BAB III. METODOLOGI PENELITIAN

3.1. Deskripsi Sistem

Sistem pakar dirancang berbasis *web*. Bahasa pemrograman yang digunakan untuk membangun sistem pakar yaitu PHP dan Javascript. Sistem pakar ini dibuat tanpa menggunakan *library neural network*, sehingga pemrosesan data pada jaringan saraf tiruan dilakukan menggunakan pengkodingan dari *scratch*. Adapun *database* yang digunakan pada sistem pakar adalah *database* MySQL.

Dalam menjalankan fungsinya, sistem memerlukan input data berupa data hasil kuesioner *Hamilton Anxiety Rating Scale* (HARS) yang telah diisi sebelumnya oleh pasien penderita gangguan kecemasan. Proses input data ini akan dilakukan oleh admin sistem. Adapun data yang diolah ke dalam sistem berupa data jawaban dari 14 pertanyaan yang terdapat pada kuesioner HARS dan data hasil diagnosa jenis gangguan kecemasan. Tiap pertanyaan pada kuesioner HARS mewakili kelompok gejala kecemasan yang umum diderita. Pilihan jawaban yang disediakan pada tiap poin pertanyaan yaitu tidak ada, ringan, sedang, berat, dan sangat berat untuk menunjukkan tingkat keparahan dari masing-masing gejala kecemasan yang diderita. Tiap pilihan jawaban kemudian akan dikonversi ke dalam skala angka yang dimulai dari 0 (tidak ada) sampai 4 (sangat berat) supaya bisa digunakan dalam perhitungan menggunakan metode *backpropagation* untuk menentukan *output* jenis gangguan kecemasan yang diderita. *Output* jenis gangguan kecemasan yang dihasilkan pada penelitian ini antara lain adalah *general anxiety disorder, panic disorder, social anxiety disorder, specific phobia, obsessive compulsive disorder,* dan *post traumatic stress disorder.*

Setelah seluruh data yang dibutuhkan diinputkan ke dalam sistem, admin sistem dapat menginput parameter pelatihan yang terdiri dari *learning rate*, *momentum*, *target error*, *epoch*, jumlah *neuron hidden layer*, dan perbandingan data latih dan data uji yang digunakan pada sistem untuk melakukan proses perhitungan menggunakan *backpropagation*. Keseluruhan data selanjutnya akan dibagi sesuai perbandingan data latih dan data uji yang telah diinputkan admin sistem. Data latih akan digunakan pada proses pelatihan menggunakan

backpropagation, sedangkan data uji akan digunakan pada proses pengujian menggunakan *backpropagation.* Proses pelatihan diperlukan sebagai pembelajaran bagi jaringan syaraf tiruan untuk mengenali pola input yang dimasukkan pada sistem sehingga sistem dapat menghasilkan *output* jenis gangguan kecemasan dengan akurat. Sementara proses pengujian dilakukan untuk menguji akurasi sistem dalam mendiagnosa jenis gangguan kecemasan menggunakan metode *backpropagation*.

Setelah keseluruhan data pada sistem dibagi menjadi data latih dan data uji, sistem selanjutnya akan melakukan inisialisasi bobot dan bias secara acak untuk perhitungan menggunakan metode *backpropagation*. Proses pelatihan pada sistem akan dilakukan melalui tahap perambatan maju, tahap perambatan balik, dan tahap perhitungan perubahan bobot. Pelatihan dilakukan sesuai jumlah iterasi (*epoch*) atau target *error* yang telah diinputkan. Bila jumlah iterasi atau target *error* sudah mencapai input yang diberikan, maka proses pelatihan akan dihentikan. Bobot akhir dari proses pelatihan kemudian disimpan dalam *database* untuk selanjutnya digunakan pada tahap pengujian. Pada proses pengujian, data pengujian akan melalui tahap perambatan maju menggunakan bobot yang dihasilkan dari tahap pelatihan. Setelah dilakukan tahap pengujian, maka sistem akan menampilkan hasil klasifikasi atau diagnosa gangguan kecemasan pada data uji.

Selain digunakan pada proses pengujian, bobot akhir dari proses pelatihan juga dapat digunakan pada tes individu atau *self-assessment* yang dilakukan oleh pasien gangguan kecemasan untuk mendiagnosa jenis gangguan kecemasan yang diderita. Sistem akan menampilkan 14 pertanyaan dari kuesioner HARS kepada pasien dan pasien menjawab seluruh pertanyaan yang ditampilkan pada sistem. Selanjutnya sistem mengkonversi jawaban pasien ke dalam bentuk skala angka. Bila skor jawaban dari kuesioner HARS yang diinputkan ke dalam sistem bernilai kurang dari atau sama dengan 6, maka hasil diagnosa yang ditampilkan sistem adalah normal/sehat. Namun jika skor jawaban kuesioner HARS yang diinputkan memiliki nilai lebih dari 6, maka sistem akan melakukan perhitungan berdasarkan jawaban kuesioner HARS menggunakan metode *backpropagation.* Sistem kemudian menampilkan *output* hasil diagnosa gangguan kecemasan yang diderita pasien berdasarkan jawaban yang telah diinputkan pasien.

3.2. Metode Pengumpulan Data

Jumlah data yang akan digunakan pada penelitian ini sejumlah 120 data. Data tersebut didapatkan dari penelitian skripsi berjudul "Identifikasi Jenis Penyakit Mental Ansietas Menggunakan Metode *Modified K-Nearest Neighbor*" yang dilakukan oleh Sakti pada tahun 2019. Pada penelitian tersebut, data yang dikumpulkan berupa gabungan dari data sekunder dan data primer yang didapatkan dari Rumah Sakit Immanuel Bandung (Sakti, 2019). Data primer diperoleh dari hasil kuesioner *Hamilton Anxiety Rating Scale* (HARS) yang disebarkan oleh pihak Rumah Sakit Immanuel Bandung kepada pasien penderita gangguan kecemasan. Sedangkan data sekunder dikumpulkan secara langsung dari hasil kuesioner HARS yang disebarkan oleh peneliti kepada pasien penderita gangguan kecemasan di Rumah Sakit Immanuel Bandung. Pengumpulan data sekunder dilakukan dengan tujuan untuk menambah jumlah data dan menyeimbangkan jumlah data dari data primer yang telah dikumpulkan sebelumnya.

Setelah semua data dari hasil kuesioner HARS terkumpul, data tersebut kemudian divalidasi kepada pakar psikologi dari Program Studi Psikologi Fakultas Ilmu Sosial dan Ilmu Politik Universitas Brawijaya. Proses validasi kepada pakar psikologi dilakukan pada penelitian ini karena kuesioner HARS umumnya digunakan untuk menentukan tingkat keparahan kecemasan yang diderita pasien. Pada kuesioner HARS, penentuan tingkat keparahan dari kecemasan yang diderita dilakukan dengan menjumlahkan seluruh skor jawaban dari masing-masing pertanyaan yang terdapat pada kuesioner HARS. Akan tetapi, kuesioner HARS juga dapat digunakan untuk menentukan jenis gangguan kecemasan yang diderita pasien gangguan kecemasan. Namun demikian, penentuan jenis gangguan kecemasan menggunakan kuesioner HARS harus dilakukan dengan bantuan pakar yang ahli dalam bidang psikologi. Sehingga untuk menentukan jenis gangguan kecemasan yang diderita berdasarkan hasil dari tiap kuesioner HARS yang telah disebarkan kepada pasien gangguan kecemasan, maka perlu dilakukan validasi terlebih dahulu kepada pakar psikologi yang telah berpengalaman di bidangnya. Pakar melakukan validasi data dengan menentukan kelompok gejala mana yang memiliki skor tertinggi dan terendah pada hasil kuesioner HARS. Korelasi antar kelompok gejala

tersebut selanjutnya dapat digunakan untuk menentukan jenis gangguan kecemasan yang diderita pasien.

Selain menggunakan data primer dan data sekunder, penelitian ini juga menggunakan data tambahan yang didapatkan dari hasil studi literatur. Studi literatur dilakukan dengan pencarian informasi atau pengetahuan dari sumber-sumber ilmiah seperti buku dan jurnal penelitian terkait. Data hasil studi literatur akan digunakan sebagai data pendukung yang diharapkan dapat membantu meningkatkan performa dari sistem pakar sehingga sistem pakar mampu menghasilkan tingkat akurasi yang baik dalam mendiagnosa jenis gangguan kecemasan yang diderita pasien.

3.3. Metode Pengolahan Data

Setelah melakukan proses pengumpulan data, selanjutnya data akan diolah supaya bisa digunakan sebagai inputan pada sistem. Pengolahan data dilakukan dengan mentransformasi nilai data jawaban yang terdapat pada kuesioner HARS. Kuesioner HARS terdiri dari 14 pertanyaan, masing-masing pertanyaan mewakili kelompok gejala kecemasan yang umum diderita pasien penderita kecemasan. Tiap pertanyaan terdiri dari 5 pilihan jawaban yang mewakili tingkat keparahan dari masing-masing gejala kecemasan yang diderita. Pilihan jawaban yang disediakan pada tiap pertanyaan pada kuesioner HARS antara lain yaitu tidak ada, ringan, sedang, berat, dan sangat berat. Masing-masing poin jawaban akan dikonversi pada skala 0 (tidak ada) sampai 4 (sangat berat), sehingga nilai pada data tersebut dapat digunakan dalam perhitungan menggunakan *backpropagation.* Selain data jawaban kuesioner HARS, proses transformasi juga dilakukan pada data diagnosa gangguan kecemasan yang dihasilkan dari inputan kuesioner HARS. Data hasil diagnosa gangguan kecemasan akan ditransformasi menjadi kelas-kelas yang terdiri dari kelas 1 sampai kelas 6.

Setelah melalui proses transformasi data, maka selanjutnya keseluruhan hasil rekapitulasi data hasil kuesioner HARS akan dibagi menjadi data latih dan data uji. Data latih adalah data untuk melatih sistem pakar yang telah dibangun menggunakan metode *backpropagation*. Sedangkan data uji digunakan untuk

melakukan proses pengujian pada sistem pakar yang telah melalui proses pelatihan menggunakan data latih.

Pada saat pembagian data, data akan diseleksi untuk menentukan data mana saja yang akan dijadikan data latih dan data mana saja yang akan dijadikan data uji berdasarkan perbandingan *split test* yang telah ditentukan. Pembagian data dalam penelitian ini menggunakan perbandingan 90:10, 80:20, dan 70:30. Setelah melalui proses pembagian data, selanjutnya data akan disimpan dalam *database*. Tabel 3.1 berikut menunjukkan jumlah pembagian data latih dan data uji sesuai perbandingan *split test* yang telah ditentukan.

Tabel 3.1 Jumlah Pembagian Data Latih dan Data Uji

	90:10	**80:20**	**70:30**
Data Latih	108	96	84
Data Uji	12	24	36

3.4. Metode Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak yang digunakan dalam penelitian ini adalah metode *waterfall*. Metode *waterfall* merupakan metode di mana pengerjaan proyek dilakukan secara bertahap sehingga metode ini bersifat sistematis dan berurutan. Dalam metode ini, jika tahap pertama belum selesai dilakukan, maka tidak bisa langsung mengerjakan tahap selanjutnya. Terdapat 5 tahap yang dilakukan dalam pengembangan perangkat lunak menggunakan metode *waterfall*, yaitu tahap *requirement definition, system and software design, implementation and unit testing, integration and system testing,* dan *operation and maintenance*.

1. *Requirement Definition*

   Pada tahap ini, terdapat proses pengumpulan kebutuhan secara lengkap untuk kemudian dianalisis. Namun sebelum dilakukan proses pengumpulan kebutuhan, perlu terlebih dahulu dilakukan proses pengumpulan data. Dari data yang dikumpulkan, maka dapat ditentukan apa saja kebutuhan dari sistem dan selanjutnya dilakukan analisis kebutuhan sistem baik kebutuhan fungsional maupun kebutuhan non-fungsional. Adapun pengambilan data pada penelitian ini dilakukan melalui observasi, studi literatur terhadap sejumlah penelitian

terkait, dan mengambil data hasil jawaban kuesioner HARS dari penelitian yang dilakukan oleh Sakti pada tahun 2019.

2. *System and Software Design*

Selanjutnya pada tahap ini terdapat proses merancang tampilan antar muka untuk menghubungkan antara pengguna aplikasi dengan aplikasi atau sistem pakar yang telah dibangun sehingga pengguna dapat melakukan interaksi pada aplikasi dengan mudah. Proses perancangan aplikasi membantu untuk menentukan dan mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan.

3. *Implementation and Unit Testing*

Tahap implementasi merupakan tahapan di mana terdapat proses pengkodean piranti lunak berupa penulisan bahasa pemrograman sesuai dengan analisa dan perancangan yang telah dilakukan. Pengkodean piranti lunak diperlukan agar piranti lunak dapat dijalankan mesin. Pada penelitian ini, sistem dibangun berbasis *website* menggunakan bahasa pemrograman PHP dan Javascript serta menggunakan *database* MySQL.

Setelah melakukan implementasi sistem, maka sistem akan diuji per unit, artinya pengujian dilakukan pada masing-masing unit yang terdapat pada sistem. Pengujian ini diperlukan untuk menemukan kesalahan pada tiap unit sistem sehingga kesalahan tersebut kemudian dapat diperbaiki sebelum keseluruhan unit tersebut digabung menjadi satu kesatuan yang utuh. Pengujian unit ini merupakan verifikasi bahwa tiap unit sistem telah memenuhi spesifikasinya.

4. *Integration and System Testing*

Pada tahap ini, dilakukan proses integrasi atau penggabungan tiap unit sistem untuk kemudian diuji sebagai sebuah sistem yang lengkap untuk memastikan apakah sudah sesuai dengan kebutuhan perangkat lunak serta mengecek kesalahan yang terdapat pada perangkat lunak. Pengujian sistem dilakukan pada fungsionalitas sistem dan akurasi yang dihasilkan dari perhitungan menggunakan metode *backpropagation* pada sistem. Apabila hasil pengujian menunjukkan bahwa masih terdapat kesalahan pada sistem, maka

perlu dilakukan perbaikan pada sistem hingga sistem memenuhi spesifikasi yang diinginkan. Adapun pengujian fungsional pada penelitian ini menggunakan *black box testing,* sedangkan pengujian akurasi dilakukan menggunakan *confusion matrix.*

5. *Operation and Maintenance*

Tahapan ini merupakan tahap akhir dalam metode *waterfall*. Tahapan ini adalah tahap di mana perangkat lunak dioperasikan dan terdapat pemeliharaan terhadap perangkat lunak tersebut. Pemeliharaan termasuk dalam koreksi berbagai *error* yang tidak ditemukan pada pengujian di tahap-tahap sebelumnya, perbaikan sistem, dan pengembangan pelayanan sistem. Pada proses pemeliharaan, tidak menutup kemungkinan akan terdapat *upgrade* pada sistem karena sistem dapat mengalami perubahan untuk menyesuaikan dengan kebutuhan pengguna.

3.5. Metode Pengujian

Pengujian bertujuan untuk mengetahui apakah sistem telah berjalan sesuai harapan. Pada penelitian ini, metode pengujian yang digunakan untuk menguji sistem secara keseluruhan adalah metode *black box testing* dan *confusion matrix.* Metode *black box testing* digunakan untuk menguji fungsional sistem, sedangkan *confusion matrix* digunakan untuk menguji penerapan algoritma pada sistem.

Pada pengujian fungsional sistem, penerapan *black box testing* dilakukan dengan mengawasi hasil eksekusi dan memeriksa fungsional dari perangkat lunak tanpa mengetahui struktur internal kode atau program. *Black box testing* umumnya hanya mengacu pada detail aplikasi seperti tampilan aplikasi, fungsi–fungsi yang ada pada aplikasi, dan kesesuaian alur fungsi dengan bisnis proses yang diinginkan oleh pengguna. Fitur-fitur sistem yang diuji menggunakan metode *black box testing* dibagi menjadi 2, yaitu fitur dari sisi admin dan fitur dari sisi pasien. Fitur-fitur yang diuji dari sisi admin terdapat dalam tabel 3.2.

Tabel 3.2 Uji Coba Fitur Admin

No	Fitur	Keterangan Hasil
1	Melakukan registrasi	Valid/Tidak Valid
2	Melakukan *login*	Valid/Tidak Valid
3	Melakukan manajemen data latih	Valid/Tidak Valid
4	Melakukan manajemen data uji	Valid/Tidak Valid
5	Melakukan manajemen data *user*	Valid/Tidak Valid
6	Menjalankan proses *backpropagation* pada sistem	Valid/Tidak Valid
7	Melakukan *logout*	Valid/Tidak Valid

Fitur yang diuji dari sisi pasien terdapat dalam tabel 3.3 berikut.

Tabel 3.3 Uji Coba Fitur Pasien

No	Fitur	Keterangan Hasil Uji
1	Melakukan registrasi	Valid/Tidak Valid
2	Melakukan *login*	Valid/Tidak Valid
3	Melakukan *self-assessment* untuk mendiagnosa jenis gangguan kecemasan	Valid/Tidak Valid
4	Melakukan *logout*	Valid/Tidak Valid

Adapun dalam pengujian akurasi, *confusion matrix* digunakan untuk mengukur performa dari model klasifikasi pada *machine learning* di mana *output* yang dihasilkan dapat berupa 2 kelas (*binary classification*) atau lebih dari 2 kelas (*multi classification*). *Confusion matrix* umumnya ditampilkan dalam bentuk tabel untuk menyatakan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan. Persamaan berikut merupakan rumus untuk menghitung akurasi berdasarkan *confusion matrix*.

$$Akurasi = \frac{TP + TN}{TP + FN + FP + TN} \times 100\% \quad (3.1)$$

Keterangan:

TP = *True Positive*, adalah jumlah data pada suatu kelas yang diprediksi benar dan pada kondisi aktual, data tersebut tergolong benar.

TN = *True Negative*, yaitu jumlah data pada suatu kelas yang diprediksi salah dan pada kondisi aktual, data tersebut memang salah.

FP = *False Positive*, yaitu jumlah data pada suatu kelas yang diprediksi benar namun pada kondisi aktual, data tersebut seharusnya salah.
FN = *False Negative*, yaitu jumlah data pada suatu kelas yang diprediksi salah namun pada kondisi aktual, data tersebut seharusnya benar.

Selain digunakan untuk menghitung akurasi, *confusion matrix* juga dapat digunakan dalam perhitungan *precision, recall,* dan *F-Measure*. *Precision* adalah pembagian dari jumlah total data positif yang diklasifikasikan bernilai benar dengan jumlah total data positif yang diprediksi. Berikut adalah persamaan untuk menghitung *precision*.

$$Presisi = \frac{TP}{TP + FP} \times 100\% \quad (3.2)$$

Sementara *recall* adalah perbandingan jumlah total data positif yang diklasifikasikan bernilai benar dengan jumlah total data positif. Berikut adalah persamaan untuk menghitung *recall*.

$$Recall = \frac{TP}{TP + FN} \times 100\% \quad (3.3)$$

F-Measure atau *F1-Score* merupakan nilai rataan harmonik dari *precision* dan *recall*. Berikut adalah persamaan untuk menghitung *F-Measure*.

$$F - Measure = \frac{2 \text{ x } Recall \text{ x } Precision}{Recall + Precision} \quad (3.4)$$

Contoh pengujian pada model klasifikasi *multi classification* dengan *confusion matrix* dapat ditunjukkan pada tabel 3.4.

Tabel 3.4 Tabel Contoh Hasil Pengujian *Confusion Matrix*

		Kelas Hasil Uji					
		1	**2**	**3**	**4**	**5**	**6**
Kelas Sebenarnya	**1**	**6**	0	0	0	0	0
	2	1	**4**	1	0	0	0
	3	0	0	**6**	0	0	0
	4	0	0	0	**6**	0	0
	5	0	0	0	0	**6**	0
	6	0	0	0	0	0	**6**

Keterangan:

= Hasil kelas benar

= Hasil kelas salah

Pada tabel 3.4, jumlah keseluruhan data adalah 36 data. Berikut ini adalah perhitungan akurasi model klasifikasi menggunakan persamaan 3.1.

$$Akurasi = \frac{6+4+6+6+6+6}{6+4+6+6+6+6+1+1} \times 100\% = \frac{34}{36} \times 100\% = 94.44\%$$

Untuk menghitung hasil *precision* pada data dari tabel 3.4, perhitungan yang digunakan terdapat pada persamaan 3.2. Berikut adalah contoh perhitungan *precision* pada tiap kelas klasifikasi menggunakan *confusion matrix.*

$$P(1) = \frac{6}{6+1} \times 100\% = \frac{6}{7} \times 100\% = 85.71\%$$

$$P(2) = \frac{4}{4+0} \times 100\% = \frac{4}{5} \times 100\% = 100\%$$

$$P(3) = \frac{6}{6+1} \times 100\% = \frac{6}{7} \times 100\% = 85.71\%$$

$$P(4) = \frac{6}{6+0} \times 100\% = \frac{6}{6} \times 100\% = 100\%$$

$$P(5) = \frac{6}{6+0} \times 100\% = \frac{6}{6} \times 100\% = 100\%$$

$$P(6) = \frac{6}{6+0} \times 100\% = \frac{6}{6} \times 100\% = 100\%$$

Untuk menghitung *precision* perhitungan pada keseluruhan kelas klasifikasi, maka jumlahkan seluruh hasil *precision* pada tiap kelas kemudian hasil penjumlahan tersebut dibagi dengan jumlah kelas. Rumus untuk menghitung *precision* perhitungan pada keseluruhan kelas terdapat pada persamaan 3.5 berikut.

$$Precision = \frac{P(1) + \ldots + P(Jumlah\ kelas)}{Jumlah\ kelas} \quad (3.5)$$

Berikut ini adalah contoh perhitungan *precision* keseluruhan kelas dari data pada tabel 3.4 menggunakan *confusion matrix.*

$$Precision = \frac{85.71\% + 100\% + 85.71\% + 100\% + 100\% + 100\%}{6}$$

$$= \frac{571.42\%}{6} = 95.24\%$$

Persamaan yang digunakan untuk menghitung *recall* terdapat pada persamaan 3.3. Berikut adalah contoh perhitungan *recall* pada tiap kelas menggunakan *confusion matrix.*

$$R(1) = \frac{6}{6+0} \times 100\% = \frac{8}{8} \times 100\% = 100\%$$

$$R(2) = \frac{4}{4+1+1} \times 100\% = \frac{4}{6} \times 100\% = 66.67\%$$

$$R(3) = \frac{6}{6+0} \times 100\% = \frac{8}{8} \times 100\% = 100\%$$

$$R(4) = \frac{6}{6+0} \times 100\% = \frac{8}{8} \times 100\% = 100\%$$

$$R(5) = \frac{6}{6+0} \times 100\% = \frac{8}{8} \times 100\% = 100\%$$

$$R(6) = \frac{6}{6+0} \times 100\% = \frac{8}{8} \times 100\% = 100\%$$

Perhitungan *recall* pada keseluruhan kelas dapat dilakukan dengan menjumlahkan seluruh hasil *recall* pada tiap kelas, selanjutnya hasil penjumlahan tersebut dibagi dengan jumlah kelas klasifikasi pada *confusion matrix.* Rumus untuk menghitung perhitungan *recall* pada keseluruhan kelas terdapat pada persamaan 3.6 berikut.

$$Recall = \frac{R(1) + \ldots + R(Jumlah\ kelas)}{Jumlah\ kelas} \quad (3.6)$$

Berikut adalah contoh perhitungan *recall* pada keseluruhan kelas klasifikasi menggunakan *confusion matrix.*

$$Recall = \frac{100\% + 66.67\% + 100\% + 100\% + 100\% + 100\%}{6}$$

$$= \frac{566.67\%}{6} = 94.45\%$$

Perhitungan *F-Measure* dilakukan menggunakan persamaan 3.4. Berikut adalah contoh perhitungan *F-Measure*.

$$F - Measure = \frac{2 \text{ x } 94.45\% \text{ x } 95.24\%}{94.45\% + 95.24\%} = \frac{17990.836\%}{189.69\%} = 94.84\%$$