BAB 4
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Tahap implementasi merupakan tahap pembuatan perangkat lunak, tahap lanjut dari tahap perancangan. Tahap yang dilakukan untuk menerjemahkan perancangan berdasarkan hasil analisis dalam bahasa yang dimengerti oleh komputer serta penerapan perangkat lunak pada keadaan sebenarnya.

4.1.1 Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan untuk membangun sistem prediksi banjir ini dapat dilihat pada tabel 4.1

Tabel 4.1 Spesifikasi Implementasi Perangkat Keras

No	Perangkat Keras	Spesifikasi
1	Prosesor	Intel Core i3
2	Memori	4096 MB
3	Monitor	Monitor 14”

4.1.2 Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan untuk membangun sistem prediksi banjir ini dapat dilihat pada tabel 4.2

Tabel 4.2 Spesifikasi Implementasi Perangkat Lunak

No	Perangkat Lunak
1	*System* operasi Windows 8.1
2	Web server XAMPP
3	Microsoft Visual Studio 2013

4.1.3 Implementasi *Database*

Pembuatan *database* dilakukan degan menggunakan DBMS *MySQL*. Implementasi *database* dalam bahasa SQL adalah sebagai berikut :

Tabel 4.3 Implementasi *Database*

No	Nama Tabel		SQL
1	Kecamatan		CREATE TABLE `kecamatan` (`id_kecamatan` int(4) NOT NULL AUTO_INCREMENT, `nama` varchar(50) DEFAULT NULL, `ch` int(3) DEFAULT NULL, `kl` int(3) DEFAULT NULL, `jt` int(3) DEFAULT NULL, `pl` int(3) DEFAULT NULL, `target_output` int(3) DEFAULT NULL, PRIMARY KEY (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
2	Nilai_belajar		CREATE TABLE `nilai_belajar` (`id` int(3) NOT NULL, `achr` decimal(10,8) DEFAULT NULL, `achs` decimal(10,8) DEFAULT NULL, `acht` decimal(10,8) DEFAULT NULL, `cchr` decimal(10,8) DEFAULT NULL, `cchs` decimal(10,8) DEFAULT NULL, `ccht` decimal(10,8) DEFAULT NULL, `aklr` decimal(10,8) DEFAULT NULL, `akls` decimal(10,8) DEFAULT NULL,

No	Nama Tabel		SQL
			`aklt` decimal(10,8) DEFAULT NULL, `cklr` decimal(10,8) DEFAULT NULL, `ckls` decimal(10,8) DEFAULT NULL, `cklt` decimal(10,8) DEFAULT NULL, `ajtr` decimal(10,8) DEFAULT NULL, `ajts` decimal(10,8) DEFAULT NULL, `ajtt` decimal(10,8) DEFAULT NULL, `cjtr` decimal(10,8) DEFAULT NULL, `cjts` decimal(10,8) DEFAULT NULL, `cjtt` decimal(10,8) DEFAULT NULL, `aplr` decimal(10,8) DEFAULT NULL, `apls` decimal(10,8) DEFAULT NULL, `aplt` decimal(10,8) DEFAULT NULL, `cplr` decimal(10,8) DEFAULT NULL, `cpls` decimal(10,8) DEFAULT NULL, `cplt` decimal(10,8) DEFAULT NULL, PRIMARY KEY (`id`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;

No	Nama Tabel		SQL
3	Temp_op_lapisan_1		CREATE TABLE `temp_op_lapisan_1` (`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int(4) DEFAULT NULL, `chr` decimal(30,8) DEFAULT NULL, `chs` decimal(30,8) DEFAULT NULL, `cht` decimal(30,8) DEFAULT NULL, `klr` decimal(30,8) DEFAULT NULL, `kls` decimal(30,8) DEFAULT NULL, `klt` decimal(30,8) DEFAULT NULL, `jtr` decimal(30,8) DEFAULT NULL, `jts` decimal(30,8) DEFAULT NULL, `jtt` decimal(30,8) DEFAULT NULL, `plr` decimal(30,8) DEFAULT NULL, `pls` decimal(30,8) DEFAULT NULL, `plt` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `temp_op_lapisan_1_ibfk_1` FOREIGN KEY (`id_kecamatan`)

No	Nama Tabel		SQL
			REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
4	Temp_op_lapisan_2		CREATE TABLE `temp_op_lapisan_2` (`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int(4) DEFAULT NULL, `w1` decimal(30,8) DEFAULT NULL, `w2` decimal(30,8) DEFAULT NULL, `w3` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `temp_op_lapisan_2_ibfk_1` FOREIGN KEY (`id_kecamatan`) REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
5	Temp_op_lapisan_3		CREATE TABLE `temp_op_lapisan_3` (`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int (4) DEFAULT NULL, `w^1` decimal(30,8) DEFAULT NULL, `w^2` decimal(30,8) DEFAULT NULL,

No	Nama Tabel		SQL
			`w^3` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `temp_op_lapisan_3_ibfk_1` FOREIGN KEY (`id_kecamatan`) REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
6	Temp_op_lapisan_4		CREATE TABLE `temp_op_lapisan_4` (`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int (4) DEFAULT NULL, `w1f1` decimal(30,8) DEFAULT NULL, `w2f2` decimal(30,8) DEFAULT NULL, `w3f3` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `temp_op_lapisan_4_ibfk_1` FOREIGN KEY (`id_kecamatan`) REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
7	Temp_op_lapisan_5		CREATE TABLE `temp_op_lapisan_5` (

No	Nama Tabel		SQL
			`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int (4) DEFAULT NULL, `wifi_tot` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `temp_op_lapisan_5_ibfk_1` FOREIGN KEY (`id_kecamatan`) REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
8	Uji		CREATE TABLE `uji` (`id` int(8) NOT NULL AUTO_INCREMENT, `nama` varchar(50) DEFAULT NULL, `ch` int(3) DEFAULT NULL, `kl` int(3) DEFAULT NULL, `jt` int(3) DEFAULT NULL, `pl` int(3) DEFAULT NULL, `target_output` int(3) DEFAULT NULL, PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=latin1;
9	Rmse		CREATE TABLE `rmse` (`id` int(3) NOT NULL, `rmse` double DEFAULT NULL, PRIMARY KEY (`id`)

No	Nama Tabel		SQL
			) ENGINE=InnoDB DEFAULT CHARSET=latin1;
10	Hasil_akhir		CREATE TABLE `hasil_akhir` (`id` int(8) NOT NULL AUTO_INCREMENT, `id_kecamatan` int (4) DEFAULT NULL, `target_output` int(3) DEFAULT NULL, `output_jaringan` decimal(30,8) DEFAULT NULL, `error` decimal(30,8) DEFAULT NULL, PRIMARY KEY (`id`), KEY `id_kecamatan` (`id_kecamatan`), CONSTRAINT `hasil_akhir_ibfk_1` FOREIGN KEY (`id_kecamatan`) REFERENCES `kecamatan` (`id_kecamatan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;

4.1.4 Implementasi Antarmuka

Berikut adalah Implementasi antarmuka untuk sistem prediksi banjir :

Tabel 4.4 Implementasi Antarmuka

Nama	Deskripsi	Nama File
Home	Digunakan untuk tampilan utama dari sistem prediksi banjir	Awal.cs
Pengolahan Data	Digunakan untuk rekapitulasi kecamatan yang sudah diinputkan, selain itu dapat juga	Tampilkecamatan.cs Tambahkecamatan.cs

Nama	Deskripsi	Nama File
	melakukan hapus kecamatan ,tambah kecamatan dan juga ubah kecamatan	
Pembelajaran ANFIS	Digunakan untuk menghitung dan melihat hasil prediksi banjir	Belajar.cs
Uji Coba	Digunakan untuk menghitung dan melihat hasil ujicoba prediksi banjir berdasarkan proses pembelajaran	Uji_coba.cs

Untuk seluruh *design* antarmuka dapat dilihat pada Lampiran A.

4.2 Pengujian Metode

Pengujian metode bertujuan untuk mengetahui performansi dari metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) yang diimplementasikan pada sistem prediksi tingkat potensi banjir. Pengujian metode ini menggunakan data uji sebanyak 15 kelurahan. Yang terbagi kedalam 5 kelurahan kurang rawan banjir, 5 kelurahan cukup rawan, dan 5 kelurahan sangat rawan banjir.

Pada metode ANFIS terdapat 2 buah variable yang dapat dirubah, yaitu banyak iterasi dan toleransi *error*. Perubahan banyak iterasi dan toleransi *error* dilakukan untuk membatasi proses pembelajaran metode ANFIS.

Akurasi dari metode ANFIS ini didapat dari banyaknya hasil prediksi sistem yang sesuai dengan target ouput dan hasil prediksi sistem yang tidak sesuai. Jumlah hasil prediksi yang akurat dan jumlah hasil prediksi yang tidak akurat dirubah menjadi nilai persen untuk memudahkan dalam mengetahui nilai akurasi dari metode ANFIS ini.

4.2.1 Skenario Pengujian 1

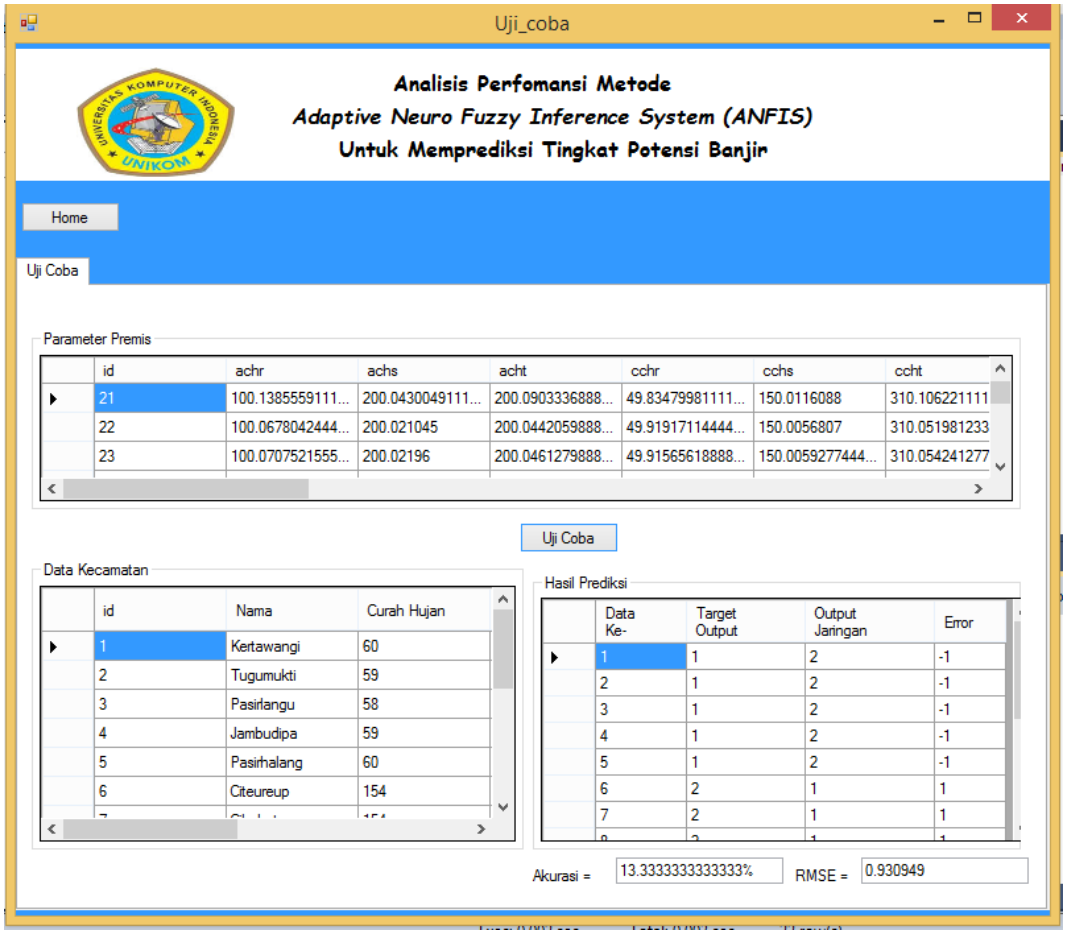
Pada pengujian 1 ini dilakukan dengan melibatkan 15 data uji kelurahan, dimana 5 kelurahan kategori kurang rawan, 5 kelurahan kategori cukup rawan dan 5 kelurahan kategori sangat rawan serta dengan mengubah nilai toleransi *error* dari

nilai 0.00014, 0.00013, 0.00012, 0.00011 dan 0.00010. dan banyak iterasi sebanyak 500. Berikut ini adalah nilai variable yang digunakan pada skenario :

Banyak Data Uji	=	15
Banyak Iterasi	=	500
Toleransi *Error*	=	0.00014, 0.00013, 0.00012, 0.00011, 0.00010

1. Toleransi *Error* 0.00014

Pengujian skenario 1 dimulai dengan toleransi *error* yang bernilai 0.00014, hasil dari pengujian ini dapat dilihat pada gambar 4.1



Gambar 4.1 Pengujian dengan toleransi *error* 0.00014

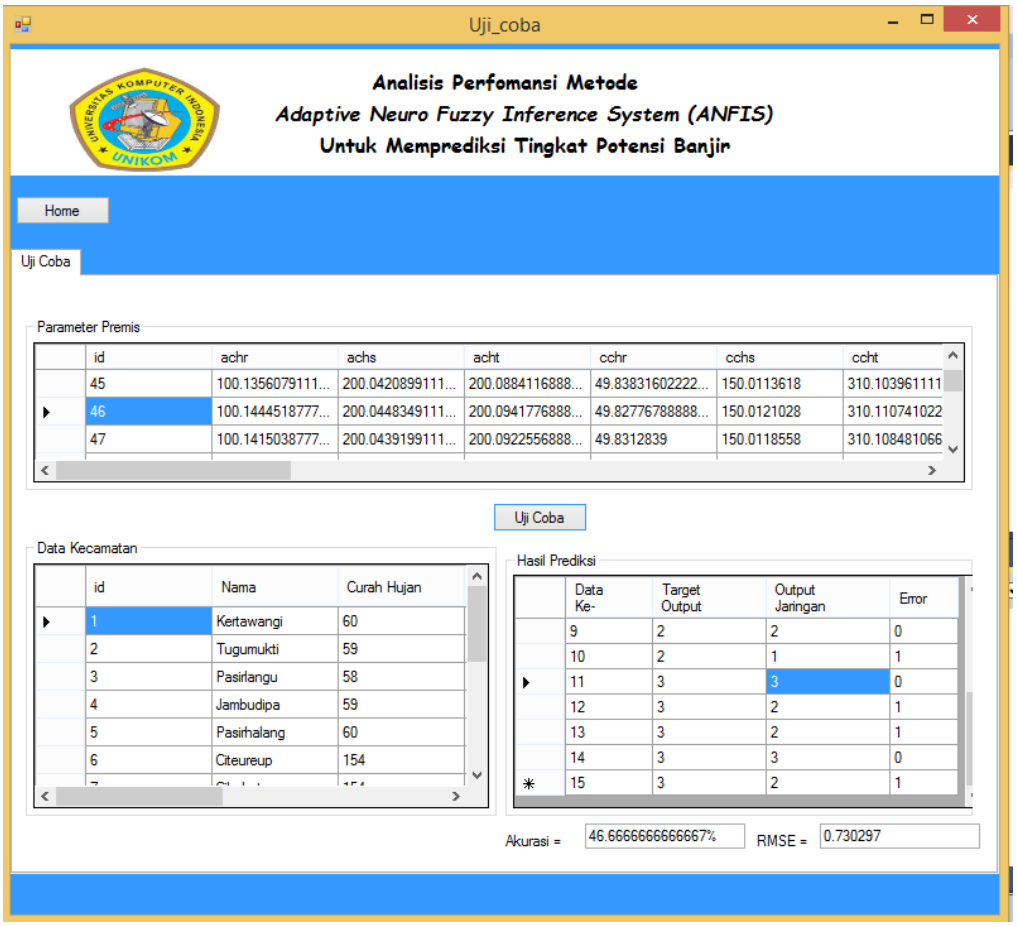
Pada pengujian dengan menggunakan toleransi *error* yang bernilai 0.00014 menghasilkan nilai akurasi 13,33%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.5.

Tabel 4.5 Hasil Keluaran Toleransi Error 0.00014

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	2	-1
3	1	2	-1
4	1	2	-1
5	1	2	-1
6	2	1	1
7	2	1	1
8	2	1	1
9	2	1	1
10	2	1	1
11	3	3	0
12	3	2	1
13	3	2	1
14	3	3	0
15	3	2	1

2. Toleransi *Error* 0.00013

Pengujian skenario 1 dimulai dengan toleransi *error* yang bernilai 0.00013, hasil dari pengujian ini dapat dilihat pada gambar 4.2



Gambar 4.2 Pengujian dengan toleransi *error* 0.00013

Pada pengujian dengan menggunakan toleransi *error* yang bernilai 0.00013 menghasilkan nilai akurasi 46.67% , dengan hasil keluaran sistem dapat dilihat pada tabel 4.6

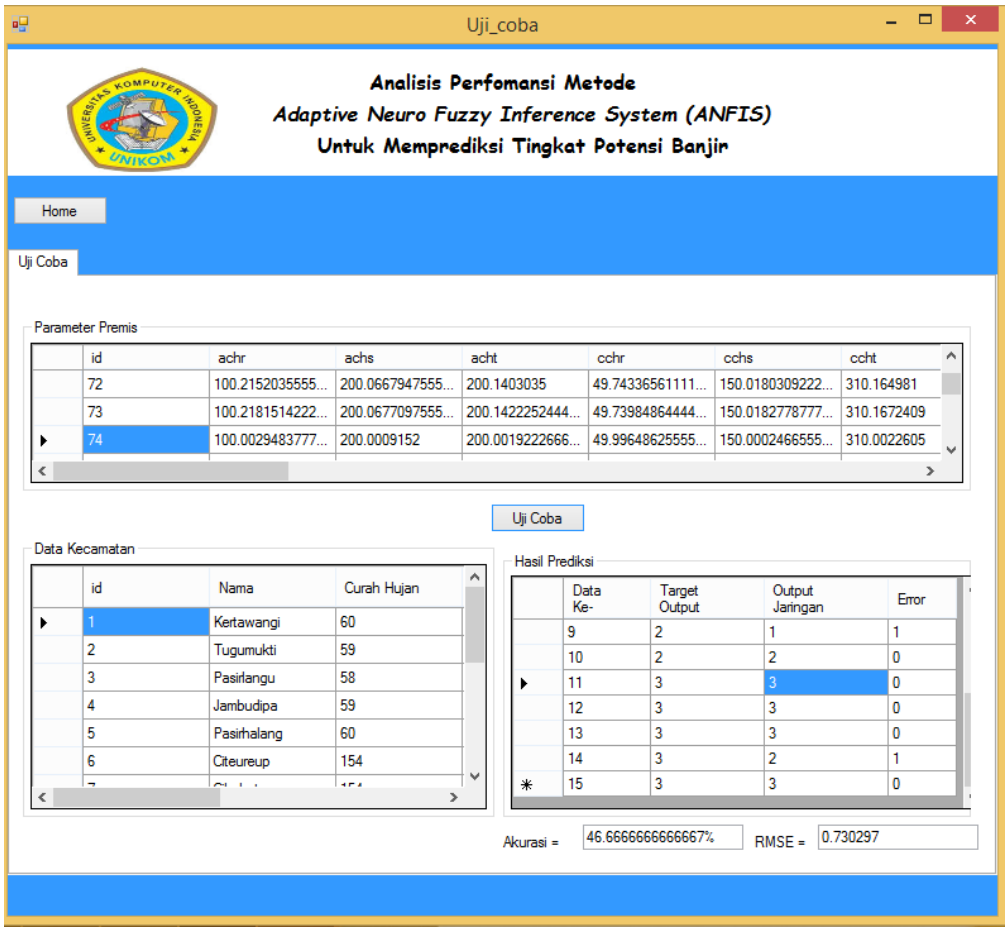
Tabel 4.6 Hasil Keluaran Toleransi Error 0.00013

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	1	0
2	1	2	-1
3	1	1	0
4	1	1	0
5	1	1	0
6	2	1	1
7	2	1	1
8	2	1	1
9	2	2	0

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
10	2	1	1
11	3	3	0
12	3	2	1
13	3	2	1
14	3	3	0
15	3	2	1

3. Toleransi *Error* 0.00012

Pengujian skenario 1 dimulai dengan toleransi *error* yang bernilai 0.00012, hasil dari pengujian ini dapat dilihat pada gambar 4.3



Gambar 4.3 Pengujian dengan toleransi *error* 0.00012

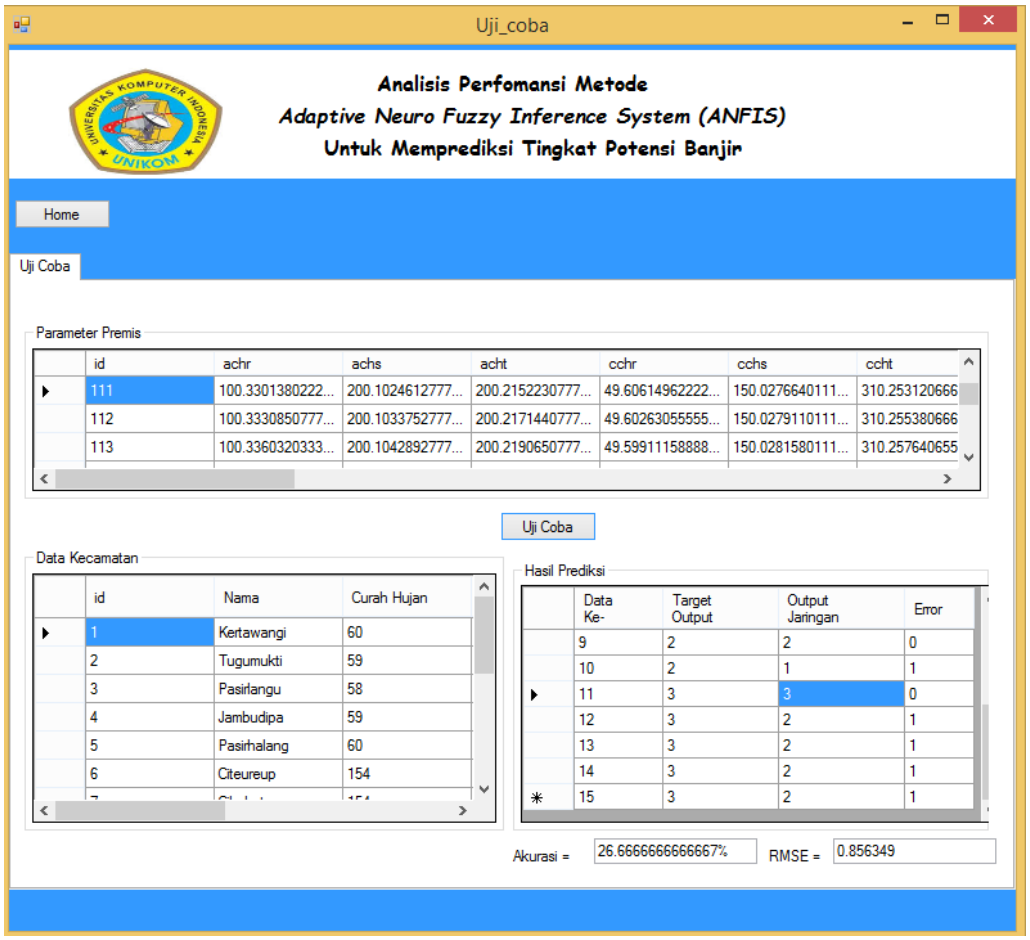
Pada pengujian dengan menggunakan toleransi *error* yang bernilai 0.00012 menghasilkan nilai akurasi 46.67%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.7

Tabel 4.7 Hasil Keluaran Toleransi Error 0.00012

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	2	-1
3	1	2	-1
4	1	2	-1
5	1	2	-1
6	2	2	0
7	2	2	0
8	2	1	1
9	2	1	1
10	2	2	0
11	3	3	0
12	3	3	0
13	3	3	0
14	3	2	1
15	3	3	0

4. Toleransi *Error* 0.00011

Pengujian skenario 1 dimulai dengan toleransi *error* yang bernilai 0.00011, hasil dari pengujian ini dapat dilihat pada gambar 4.4



Gambar 4.4 Pengujian dengan toleransi *error* 0.00011

Pada pengujian dengan menggunakan toleransi *error* yang bernilai 0.00011 menghasilkan nilai akurasi 26.67%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.8

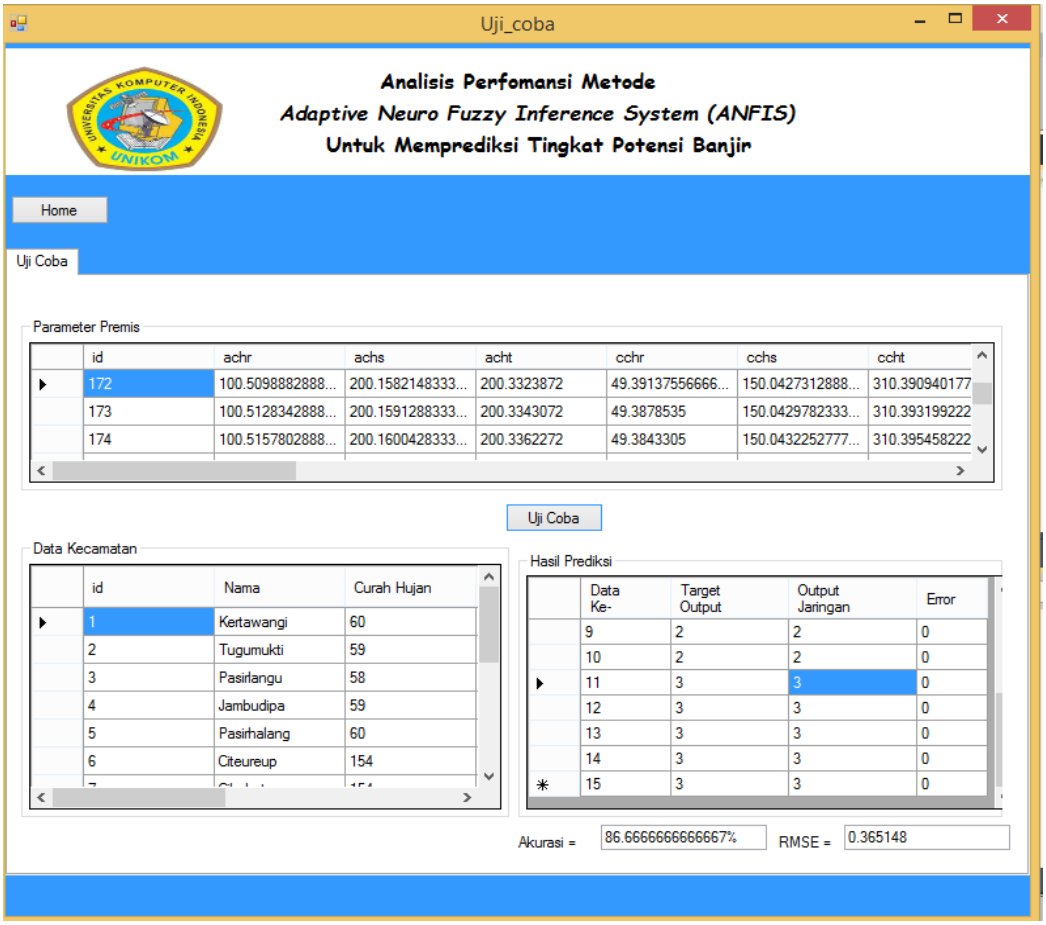
Tabel 4.8 Hasil Keluaran Toleransi Error 0.00011

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	1	0
3	1	2	-1
4	1	2	-1
5	1	1	0
6	2	1	1
7	2	1	1
8	2	1	1

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
9	2	2	0
10	2	1	1
11	3	3	0
12	3	2	1
13	3	2	1
14	3	2	1
15	3	2	1

5. Toleransi *Error* 0.00010

Pengujian skenario 1 dimulai dengan toleransi *error* yang bernilai 0.00010, hasil dari pengujian ini dapat dilihat pada gambar 4.5



Gambar 4.5 Pengujian dengan toleransi *error* 0.00010

Pada pengujian dengan menggunakan toleransi *error* yang bernilai 0.00010 menghasilkan nilai akurasi 86.67%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.9

Tabel 4.9 Hasil Keluaran Toleransi Error 0.00010

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	1	0
2	1	1	0
3	1	1	0
4	1	2	-1
5	1	2	-1
6	2	2	0
7	2	2	0
8	2	2	0
9	2	2	0
10	2	2	0
11	3	3	0
12	3	3	0
13	3	3	0
14	3	3	0
15	3	3	0

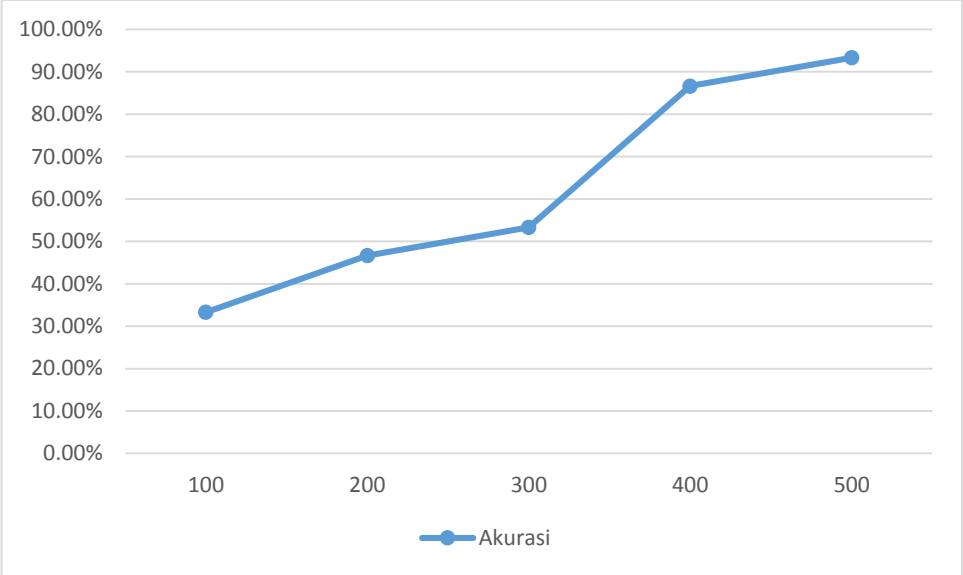
Dari data hasil skenario pengujian diatas didapatkan hasil secara keseluruhan yang dapat dilihat pada tabel 4.10

Tabel 4.10 Data Hasil Skenario Pengujian 1

Toleransi *Error* Pembelajaran	**Akurasi Pengujian**	**Iterasi Terakhir Pembelajaran**	**RMSE Pengujian**
0.00014	13.33%	21	0.930949
0.00013	46.67%	46	0.730297
0.00012	46.67%	74	0.730297
0.00011	26.67%	111	0.856349
0.00010	86.67%	172	0.365148

Dari tabel diatas terlihat bahwa jumlah iterasi mengalami perubahan untuk setiap toleransi *error*, hal itu disebabkan oleh nilai dari toleransi *error* itu sendiri,

dimana semakin kecil toleransi *error* yang diberikan maka semakin besar juga iterasi yang dilakukan.Sehingga dari data tersebut dapat dibuatkan grafik toleransi *error* terhadap akurasi, grafik terebut dapat dilihat pada gambar 4.6



Gambar 4.6 Toleransi *error* terhadap Akurasi

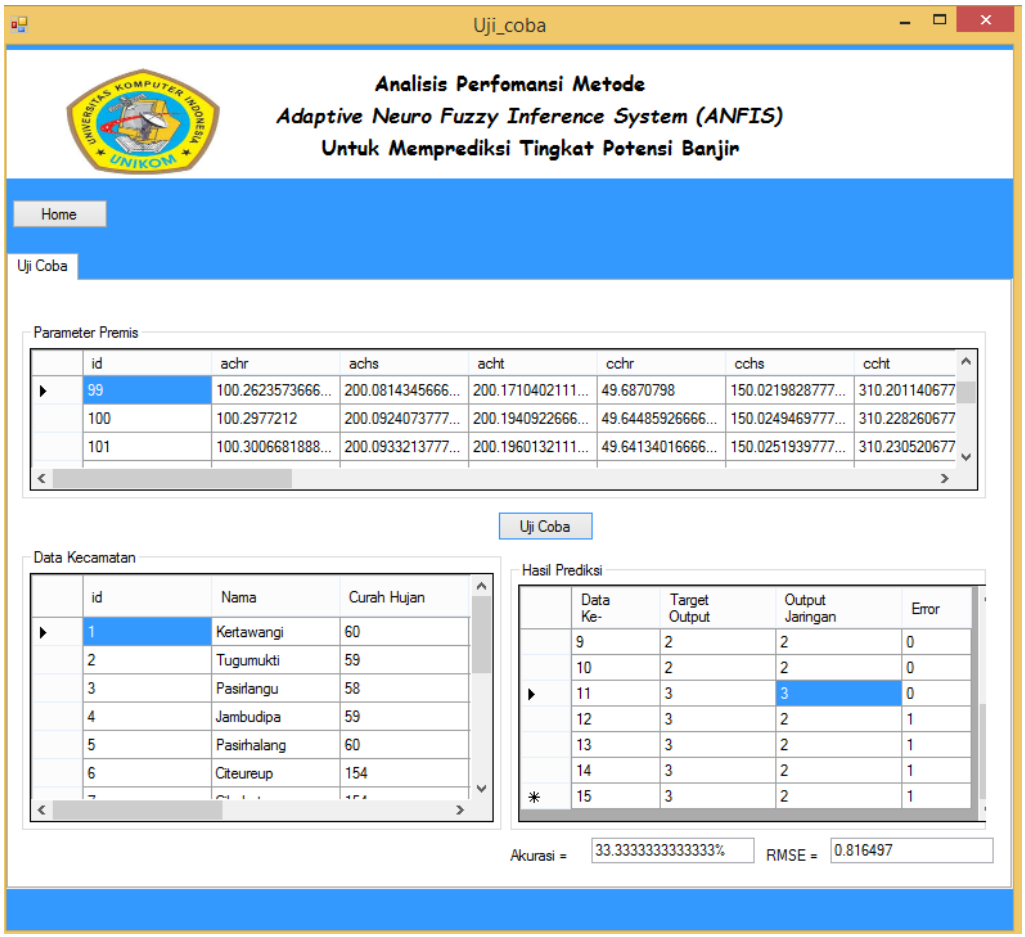
4.2.2 Skenario Pengujian 2

Pada pengujian 2 ini dilakukan dengan melibatkan 15 data uji kelurahan, dimana 5 keluarahan kategori kurang rawan, 5 kelurahan kategori cukup rawan dan 5 kelurahan kategori sangat rawan serta dengan mengubah nilai banyak iterasi dari nilai 100, 200, 300, 400 dan 500. dan nilai toleransi *error* 0.00001. Berikut ini adalah nilai variable yang digunakan pada skenario :

Banyak Data Uji = 15
Banyak Iterasi = 100,200,300,400,500
Toleransi *Error* = 0.00001

1. Banyak iterasi 100

   Pengujian skenario 2 dimulai dengan banyak iterasi 100, hasil dari pengujian ini dapat dilihat pada gambar 4.7



Gambar 4.7 Pengujian dengan banyak iterasi 100

Pada pengujian dengan menggunakan banyak iterasi 100 menghasilkan nilai akurasi 33.33%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.11.

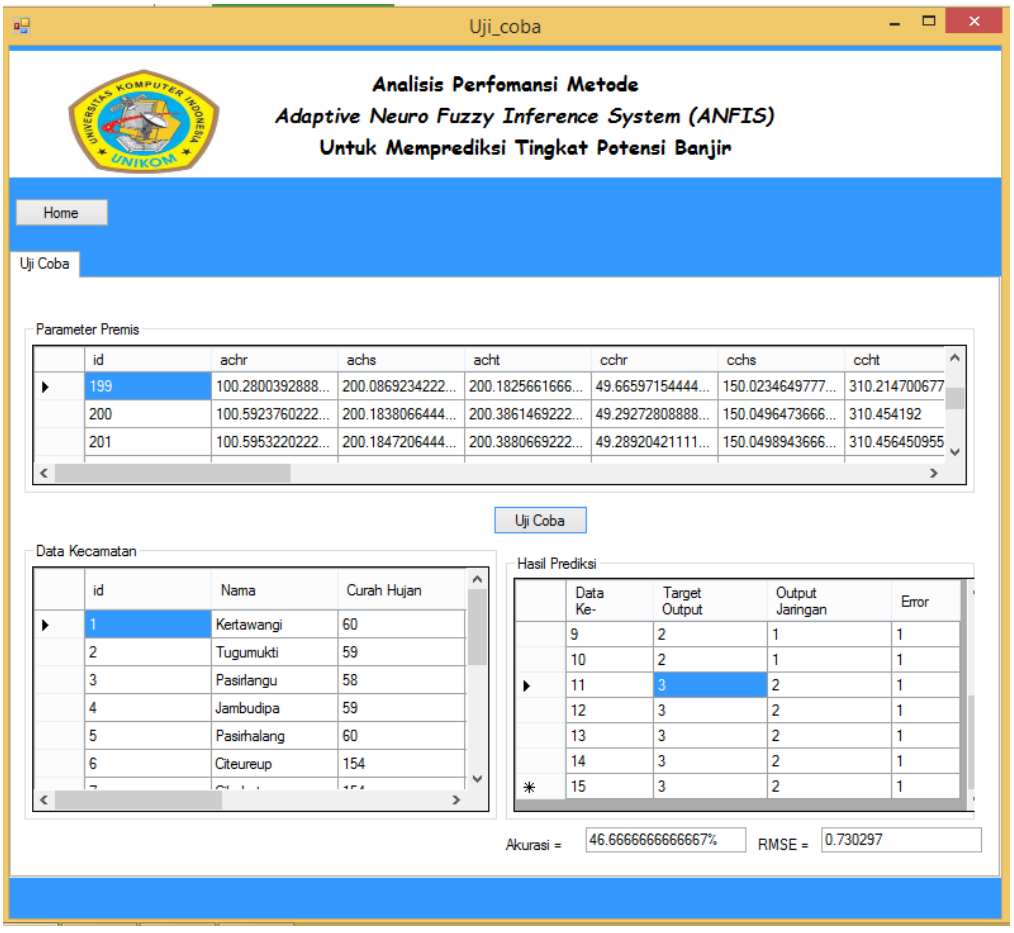
Tabel 4.11 Hasil Keluaran dengan banyak iterasi 100

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	1	0
2	1	2	-1
3	1	2	-1
4	1	2	-1
5	1	2	-1
6	2	2	0
7	2	1	1
8	2	1	1
9	2	2	0
10	2	2	0

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
11	3	3	0
12	3	2	1
13	3	2	1
14	3	2	1
15	3	2	1

2. Banyak iterasi 200

Pengujian skenario 2 dimulai dengan banyak iterasi 200, hasil dari pengujian ini dapat dilihat pada gambar 4.8



Gambar 4.8 Pengujian dengan banyak iterasi 200

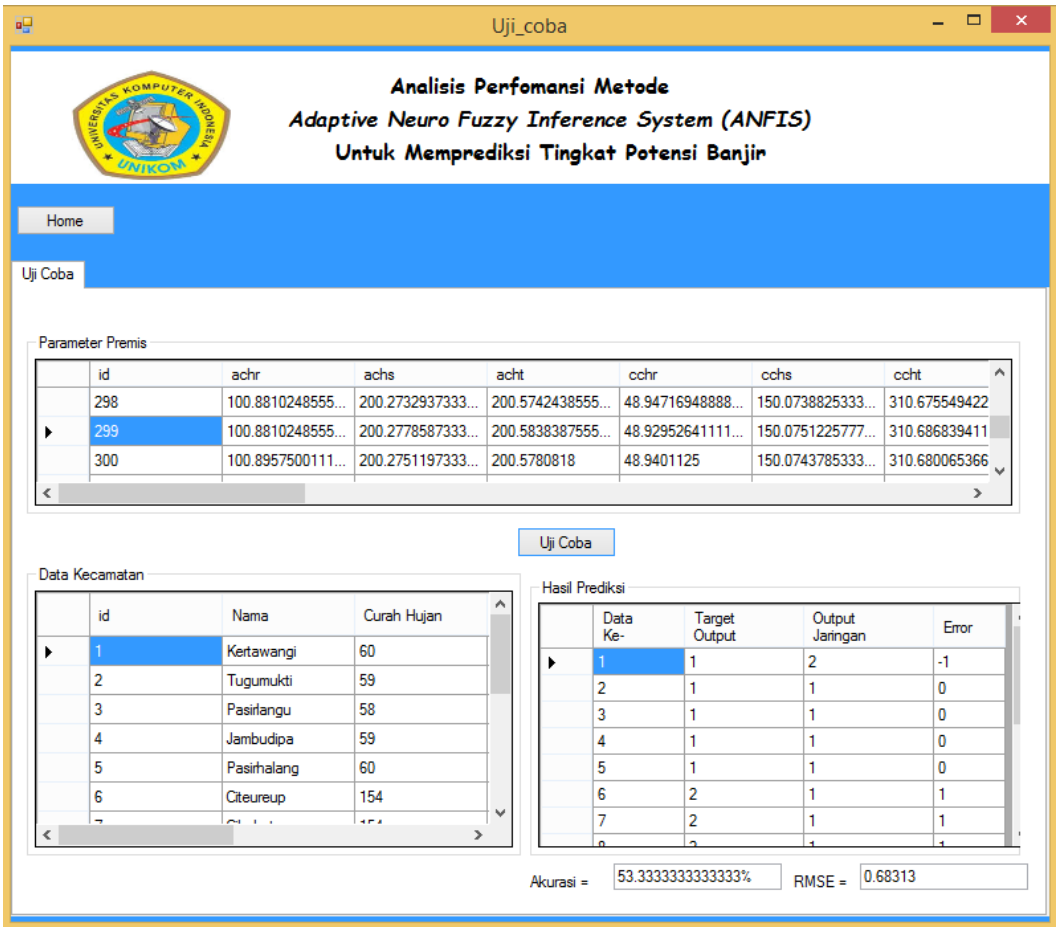
Pada pengujian dengan menggunakan banyak iterasi 200 menghasilkan nilai akurasi 46.67%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.12.

Tabel 4.12 Hasil Keluaran dengan banyak iterasi 200

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	1	0
3	1	1	0
4	1	1	0
5	1	1	0
6	2	2	0
7	2	2	0
8	2	2	0
9	2	1	1
10	2	1	1
11	3	2	1
12	3	2	1
13	3	2	1
14	3	2	1
15	3	2	1

3. Banyak iterasi 300

   Pengujian skenario 2 dimulai dengan banyak iterasi 300, hasil dari pengujian ini dapat dilihat pada gambar 4.9



Gambar 4.9 Pengujian dengan banyak iterasi 300

Pada pengujian dengan menggunakan banyak iterasi 300 menghasilkan nilai akurasi 53.33%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.13.

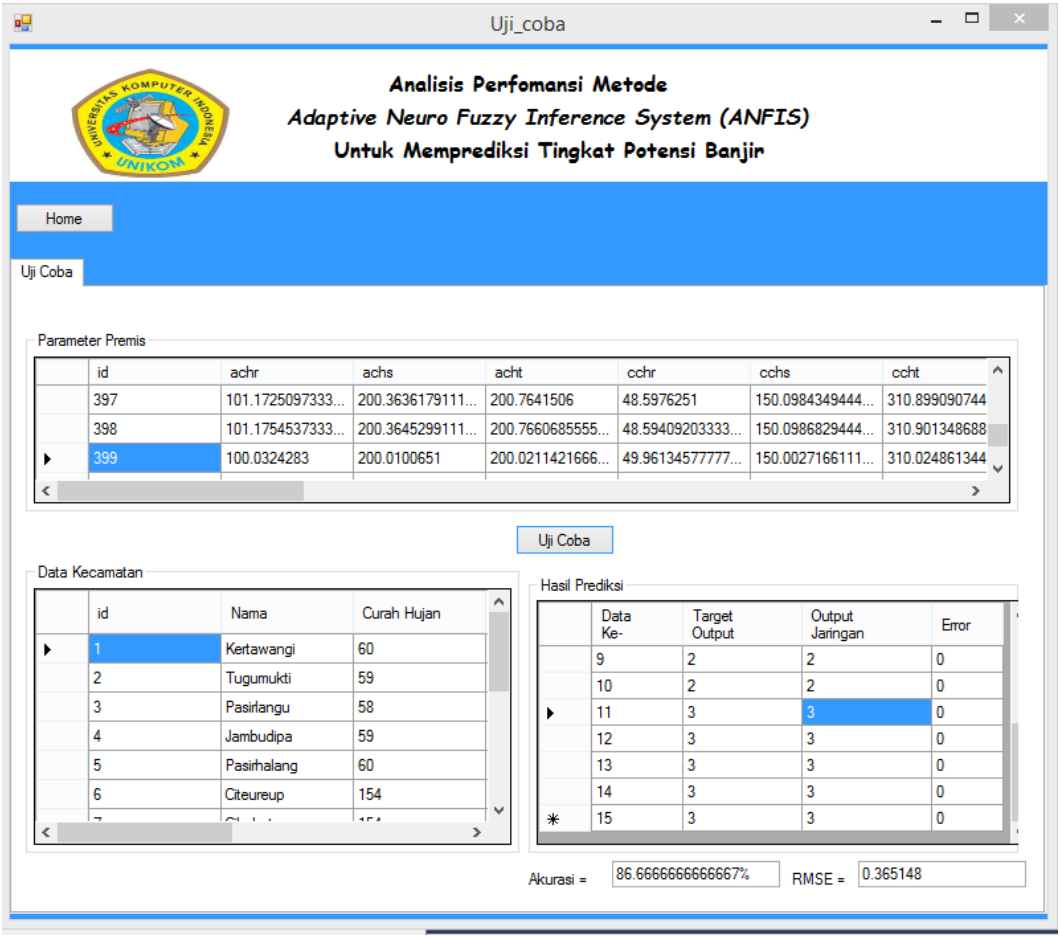
Tabel 4.13 Hasil Keluaran dengan banyak iterasi 300

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	1	0
3	1	1	0
4	1	1	0
5	1	1	0
6	2	1	1
7	2	1	1
8	2	1	1
9	2	1	1

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
10	2	2	0
11	3	3	0
12	3	2	1
13	3	2	1
14	3	3	0
15	3	3	0

4. Banyak iterasi 400

   Pengujian skenario 2 dimulai dengan banyak iterasi 400, hasil dari pengujian ini dapat dilihat pada gambar 4.10



Gambar 4.10 Pengujian dengan banyak iterasi 400

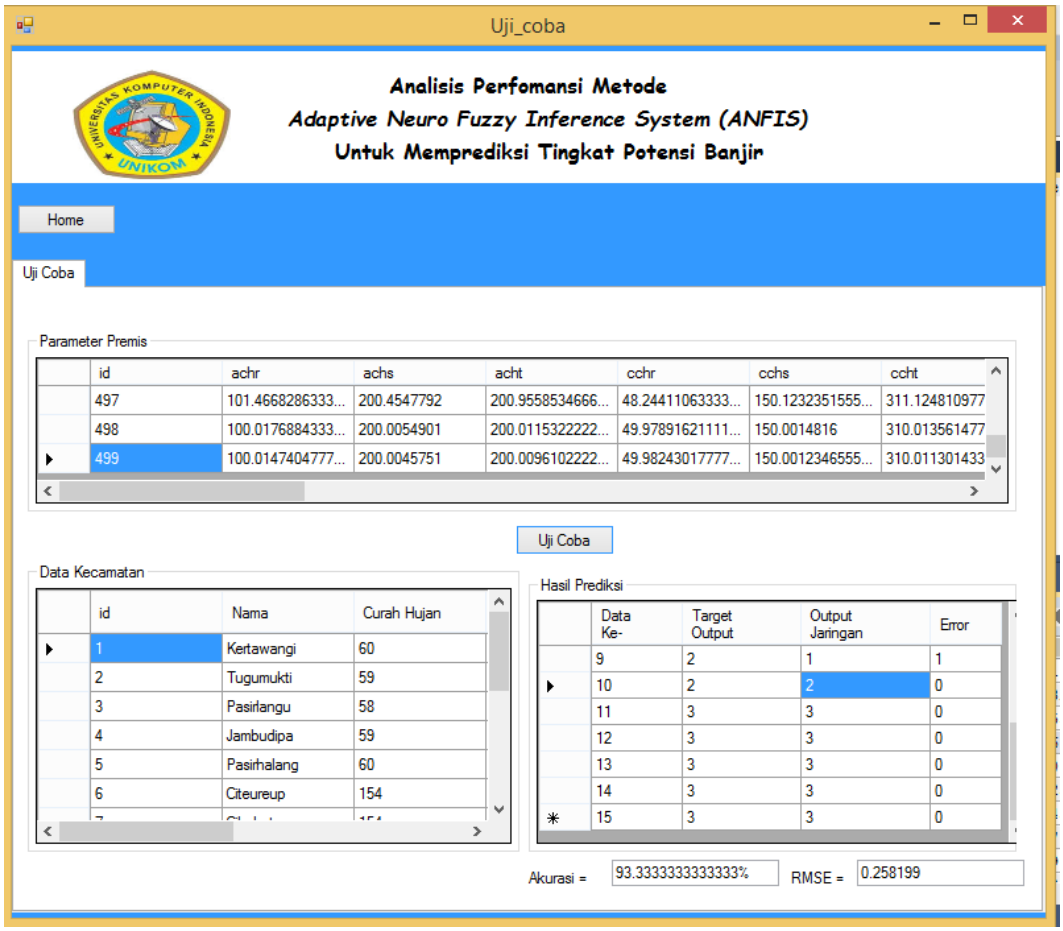
Pada pengujian dengan menggunakan banyak iterasi 400 menghasilkan nilai akurasi 86.67%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.14.

Tabel 4.14 Hasil Keluaran dengan banyak iterasi 400

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	2	-1
2	1	1	0
3	1	1	0
4	1	1	0
5	1	1	0
6	2	2	0
7	2	1	1
8	2	2	0
9	2	2	0
10	2	2	0
11	3	3	0
12	3	3	0
13	3	3	0
14	3	3	0
15	3	3	0

5. Banyak iterasi 500

   Pengujian skenario 2 dimulai dengan banyak iterasi 500, hasil dari pengujian ini dapat dilihat pada gambar 4.11



Gambar 4.11 Pengujian dengan banyak iterasi 500

Pada pengujian dengan menggunakan banyak iterasi 500 menghasilkan nilai akurasi 93.33%, dengan hasil keluaran sistem dapat dilihat pada tabel 4.15.

Tabel 4.15 Hasil Keluaran dengan banyak iterasi 500

Data Ke-	Target *Output*	*Output* Jaringan	*Error*
1	1	1	0
2	1	1	0
3	1	1	0
4	1	1	0
5	1	1	0
6	2	2	0
7	2	2	0
8	2	2	0
9	2	1	1
10	2	2	0

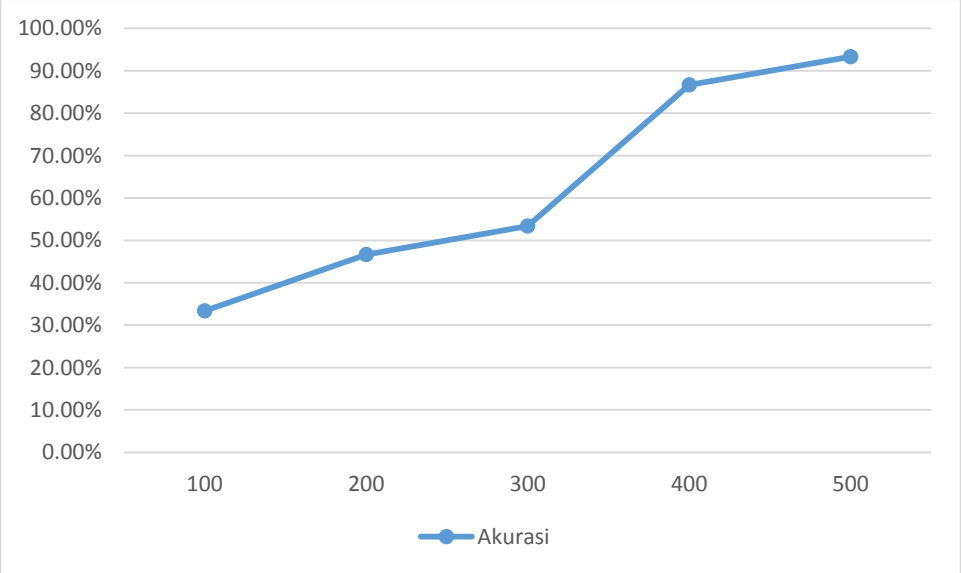
Data Ke-	Target *Output*	*Output* Jaringan	*Error*
11	3	3	0
12	3	3	0
13	3	3	0
14	3	3	0
15	3	3	0

Dari data hasil skenario pengujian diatas didapatkan hasil secara keseluruhan yang dapat dilihat pada tabel 4.16

Tabel 4.16 Data Hasil Skenario Pengujian 2

Iterasi Pembelajaran	**Toleransi *Error* Pembelajaran**	**Akurasi Pengujian**	**RMSE Pengujian**
100	0.00001	33.33%	0.816497
200	0.00001	46.67%	0.730297
300	0.00001	53.33%	0.68313
400	0.00001	86.67%	0.365148
500	0.00001	93.33%	0.258199

Dari data diatas dapat terlihat, semakin banyak iterasi yang diberikan kedalam *system* maka akurasi yang diberikan akan semakin besar, sehingga dari data tersebut dapat dibuatkan grafik banyak iterasi terhadap akurasi, grafik terebut dapat dilihat pada gambar 4.12



Gambar 4.12 Banyak iterasi terhadap akurasi

4.3 Kesimpulan Pengujian

Berdasarkan hasil skenario pengujian 1 dan skenario pengujian 2 dapat ditarik kesimpulan bahwa banyak iterasi dan penentuan toleransi cukup mempengaruhi terhadap besarnya akurasi, dapat terlihat dari hasil skenario pengujian ke 1 dan ke 2 dimana semakin banyak iterasi dan semakin kecil toleransi *error* yang diberikan, maka akurasi yang didapatkan menjadi cukup besar.

Selain itu, berdasarkan hasil skenario pengujian 1 dan skenario pengujian 2 metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) dapat memprediksi tingkat potensi banjir dengan akurasi terbesar sebesar 93.33% dengan rata-rata akurasi sebesar 62.67%.