
$$= \frac{1}{10}(2)$$

$$= \frac{2}{10}$$

Selanjutnya, mengulangi Langkah 2 hingga kondisi untuk berhenti melakukan *split data* terpenuhi. Diantaranya: nilai maksimal kedalaman pohon telah memenuhi nilai yang didefinisikan oleh *user* sebelumnya, nilai sampel pada *node* telah terpenuhi, nilai minimal sampel pada *node leaf* terpenuhi, data sudah tidak dapat di-*split* lagi karena nilai *impurity* = 0 dan nilai *information gain child* lebih kecil atau sama dengan nilai *information gain child* yang telah didefinisikan *user* sebelumnya.

Pada hasil perhitungan sebelumnya dapat diketahui bahwa nilai *impurity* dari *leaf* sebelah kiri adalah 0.444 dan nilai impurity dari *leaf* sebelah kanan adalah 0 sehingga harus *leaf* sebealah perlu dilakukan pengulangan Langkah 2 hingga salah satu kondisi berhenti terpenuhi.

Setelah dilakukan perhitungan pada *leaf* sebelah kiri dengan fitur pilihan *characterisation* dengan nilai *threshold 1* dan *location* dengan nilai *threshold 1* dan *2* maka dihasilkan *tree* baru yang diilustrasikan pada Gambar 3.5.

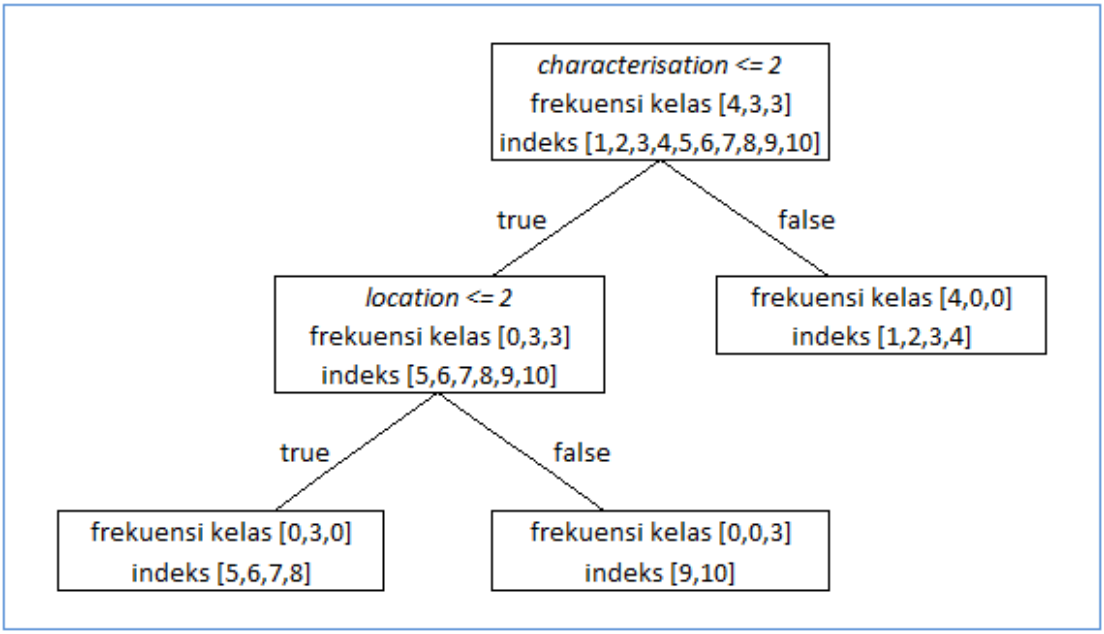


Gambar 3.5. Hasil Pohon ke-1

3. Langkah 3 – mengulangi Langkah 1 – Langkah 2 sebanyak *k* kali. Pada kasus ini nilai *k* adalah 2, sehingga terdapat 2 pohon keputusan. Ilustrasi pohon ke-1 dapat dilihat pada Gambar 3.3 dan ilustrasi pohon ke-2 dapat dilihat pada gambar 3.6. Untuk *dataset* pohon ke-2 bisa dilihat pada Tabel 3.7.

Tabel 3.7. *Dataset* untuk Membangun Model

No.	Fitur (x)						Label (y)
	Duration	*Location*	*Severity*	*Characterisation*	*Nausea*	*Photophobia*	*Class*
1	8	2	2	3	1	1	1
2	8	1	3	3	1	1	1
3	8	1	2	3	1	1	1
4	8	1	3	3	1	1	1
5	6	2	2	1	0	1	2
6	9	2	2	1	0	0	2
7	8	2	2	1	0	0	2
8	6	3	3	2	1	0	3
9	4	3	3	2	0	0	3
10	6	3	3	2	0	0	3



Gambar 3.6. Hasil Pohon ke-2

Setelah diketahui semua hasil pohon pada *Random Forest*, proses selanjutnya adalah menghitung *feature importance*. Berikut adalah penjabarannya:

1) Mencari nilai *information gain* untuk setiap node di setiap pohon pada *Random Forest* berdasarkan persamaan 2.10.

$ni_j = IG(Q, \theta)$

Pohon ke-1:

$$IG_{nausea} = \frac{N_m}{N}\left(H\left(Q(\theta)\right) - G_{child}(Q, \theta)\right)$$
$$= \frac{10}{10}(0.640 - 0.2667)$$
$$= 1(0.3733)$$
$$= 0.3733$$

$$IG_{characterisation} = \frac{N_m}{N}\left(H\left(Q(\theta)\right) - G_{child}(Q, \theta)\right)$$
$$= \frac{6}{6}(0.4444 - 0)$$
$$= 1(0.4444)$$
$$= 0.4444$$

Pohon ke-2:

$$IG_{characterisation} = \frac{N_m}{N}\left(H\left(Q(\theta)\right) - G_{child}(Q,\theta)\right)$$
$$= \frac{10}{10}(0.6600 - 0.3000)$$
$$= 1(0.3600)$$
$$= 0.3600$$

$$IG_{location} = \frac{N_m}{N}\left(H\left(Q(\theta)\right) - G_{child}(Q,\theta)\right)$$
$$= \frac{6}{6}(0.5000 - 0)$$
$$= 1(0.5000)$$
$$= 0.5000$$

2) Mencari nilai *importance* untuk setiap pohon pada *Random Forest* berdasarkan persamaan 2.11.

$$FI_i = \frac{\sum_j IG_j}{\sum_k IG_k}$$

Pohon ke-1:

$$FI_{nausea} = \frac{IG_{nausea}}{IG_{nausea} + IG_{characterisation}}$$
$$= \frac{0.3733}{0.3733 + 0.4444}$$
$$= \frac{0.3733}{0.8177}$$
$$= 0.4565$$

$$FI_{characterisation} = \frac{IG_{characterisation}}{IG_{nausea} + IG_{characterisation}}$$
$$= \frac{0.4444}{0.3733 + 0.4444}$$
$$= \frac{0.4444}{0.8177}$$
$$= 0.5435$$

Pohon ke-2:

$$FI_{characterisation} = \frac{IG_{characterisation}}{IG_{characterisation} + IG_{location}}$$
$$= \frac{0.3600}{0.3600 + 0.5000}$$

$$= \frac{0.3600}{0.8600}$$
$$= 0.4186$$

$$FI_{location} = \frac{IG_{location}}{IG_{location} + IG_{characterisation}}$$
$$= \frac{0.5000}{0.5000 + 0.3600}$$
$$= \frac{0.5000}{0.8600}$$
$$= 0.5814$$

3) Selanjutnya, menghitung nilai *feature importance* pada *Random Forest* berdasarkan persamaan 2.12.

$$RFFI_i = \frac{\sum_j FI_{ij}}{T}$$

$$RFFI_{nausea} = \frac{\sum FI_{i\ nausea}}{2}$$
$$= \frac{0.4565}{2}$$
$$= 0.22825$$

$$RFFI_{characterisation} = \frac{\sum FI_{i\ characterisation}}{2}$$
$$= \frac{0.5435 + 0.4186}{2}$$
$$= 0.48105$$

$$RFFI_{location} = \frac{\sum FI_{i\ location}}{2}$$
$$= \frac{0.5814}{2}$$
$$= 0.2907$$

Hasil akhir perhitungan nilai *feature importance* pada *Random Forest* diperoleh 3 fitur terpenting, yaitu: ***characterisation*** dengan nilai ***feature importance 0.48105***, ***location*** dengan nilai ***feature importance 0.2907***, dan ***nausea*** dengan nilai ***feature importance 0.22825***.

4. Langkah 4 – Prediksi untuk menetapkan label kelas diambil dari suara terbanyak dari masing-masing pohon pada *Random Forest*.

3.5. Evaluasi Kinerja Model

Ditahap ini model yang telah dibangun pada tahap sebelumnya akan dievaluasi nilai akurasi. Model akan dievaluasi menggunakan 10 data dengan isi 4 kelas *migrain*, 3 kelas *tension*, dan 3 kelas *cluster* yang hasil prediksinya dapat dilihat pada Tabel 3.8.

Tabel 3.8. Data Label Hasil Uji

Data	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Kelas Asli Data	1	3	2	1	2	3	2	1	3	1
Kelas Hasil Prediksi	1	3	2	1	2	3	2	1	3	1

Setelah diperoleh hasil prediksi, kemudian dibuat *confusion matrix* berdasarkan data tersebut yang dapat dilihat pada Tabel 3.9.

Tabel 3.9.1 Hasil *Confusion Matrix*

		Actual Class		
		Migrain	*Tension*	*Cluster*
Predicted Class	*Migrain*	4	0	0
	Tension	0	3	0
	Cluster	0	0	3

Berdasarkan hasil *confusion matrix* pada Tabel 3.9 diperoleh nilai *TTP* untuk semua (*predicted class* yang menghasilkan prediksi yang sesuai dengan *actual class*) dengan menggunakan persamaan 2.16:

$$TTP_{all} = \sum_{j=1}^{n} x_{jj}$$

$$TTP_{all} = 10$$

Kemudian untuk nilai dari *TFP* menggunakan persamaan 2.14:

$$TFP = \sum_{i}^{n} FP_i$$

$$TFP_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ji}$$

$$TFP = 0$$

Untuk nilai *TFN* diperoleh dengan persamaan 2.13:

$$TFN = \sum_{i}^{n} FN_i$$

$$TFN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij}$$

$$TFN = 0$$

Setelah diperoleh nilai-nilai diatas selanjutnya dicari nilai *precision*, *recall*, dan akurasinya berdasarkan persamaan 2.17. Berikut adalah nilai-nilai yang diperoleh:

$$akurasi = \frac{TTP}{\sum x}$$

$$akurasi = \frac{10}{10}$$

$$akurasi = 1$$

$$akurasi = 1 \times 100 = 100\%$$

Setelah dilakukan perhitungan terhadap akurasi terhadap data uji pada Tabel 3.7 diperoleh hasil dengan nilai akurasi = *100%*.

3.6. Optimasi Parameter

Untuk mendapatkan hasil kinerja terbaik dari model *Random Forest* perlu dilakukan beberapa pengaturan parameter secara manual. Pengaturan jumlah pohon atau *n_estimators* secara manual dapat mengurangi *error rate* pada kinerja dari sebuah model *Random Forest* (Probst & Boulesteix, 2018). *Underfitting* dapat terjadi jika jumlah pohon yang digunakan

terlalu sedikit. Agar tidak terjadi *underfitting*, nilai minimal *n_estimators* yang digunakan adalah 10 dengan nilai *max_depth* = 4 atau sebaliknya. Nilai-nilai pada parameter tersebut akan diatur dengan nilai minimal yaitu *n_estimators* = 10 dan *max_depth* = 4 dan akan ditingkatkan secara bertahap. Untuk *n__estimators* nilai yang akan digunakan adalah 10, 20, 50, dan 100. Sedangkan untuk *max_depth* nilai yang akan digunakan adalah 4, 5, dan *none* (Kertesz, 2016).

Untuk parameter lain yang akan diatur secara manual adalah parameter *max_features* dengan 2 pilihan yaitu *n_features* atau 6 fitur. *N_features* menggunakan semua fitur yang ada pada *dataset* yang akan digunakan yaitu berjumlah 14. Sedangkan 6 fitur merupakan fitur-fitur yang digunakan untuk melakukan klasifikasi sakit primer pada penelitian McGeeney (2009). Enam fitur tersebut adalah *durationGroup*, *location*, *severity*, *characterisation*, *nausea*, dan *photophobia*.

BAB IV
HASIL DAN PEMBAHASAN

Bab ini menyajikan informasi tentang lingkungan perangkat dan perangkat yang digunakan untuk penelitian, menyajikan analisis hasil eksperimen dalam penentuan *hyper-parameter* terbaik dalam *Random Forest* untuk melakukan klasifikasi jenis sakit kepala.

4.1. Lingkungan dan Perangkat yang Digunakan untuk Penelitian

Penelitian ini dilakukan di lingkungan *Jupyter Notebook* dengan bahasa pemrograman *Pyhton* 3.7.7 dan menggunakan perangkat keras berupa satu buah laptop dengan spesifikasi sebagai berikut:

1. Merek : Asus X455YA
2. Prosesor : AMD A8-7410 with AMD Radeon R5 Graphics (@2,20 GHz)
3. RAM : 8GB DDR3 (@1.333 MHz)
4. Hard Drive : 240GB SSD
5. Sistem Operasi : Windows 10 Education

4.2. Skenario Penentuan *Hyper-Parameter* Terbaik Untuk *Random Forest*

Skenario pelatihan dan pengujian model *Random Forest* yang dilakukan untuk mencari *hyper-parameter* terbaik. Skenario pelatihan dan pengujian dilakukan di *Jupyter Notebook* dengan menggunakan 850 *dataset* dari *migbase dataset* dengan penilaian kinerja model menggunakan *cross-validation*.

Pada eksperimen untuk menentukan *hyper-parameter* terbaik untuk *Random Forest* ini dihasilkan 24 model *Random Forest.* Model-model tersebut dilatih menggunakan kombinasi *hyper-parameter n_estimators, max_features,* dan *max_depth.* Untuk nilai-nilai *hyper-parameter* tersebut akan diatur manual. Penentuan manual *n_estimator* dengan nilai 10, 20, 50, 100. Penentuan manual *max_features* dengan nilai 6 dan *n_features*. Penentuan manual *max_depth* dengan nilai 4, 5, dan *None*. Pengukuran skor yang digunakan untuk mengevaluasi kinerja model yaitu akurasi dari perhitungan 5-*fold cross-validation*. Skor hasil pengujian model dapat dilihat pada Tabel 4.1.

Tabel 4.1. Skor Hasil Pengujian Model Menggunakan Akurasi *Cross-validation*

No.	*N_estimator*	*Max_features*	*Max_depth*	Akurasi
1	10	6	4	98,24%
2			5	98,09%
3			*None*	98,38%
4		14	4	98,97%
5			5	98,68%
6			*None*	98,53%
7	20	6	4	98,68%
8			5	98,53%
9			*None*	98,38%
10		14	4	**99,12%**
11			5	98,24%
12			*None*	98,68%
13	50	6	4	98,24%
14			5	98,38%
15			*None*	98,38%
16		14	4	98,82%
17			5	98,68%
18			*None*	98,82%
19	100	6	4	98,53%
20			5	98,24%
21			*None*	98,53%
22		14	4	98,82%
23			5	98,68%
24			*None*	98,68%

Berdasarkan skor hasil pengujian model *Random Forest* dengan hasil akurasi tertinggi yaitu 99,12% pada beberapa kombinasi parameter. Untuk hasil klasifikasi pada tiap kelas terhadap 20% data secara acak dari *dataset* dapat dilihat pada Tabel 4.2 tentang *confussion matrix* dari salah satu model *Random Forest.*

Tabel 4.2. *Confussion Matrix* dari Salah Satu Model *Random Forest*

		Actual Class		
		Migrain	*Tension*	*Cluster*
Predicted Class	*Migrain*	122	0	1
	Tension	1	32	0
	Cluster	0	0	13

Pada *confussion matrix* pada Tabel 4.x dapat diperoleh akurasi untuk masing-masing kelas. Untuk kelas *migrain* dihasilkan akurasi sebesar 99,19%, kelas *tension* dengan akurasi sebesar 100%, dan terakhir kelas *cluster* sebesar 92,86%.

Penjelasan tentang pengaruh nilai-nilai pada tiap parameter tersebut akan dijelaskan pada subbab selanjutnya.

4.3. Pengaruh *N*_estimators Terhadap Kinerja Model

N_estimators digunakan untuk menentukan jumah pohon yang terdapat pada model *Random Forest*. Semakin banyak jumlah pohon dalam model maka semakin lama waktu yang dibutuhkan untuk melakukan pelatihan model. Jumlah pohon yang digunakan pada sebuah model *Random Forest* akan berpengaruh terhadap kinerja model. Untuk meningkatkan kinerja model diperlukan jumlah pohon yang banyak. Namun pada titik tertentu kinerja model akan mencapai batas maksimal. Penentuan parameter *n_estimators* ini bertujuan untuk memperoleh kinerja model terbaik dengan efisiensi yang baik (Coeurjolly & Leclercq-Samson, 2018).

Dari Tabel 4.1 dapat dihitung nilai rata-rata akurasi tertinggi diperoleh dari nilai *n_estimators* = 20 dengan rata-rata akurasi = 98,61%, diikuti *n_estimators* = 100 dengan rata-rata akurasi = 98,56%, kemudian *n_estimators* = 10 dengan rata-rata akurasi = 98,48%, dan terakhir *n_estimators* = 50 dengan rata-rata akurasi = 98,5%. Sehingga nilai *n_estimators* terbaik untuk model *Random Forest* ini adalah 20.

4.4. Pengaruh *Max_features* Terhadap Kinerja Model

Max_features digunakan untuk menentukan jumlah maksimal fitur yang digunakan. Pada penelitian ini nilai *max_features* yang digunakan adalah 6 dan *n_features*. Nilai dari *n_features* sendiri ada nilai total dari semua fitur yang terdapat pada *dataset* yang digunakan. Pada penelitian ini nilai dari *n_features* adalah 14. Untuk nilai *n_features = 6*, fitur-fiturnya dipilih secara manual dan fitur-fitur tersebut adalah *durationGroup, location, severity, characterisation, nausea,* dan *photophobia*. Fitur-fitur tersebut merupakan fitur-fitur yang digunakan untuk mendiagnosis jenis sakit kepala (McGeeney, 2009).

Dari Tabel 4.1 dapat dihitung nilai rata-rata akurasi untuk tiap nilai *n_features. N_features* dengan nilai 6 memiliki nilai rata-rata akurasi sebesar 98,38% dan *n_features* dengan nilai *n_features* atau 14 memiliki nilai rata-rata akurasi sebesar 98,73%. *N_features* dengan nilai 14 memiliki akurasi lebih tinggi karena memiliki lebih banyak fitur yang memiliki korelasi terhadap hasil klasifikasi sedangkan *n_features* dengan nilai 6 bisa mengurangi kompleksitas dari model *Random Forest* yang dihasilkan. Namun pada penelitian ini nilai *n_features* = 6 meghasilkan model dengan kinerja sedikit lebih buruk dari nilai *n_features* = 14.

4.5. Pengaruh *Max_depth* Terhadadap Kinerja Model

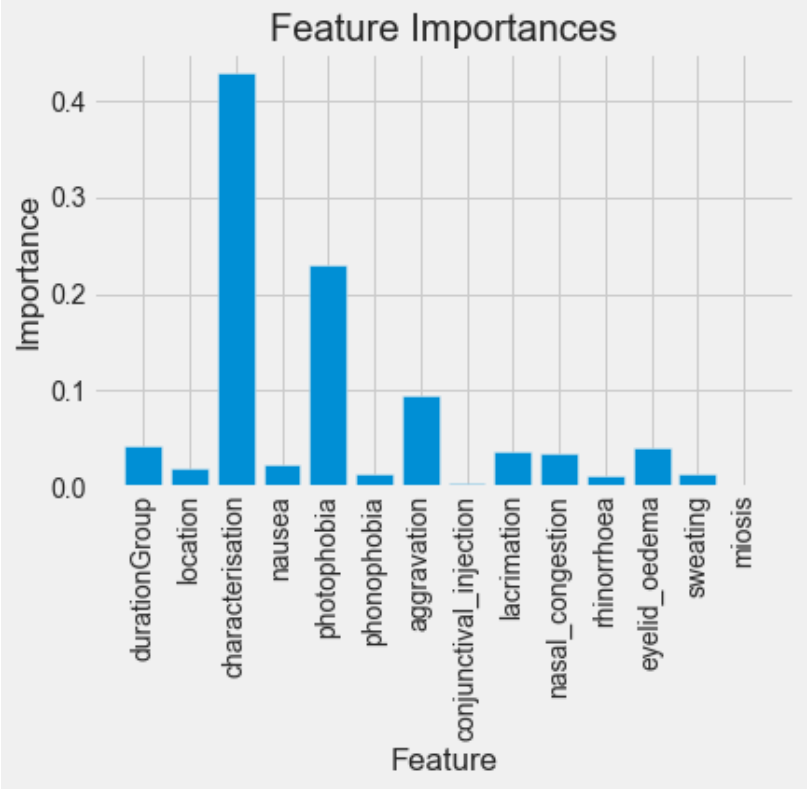
Max_depth digunakan untuk menentukan kedalaman maksimal dari sebuah pohon. Jika nilai kedalaman maksimal dari sebuah pohon telah tercapai maka sebuah *node* pada phon tidak dapat melakukan *split* lagi. Nilai *max_depth* yang digunakan untuk mencari *hyper-parameter* terbaik adalah 4, 5, dan *none*. Nilai *max_depth* digunakan untuk mengurangi variasi fitur yang akan digunakan dalam pembuatan model. Semakin tinggi nilai kedalaman sebuah pohon maka semakin kompleks pohon yang dihasilkan. Melakukan penentuan nilai pada *max_depth* dapat mempengaruhi kinerja dari *Random Forest* selama *dataset* yang digunakan untuk setiap pohonnya sama (Coeurjolly & Leclercq-Samson, 2018).

Dari Tabel 4.1 dapat dihitung nilai rata-rata akurasi untuk setiap nilai *max-depth. Max_depth* dengan nilai 4 memiliki rata-rata akurasi 98,68%. Kemudian *max_depth* dengan nilai 5 memiliki rata-rata akurasi 98,44%. Terakhir *max_depth* dengan nilai *none* memiliki akurasi 98,55%. Pada penelitian ini nilai *max_depth* hanya berpengaruh sedikit saja pada

hasil kinerja dari sebuah model *Random Forest*. Hal ini terjadi karena kompleksitas dari *dataset* yang digunakan tidak terlalu tinggi.

4.6. Hasil Perhitungan Feature Importance

Salah satu hasil dari algoritma *Random Forest* adalah *features importance.* Nilai dari *feature importance* untuk masing-masing fitur dapat dilihat pada Gambar 4.1.



Gambar 4.1. Grafik Nilai *Feature Importance*

Dari hasil perhitungan pada Gambar 4.2 dapat dilihat tidak semua fitur berkontribusi dalam pembuatan model. Fitur dengan nilai *importance value* 0 tidak memiliki pengaruh terhadap model yang dihasilkan. Sebaliknya fitur dengan dengan nilai *importance value* tinggi sangat berpengaruh dalam pembuatan model. Nilai *importance value* pada Tabel 4.2 sejalan dengan isi dari *The international classification of headache disorders* yang menyatakan masing-masing jenis sakit kepala memiliki gejalanya masing-masing dan untuk lebih jelasnya dapat dilihat pada Tabel 4.3 (Olesen, 2018).

Tabel 4.3. Fitur-fitur Untuk Mengklasifikasikan Jenis Sakit Kepala (Olesen, 2018)

No.	Gejala	*Migrain*	*Tension*	*Cluster*
1	*characterisation*	*pulsating*	*pressing*	*stabbing*
2	*photophobia and phonophobia*	*yes*	*one of them*	-
3	*aggravation*	*yes*	-	-
4	*durationGroup*	*H Group*	*F – I Group*	*E – F Group*
5	*nausea and/or vomiting*	*yes*	*no*	-
6	*severity*	*moderate or severe*	*moderate or mild*	*severe*
7	*lacrimation and/or conjunctival_injection*	-	-	*yes*
8	*location*	*unilateral*	*bilateral*	*Orbital*
9	*previous_attacks*	*at least 5 attacks*	*at least 10 attacks*	*at least 5 attacks*

Dari Tabel 4.3 dapat dilihat beberapa fitur yang digunakan untuk melaukan klasifikasi jenis sakit kepala. Beberapa diantaranya terdapat pada ketiga jenis sakit kepala dan ada 3 fitur yang digunakan hanya untuk 1 jenis sakit kepala yaitu *aggravation* untuk sakit kepala jenis *migrain* dan fitur *lacrimation and/or conjunctival_injection* untuk sakit kepala jenis *cluster*. Untuk fitur *characterisation*, *severity*, dan *location* memiliki tiga jenis sakit kepala dengan nilai data tersendiri yang bisa digunakan untuk mengklasifikasikan jenis sakit kepala.

4.7. Perbandingan Kinerja Model

Kinerja dari model *Random Forest* yang dihasilkan adalah berupa akurasi. Dari hasil penelitian ini dihasilkan nilai akurasi tertinggi adalah 99,12%. Akurasi pada penelitian ini menghasilkan nilai akurasi tertinggi dibandingakn dengan 3 penelitian sebelumnya yang telah dibahas pada bagian *state-of-the-art*. Untuk lebih jelasnya dapat dilihat pada Tabel 4.4.

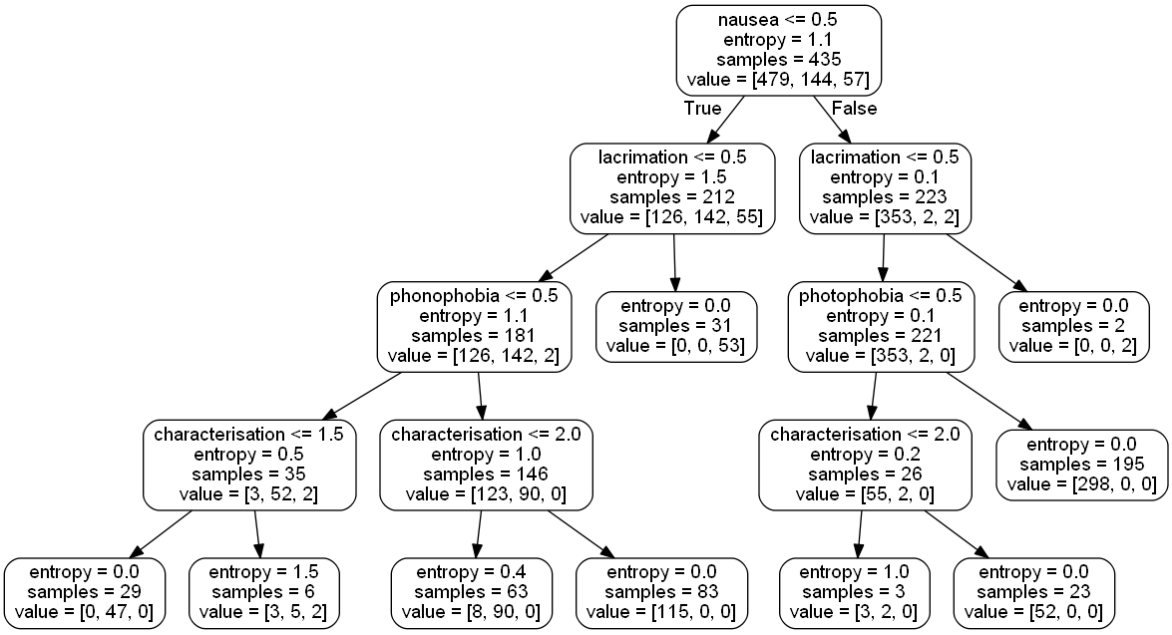
Tabel 4.4. Perbandingan Hasil Akurasi

No.	Peneliti	Judul	Akurasi Tertinggi
1	Dhiyaussalam (2020)	"Klasifikasi Jenis Sakit Kepala Menggunakan Algoritma *Random Forest*"	99,12%
2	Vandewiele et al., (2018)	"*A decision support system to follow up and diagnose primary headache patients using semantically enriched data*"	98,35%.

3	Aljaaf et al., (2015)	"*A Systematic Comparison and Evaluation of Supervised Machine Learning Classifiers Using Headache Dataset*"	96,67%.
4	(Krawczyk et al., 2013)	"*Automatic diagnosis of primary headaches by machine learning methods*"	79,97%.

4.8. Visualisasi Hasil Model Terbaik

Hasil dari model *Random Forest* terbaik digunakan untuk melakukan klasifikasi jenis sakit kepala. Model yang digunakan adalah model dengan *n_estimators = 20, max_features = 14,* dan *max_depth = 4.* Salah satu *Decision Tree*-nya dapat dilihat pada Gambar 4.2.



Gambar 4.2. Salah Satu *Decison Tree* Dari Model *Random Forest* Terbaik

Hasil dari model tersebut dapat digunakan oleh pihak dokter untuk melakukan diagnosis jenis sakit kepala. Dalam praktik klinis dokter tidak memerlukan model untuk melakukan klasifikasi. Tetapi, pada saat model ini akan sangat diperlukan pada saat dokter tidak dapat melakukan diagnosis dengan pasti (Olesen, 2018). Model ini juga dapat digunakan oleh orang awam untuk mengetahui jenis sakit kepala yang diderita agar dapat melakukan pengobatan yang sesuai.

BAB V
PENUTUP

Bab ini menyajikan kesimpulan dari uraian yang telah dijelaskan pada bab-bab sebelumnya dan saran untuk pengembangan pada penelitian selanjutnya.

5.1. Kesimpulan

Kesimpulan yang dapat diambil dari skripsi ini diantaranya adalah:

1. Dihasilkan sebuah model *Random Forest* dengan kinerja terbaik menghasilkan nilai akurasi sebesar 99,12% dan model *Random Forest* dengan kinerja paling rendah menghasilkan nilai akurasi sebesar 98,08%.
2. Model *Random Forest* paling efisien dan menghasilkan kinerja terbaik dengan nilai ratarata akurasi tertinggi diperoleh menggunakan nilai *parameter n_estimators = 20, max_features = 14,* dan *max_depth = 4.*
3. Pada pra-pemrosesan data terdapat 6 fitur pada *Migbase Dataset* yang memiliki nilai yang sama pada setiap data pada masing-masing fitur dan terdapat 19 fitur yang memiliki nilai korelasi yang rendah sehingga tidak banyak berdampak terhadap hasil kinerja dari model *Random Forest* yang dihasilkan dan dilakukan *drop fitur* terhadapa fitur-fitur tersebut.
4. Pada model *Random Forest* terbaik dihasilkan nilai *feature importance* dengan hasil nilai *feature importance* tertinggi dihasilkan oleh fitur *characterisation* yang memiliki nilai *importance* nilai sebesar 0,43 dan sebanyak 2 fitur menghasilkan nilai *importance* terkecil dengan nilai 0.

5.2. Saran

Saran yang dapat diberikan untuk penelitian selanjutnya yaitu memperluas jenis-jenis sakit kepala yang dapat diklasifikasi sehingga tidak hanya jenis sakit kepala primer saja yang dapat diklasifikasikan. Selain itu, pembangunan model berdasarkan *rules* yang ada pada *The international classification of headache disorder* sangat disarankan karena memiliki tingkat keakuratan yang tinggi secara klinis. Saran selanjutnya diharapkan model yang dihasilkan dapat diimplementasikan kedalam perangkat *mobile.*

DAFTAR PUSTAKA

Ahmed, F. (2012). Headache disorders: differentiating and managing the common subtypes. *British Journal of Pain*, *6*(3), 124–132. https://doi.org/10.1177/2049463712459691

Aljaaf, A. J., Al-Jumeily, D., Hussain, A. J., Fergus, P., Al-Jumaily, M., & Radi, N. (2015). A systematic comparison and evaluation of supervised machine learning classifiers using headache dataset. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9227*, 101–108. https://doi.org/10.1007/978-3-319-22053-6_12

Breiman, L. (2001). Random Forest. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1017/CBO9781107415324.004

Coeurjolly, J.-F., & Leclercq-Samson, A. (2018). *TUNING PARAMETERS IN RANDOM FORESTS*. *60*(2001), 144–162.

Frank, K. (2017). *Hands-On Data Science and Python Machine Learning*. https://www.packtpub.com/mapt/book/big_data_and_business_intelligence/9781787280748

Kertesz, C. (2016). Rigidity-Based Surface Recognition for a Domestic Legged Robot. *IEEE Robotics and Automation Letters*, *1*(1), 309–315. https://doi.org/10.1109/LRA.2016.2519949

Krawczyk, B., Simić, D., Simić, S., & Woźniak, M. (2013). Automatic diagnosis of primary headaches by machine learning methods. *Central European Journal of Medicine*, *8*(2), 157–165. https://doi.org/10.2478/s11536-012-0098-5

Manliguez, C. (2016). *Generalized Confusion Matrix for Multiple Classes*. *November*, 4–6. https://doi.org/10.13140/RG.2.2.31150.51523

McGeeney, B. E. (2009). An introduction to headache classification. *Techniques in Regional Anesthesia and Pain Management*, *13*(1), 2–4. https://doi.org/10.1053/j.trap.2009.03.007

Olesen, J. (2018). Headache Classification Committee of the International Headache Society (IHS) The International Classification of Headache Disorders, 3rd edition. *Cephalalgia*,

38(1), 1–211. https://doi.org/10.1177/0333102417738202

Patra, P. S. K., Sahu, D. P., & Mandal, I. (2010). An Expert System for Diagnosis Of Human Diseases. *International Journal of Computer Applications*, *1*(13), 71–74. https://doi.org/10.5120/279-439

Pedregosa, F., Weiss, R., & Brucher, M. (2011). *Scikit-learn : Machine Learning in Python*. *12*, 2825–2830.

Probst, P., & Boulesteix, A. L. (2018). To tune or not to tune the number of trees in random forest. *Journal of Machine Learning Research*, *18*(2001), 1–8.

Raschka, S., & Mirjalili, V. (2017). *Python Machine Learning Second Edition* (2nd ed.). Packt Publishing Ltd.

Rizzoli, P., & Mullally, W. J. (2018). Headache. *American Journal of Medicine*, *131*(1), 17–24. https://doi.org/10.1016/j.amjmed.2017.09.005

Ronaghan, S. (2018). *The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark*. Medium. https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, *45*(4), 427–437. https://doi.org/10.1016/j.ipm.2009.03.002

Vandewiele, G., Backere, F. De, Lannoye, K., Berghe, M. Vanden, Janssens, O., Hoecke, S. Van, Keereman, V., Paemeleire, K., Ongenae, F., & Turck, F. De. (2018). *A decision support system to follow up and diagnose primary headache patients using semantically enriched data*. *6*, 1–15.

WHO. (2016). *Headache disorders*. WHO. https://www.who.int/en/news-room/fact-sheets/detail/headache-disorders

LAMPIRAN – LAMPIRAN