TESIS

**DESAIN DAN IMPLEMENTASI IOT-BIG DATA ANALYTIC
PADA SMART ENVIRONMENT MONITORING SYSTEM**

Riyadh Arridha
NRP. 2020151004

DOSEN PEMBIMBING

Sritrusta Sukaridhoto, S.T., Ph.D.
Ir. Dadet Pramadihanto, M.Eng., Ph.D.

**PROGRAM STUDI MAGISTER TERAPAN
TEKNIK INFORMATIKA DAN KOMPUTER
PROGRAM PASCASARJANA TEKNOLOGI REKAYASA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
2017**

TESIS

**DESAIN DAN IMPLEMENTASI IOT-BIG DATA
ANALYTIC PADA SMART ENVIRONMENT
MONITORING SYSTEM**

Riyadh Arridha
NRP. 2020151004

DOSEN PEMBIMBING

Sritrusta Sukaridhoto, S.T., Ph.D.
Ir. Dadet Pramadihanto, M.Eng., Ph.D.

**PROGRAM STUDI MAGISTER TERAPAN
TEKNIK INFORMATIKA DAN KOMPUTER
PROGRAM PASCASARJANA TEKNOLOGI REKAYASA
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
2017**

HALAMAN PENGESAHAN

DESAIN DAN IMPLEMENTASI IOT-BIG DATA ANALYTIC PADA SMART ENVIRONMENT MONITORING SYSTEM

Oleh:
Riyadh Arridha
NRP. 2020151004

Tesis ini digunakan sebagai salah satu syarat untuk memperoleh
Gelar Magister Terapan (M.T.)
di
Politeknik Elektronika Negeri Surabaya
2017

Disetujui oleh:

Pembimbing Utama

Sritrusta Sukaridhoto, S.T., Ph.D.
NIP. 197903062002121002

Pembimbing Pendamping :

Ir. Dadet Pramadihanto, M.Eng., Ph.D.
NIP. 196202111988111001

Penguji

Amang Sudarsono, S.T., Ph.D.
NIP. 197409202002121001

Penguji

Dr. Eng. Bima Sena Bayu D, S.ST., M.T.
NIP. 197612151999031003

Penguji

M. Udin Harun Al Rasyid, S.Kom., Ph.D.
NIP. 198108082005011001

Mengetahui,

Ketua Program Pascasarjana
Politeknik Elektronika Negeri Surabaya

Alimdo Barakbah, S.Kom., Ph.D.
NIP. 197308162001121001

HALAMAN PERNYATAAN ORISINALITAS

Dengan ini saya menyatakan bahwa bagian atau keseluruhan tesis ini:

1. adalah hasil karya sendiri dan tidak mengandung unsur plagiat dari pihak lain
2. tidak pernah diajukan untuk mendapatkan gelar akademis pada suatu Perguruan Tinggi
3. tidak pernah dipublikasikan atau ditulis oleh pihak lain
4. mencantumkan rujukan dan kutipan dengan jujur dan benar terhadap sumber referensi lain yang menunjang pembahasan pada tesis.

Apabila ditemukan bukti bahwa pernyataan saya diatas tidak benar, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Politeknik Elektronika Negeri Surabaya.

Surabaya, 8 Agustus 2017

Yang menyatakan,

A handwritten signature in black ink, consisting of a large loop followed by several vertical strokes and a horizontal line extending to the right.

(Riyadh Arridha)

HALAMAN KESEPAKATAN PUBLIKASI

Demi pengembangan ilmu pengetahuan, dengan ini saya menyatakan:

1. memberikan persetujuan kepada Politeknik Elektronika Negeri Surabaya untuk menyimpan, mengolah dalam bentuk pangkalan data, merawat, mengalihmedia/formatkan dan mempublikasikan tesis ini selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.
2. tidak melakukan alihmedia/format dan publikasi dalam bentuk makalah ilmiah dari bagian atau keseluruhan tesis ini ke suatu publikasi ilmiah, pada seminar ataupun jurnal, skala nasional ataupun internasional, kecuali ada persetujuan dari saya dan Dosen Pembimbing Utama, dan mencantumkan nama saya, Dosen Pembimbing Utama dan nama-nama lain (jika ada) yang berkontribusi pada makalah.

Surabaya, 8 Agustus 2017

Yang menyatakan,

A handwritten signature in black ink, consisting of a large loop followed by several vertical strokes and a horizontal line at the end.

(Riyadh Arridha)

KATA PENGANTAR

Segala Puji dan syukur ke hadirat Allah SWT atas rahmat, taufik dan hidayah hingga penulis dapat menyelesaikan penyusunan penelitian yang berjudul “Desain dan Implementasi IoT-Big Data Analytic pada Smart Environment Monitoring System”. Sholawat dan salam kepada Rasulullah SAW, utusan Allah SWT yang membawa manusia dari zaman kegelapan menuju zaman terang benderang. Penelitian ini disusun dalam rangka penyusunan Tesis yang menjadi salah satu persyaratan untuk memperoleh gelar Magister Terapan (M.T.) di Politeknik Elektronika Negeri Surabaya.

Dalam penyelesaian tesis ini, penulis banyak mendapatkan bantuan dari berbagai pihak. Untuk itu penulis menyampaikan ucapan terima kasih setulusnya kepada:

1. Bapak Sritrusta Sukaridhoto, S.T., Ph.D., selaku Dosen Pembimbing Utama yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga penyelesaian tesis ini.
2. Bapak Ir. Dadet Pramadihanto, M.Eng., Ph.D., selaku Dosen Pembimbing Pendamping yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga tesis ini.
3. Bapak Dr. Zainal Arief, S.T., M.T., selaku Direktur Politeknik Elektronika Negeri Surabaya atas segala fasilitas yang telah disediakan.
4. Bapak Ir. Mujiaman sebagai Direktur Utama dan Ibu Siti Nurjannah sebagai Manajer Pengendalian Proses PDAM Surya Sembada Surabaya, yang telah memberikan izin dan kesempatan untuk memperoleh data dan informasi yang mendukung tesis ini.
5. Bapak Ali Ridho Barakbah, S.Kom., Ph.D., selaku Ketua Program Pasca Sarjana atas segala fasilitas yang telah disediakan.
6. Bapak Dosen Penguji yang telah memberikan saran dan masukan dalam penyempurnaan dan pengembangan tesis ini, Bapak Amang Sudarsono, S.T., Ph.D., Bapak Dr. Eng. Bima Sena Bayu Dewantara, S.ST., M.T.,

Bapak M. Udin Harun Al Rasyid, S.Kom., Ph.D., Bapak Iwan Syarif, S.Kom., M.Kom., M.Sc., Ph.D., Bapak Tri Harsono, S.Si., M.Kom., Ph.D., Bapak Dr. Sanggar Dewanto, Bapak Dr. Ir. Endra Pitowarno, M.Eng.

7. Bapak dan Ibu dosen pengajar dan staf program Pascasarjana Terapan Politeknik Elektronika Negeri Surabaya yang telah memberikan ilmu dan bantuan selama masa studi, terimakasih banyak atas ilmu dan bantuan yang diberikan selama ini.
8. KEMENRISTEKDIKTI dan LPDP KEMENKEU atas bantuan Pendidikan BPP-DN tahun 2015 hingga 2017 sehingga memudahkan penulis dalam menjalani pendidikan.
9. Tim mahasiswa peneliti Smart Environment Monitoring and Analytic in Realtime (SEMAR) system, Jordan, Andika, Tegar, Atit, RD, Herwin, Buyung, Luhung, atas bantuan, dukungan dan pertemanan selama penelitian dan penyelesaian tesis ini.
10. Teman-teman seperjuangan Magister Terapan Teknik Informatika dan Komputer angkatan 2015: Pak Rengga Asmara, Muh.Subhan, Munsyi, Ubaidillah Umar, Khotibul Umam, Adhe Widianjaya, Irsal Shabirin, Yesta Medya Mahardhika, Galih Hendra Wibowo, Ardiansyah Alfarouq, Angga Rahagiyanto dan Tresna Maulana Fahrudin. Telah memberikan pengalaman dan menjadi teman seperjuangan.
11. Teman-teman Magister Terapan Teknik Elektro angkatan 2015: Indra Ferdiansyah, Safriudin Rifandi, Cindha Riri Pratiwi, Rafina Destiarti Ainul, Ibu Nofriyani, Gezaq Abror, Ocsirendi, Novrian Eka Sandhi, M. Ikrar Yamin, Pak Djoko Santoso dan Pak Justiawan. Terimakasih atas pertemanan yang terjalin selama ini.
12. Anggota laboratorium lantai 9 (Iseng4h 2016/2017) dan anggota EEPIS Robotic Research Center (ER2C) atas bantuan, dukungan dan pertemanan selama penelitian dan penyelesaian tesis ini.
13. Bapak dan Ibu tercinta (Bapak Alm. Drs. Muh. Yakub Halide dan Ibu Hj. Nasmiah Abdullah, S.Pd.I), yang terus mendoakan dan mendukung penulis hingga saat ini. Semoga rahmat Allah SWT selalu tercurahkan kepada beliau.

14. Bapak dan Ibu mertua tercinta (Bapak Muh. Yusuf B dan Ibu Hasna), yang terus mendoakan dan mendukung penulis hingga saat ini. Semoga rahmat Allah SWT selalu tercurahkan kepada beliau.
15. Istri dan anak tercinta dan tersayang (Titin Magfirah, S.Pd. dan Akhtar Asyira Arridha), istri yang selama penulisan menjadi teman yang saling mendukung dan memotivasi dalam menyelesaikan tesis masing-masing. Semoga segala perjuangan dan pengorbanan membawa berkah ke depannya.
16. Kakak saya tercinta dan suami, (Rina Arfina, S.Pd.I dan Jamaluddin Saleh, S.PI.), dan adik saya tersayang (Rizky Arrafiky, S.H.), atas segala dorongan dan motivasi untuk menjadi lebih baik.
17. Semua pihak yang tidak dapat disebutkan satu-persatu atas semua dukungannya.

Semoga Allah SWT senantiasa memberikan berkat dan anugrah-Nya berlimpah bagi beliau-beliau yang tersebut di atas. Sangat disadari dalam tesis ini terdapat banyak kekurangan oleh karena itu semua saran dan kritik penulis terima dengan lapang dada demi kesempurnaan penulisan tesis ini. Akhirnya harapan penulis semoga tesis ini bermanfaat bagi kita semua.

Surabaya, 8 Agustus 2017

Penulis,

ABSTRAK

Sistem monitoring lingkungan air secara *realtime* merupakan hal yang harus dilakukan dalam upaya menjaga ekosistem air pada negara maritim dan kepulauan, seperti contoh Indonesia yang memiliki kekayaan sumber daya air yang sangat besar. Dalam rangka mengintegrasikan sistem monitoring dengan teknologi *big data* untuk analisa *realtime*, kami membangun sebuah sistem yang dinamakan SEMAR (*Smart Environment Monitoring and Analytic in Realtime system*), yang merupakan platform *IoT-Big data* untuk monitoring lingkungan air. Akan tetapi, SEMAR belum memiliki sistem analisa. Sehingga kami melakukan penelitian untuk membangun sistem analisa untuk klasifikasi kualitas air menggunakan metode Indeks Pencemaran, yang mana akan menjadi ekstensi buat sistem SEMAR. Hasil evaluasi menunjukkan algoritma Linear SVM dan Decision Tree yang digunakan memiliki tingkat akurasi di atas 90% dan rata-rata MSE sebesar 0.019075. Sementara untuk waktu pemrosesan sistem SEMAR hanya membutuhkan rata-rata 0.5 detik untuk memproses data yang diterima hingga proses visualisasi.

Kata Kunci

SEMAR, monitoring kualitas air, analisa realtime, IoT, Big Data, klasifikasi, machine learning

ABSTRACT

Monitoring water conditions in real-time is a critical mission to preserve the water ecosystem in maritime and archipelagic countries, such as Indonesia that is relying on the wealth of water resources. To integrate the water monitoring system into the big data technology for real-time analysis, we have engaged in the ongoing project named SEMAR (Smart Environment Monitoring and Analytic in Real-time system), which provides the IoT-Big Data platform for water monitoring. However, SEMAR does not have an analytical system yet. This paper proposes the analytical system for water quality classification using Pollution Index method, which is an extension of SEMAR. Besides, the communication protocol is updated from REST to MQTT. Furthermore, the real-time user interface is implemented for visualisation. The evaluations confirmed that the data analytic function adopting the linear SVM and Decision Tree algorithms achieves more than 90% for the estimation accuracy with 0.019075 for the MSE. The processing time of the SEMAR system only takes an average 0.5 seconds to process the data to be visualized.

Keywords

SEMAR, water condition monitoring, real-time analysis, IoT, big data, classification, machine learning.

DAFTAR ISI

HALAMAN JUDUL	
HALAMAN PENGESAHAN	
HALAMAN PERNYATAAN ORISINALITAS	i
HALAMAN KESEPAKATAN PUBLIKASI	ii
KATA PENGANTAR	iii
ABSTRAK	vi
DAFTAR ISI	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL	xiv
1 PENDAHULUAN	1
1.1 LATAR BELAKANG	1
1.2 PERMASALAHAN	3
1.3 TUJUAN	4
1.4 MANFAAT	4
1.5 SISTEMATIKA PENULISAN	4
2 KAJIAN PUSTAKA	7
2.1 TEORI PENUNJANG	7
2.1.1 Sumber Daya Air	7
2.1.2 Parameter Mutu Air	8
2.1.3 Metode Indeks Pencemaran	11
2.1.4 <i>Internet of Things & Big Data</i>	13
2.1.4.1 <i>Physical Devices and Controllers</i>	15
2.1.4.2 <i>Connectivity</i>	23
2.1.4.3 <i>Edge Computing</i>	24
2.1.4.4 <i>Data Accumulation</i>	26
2.1.4.5 <i>Data Abstraction</i>	30

2.1.4.6	<i>Application</i>	30
2.1.4.7	<i>Collaboration & Processes</i>	38
2.2	PENELITIAN TERKAIT	38
3	DESAIN SISTEM	41
3.1	DESAIN SISTEM	41
3.1.1	<i>Physical Devices & Controllers</i>	43
3.1.2	<i>Connectivity</i>	45
3.1.3	<i>Edge Computing</i>	46
3.1.4	<i>Data Accumulation</i>	46
3.1.5	<i>Data Abstraction</i>	47
3.1.6	<i>Application</i>	48
3.1.7	<i>Collaboration & Processes</i>	56
3.2	DESAIN <i>BIG DATA</i> SERVER.....	56
3.3	PENGEMBANGAN <i>BIG DATA FRAMEWORK</i>	58
3.3.1	Apache Hadoop.....	58
3.3.2	Apache Hive.....	62
3.3.3	Apache Spark	65
4	EKSPERIMEN DAN ANALISIS	69
4.1	PARAMETER EKSPERIMEN	69
4.2	KARAKTERISTIK DATA	70
4.3	TEMPAT UJI COBA	70
4.4	WAKTU UJI COBA	70
4.5	SPESIFIKASI PERALATAN UJI COBA	71
4.6	HASIL EKSPERIMEN	72
4.6.1	Uji Coba Program	72
4.6.1.1	Lapisan <i>Physical Devices & Controllers</i>	73
4.6.1.2	Lapisan <i>Edge Computing</i>	73
4.6.1.3	Lapisan <i>Data Accumulation</i>	73
4.6.1.4	Lapisan <i>Data Abstraction</i>	75
4.6.1.5	Lapisan <i>Application</i>	75

4.6.1.6 Lapisan <i>Collaboration & Processes</i>	78
4.6.2 Eksperimen pada algoritma klasifikasi	79
4.6.2.1 <i>Confusion Matrix</i>	80
4.6.2.2 MSE, <i>Mislabel</i> , dan <i>Score</i>	81
4.6.2.3 ROC	83
4.6.2.4 <i>Processing Time</i>	86
4.7 ANALISIS HASIL EKSPERIMEN	87
4.8 DISKUSI.....	88
 5 PENUTUP	 91
5.1 KESIMPULAN	91
5.2 SARAN	92

DAFTAR PUSTAKA

LAMPIRAN

DAFTAR GAMBAR

Gambar 1.1	SEMAR system	2
Gambar 2.1	Dimensi IoT menurut ITU-T Y.2060	13
Gambar 2.2	Model referensi dari IoT menurut ITU-T Y.2060	14
Gambar 2.3	Model referensi dari IoT menurut Cisco	15
Gambar 2.4	Struktur pH meter	17
Gambar 2.5	Perbandingan <i>Oxygen Content</i> dengan <i>Temperature</i>	18
Gambar 2.6	Perbandingan <i>Oxygen Content</i> dengan <i>Altitude</i>	19
Gambar 2.7	Nilai EC pada beberapa kasus	20
Gambar 2.8	Skala suhu.....	21
Gambar 2.9	Raspberry Pi 3	22
Gambar 2.10	USB Modem 4G/LTE DT-100 Plus + Soft AP	23
Gambar 2.11	Alur Pengiriman Pesan Topik pada MQTT.....	25
Gambar 2.12	Hadoop Ekosistem.....	28
Gambar 2.13	<i>Machine Learning Flow</i>	31
Gambar 2.14	Visualisasi <i>tree</i> dari algoritma decision tree	32
Gambar 2.15	SVM dengan <i>hyperplane</i> dan margin.....	34
Gambar 2.16	Apache Spark.....	35
Gambar 2.17	Tipe data abstraksi pada Spark	36
Gambar 2.18	Contoh Penggunaan RDD	36
Gambar 2.19	Contoh Penggunaan Dataframe	37
Gambar 2.20	Contoh Penggunaan Dataset.....	37
Gambar 3.1	IoT <i>Reference Layer</i>	41
Gambar 3.2	Desain Sistem	42
Gambar 3.3	Desain sensor kualitas air	43
Gambar 3.4	Lokasi node sensor	44
Gambar 3.5	Algoritma aplikasi pada node sensor.....	45
Gambar 3.6	Data air uji laboratorium PDAM Surya Sembada Surabaya	50
Gambar 3.7	Data air sensor <i>live</i> PDAM Surya Sembada Surabaya	51

Gambar 3.8	Dataset uji laboratorium setelah <i>preprocessing</i>	51
Gambar 3.9	Dataset sensor <i>live</i> setelah <i>preprocessing</i>	51
Gambar 3.10	Prosedur Learning	52
Gambar 3.11	Algoritma klasifikasi real-time.....	53
Gambar 3.12	Proses aliran data pada proses visualisasi	54
Gambar 3.13	Tampilan antar muka lokasi node sensor	55
Gambar 3.14	Monitoring dan klasifikasi <i>realtime</i>	55
Gambar 3.15	Proses <i>learning</i> pada Zeppelin dengan Python	56
Gambar 3.16	Desain infrastruktur server	57
Gambar 3.17	Menjalankan <i>service</i> Hadoop	61
Gambar 3.18	Service Hadoop pada node master dan slave	61
Gambar 3.19	Hadoop Web UI.....	61
Gambar 3.20	Informasi Datanode	62
Gambar 3.21	Pembuatan tabel menggunakan Hive	64
Gambar 3.22	Hasil <i>Query</i> pada Apache Hive	64
Gambar 3.23	Service Spark (Worker) pada node master dan slave.....	67
Gambar 4.1	Aplikasi yang berjalan pada node sensor	73
Gambar 4.2	Aplikasi pada MQTT Broker	73
Gambar 4.3	Aplikasi penyimpanan data pada Hadoop HDFS.....	74
Gambar 4.4	Notifikasi <i>load</i> data ke tabel HDFS	74
Gambar 4.5	Data yang tersimpan pada Hadoop HDFS	75
Gambar 4.6	Aliran data menuju klasifikasi <i>realtime</i>	75
Gambar 4.7	Proses pengiriman hasil klasifikasi data air ke Kafka <i>Broker</i> .	76
Gambar 4.8	Data yang diterima oleh <i>backend</i> Node JS	76
Gambar 4.9	Visualisasi lokasi node sensor	77
Gambar 4.10	Visualisasi data sensor dan hasil klasifikasi	77
Gambar 4.11	Informasi <i>job</i> pada Hadoop <i>cluster</i>	78
Gambar 4.12	Akses data pada Hadoop HDFS menggunakan <i>web interface</i>	78
Gambar 4.13	Proses <i>training</i> dataset	79
Gambar 4.14	Perbandingan skor antar dataset dan algoritma	83
Gambar 4.15	ROC dataset uji laboratorium menggunakan Linear SVM	84
Gambar 4.16	ROC dataset uji laboratorium menggunakan Decision Tree ...	85

Gambar 4.17 ROC dataset sensor <i>live</i> menggunakan Linear SVM	85
Gambar 4.18 ROC dataset sensor <i>live</i> menggunakan Decision Tree.....	86
Gambar 4.19 <i>Processing time</i> dalam mS vs jumlah node sensor.....	86

DAFTAR TABEL

Tabel 2.1	Label Indeks Pencemaran	12
Tabel 2.2	Spesifikasi Raspberry Pi 3.....	22
Tabel 2.3	Spesifikasi USB Modem 4G/LTE DT-100 Plus + Soft AP	23
Tabel 3.1	Titik Latitude dan Longitude lokasi node sensor	44
Tabel 3.2	Kategori Indeks Pencemaran	49
Tabel 3.3	Penentuan label Indeks Pencemaran	50
Tabel 3.4	Standar Kualitas Air	50
Tabel 3.5	Konfigurasi alamat IP	57
Tabel 4.1	Jadwal Pelaksanaan	71
Tabel 4.2	Spesifikasi <i>Hardware</i> dan <i>Software</i>	71
Tabel 4.3	<i>Confusion matrix</i> dataset uji laboratorium menggunakan Linear SVM	80
Tabel 4.4	<i>Confusion matrix</i> dataset uji laboratorium menggunakan Decision Tree	80
Tabel 4.5	<i>Confusion matrix</i> dataset sensor <i>live</i> menggunakan linear SVM	81
Tabel 4.6	<i>Confusion matrix</i> dataset sensor <i>live</i> menggunakan Decision Tree	81
Tabel 4.7	Nilai <i>mislabel</i> , <i>score</i> dan MSE	82
Tabel 4.8	Tabel nilai AUC	84

HALAMAN PERNYATAAN ORISINALITAS

Dengan ini saya menyatakan bahwa bagian atau keseluruhan tesis ini:

1. adalah hasil karya sendiri dan tidak mengandung unsur plagiat dari pihak lain
2. tidak pernah diajukan untuk mendapatkan gelar akademis pada suatu Perguruan Tinggi
3. tidak pernah dipublikasikan atau ditulis oleh pihak lain
4. mencantumkan rujukan dan kutipan dengan jujur dan benar terhadap sumber referensi lain yang menunjang pembahasan pada tesis.

Apabila ditemukan bukti bahwa pernyataan saya diatas tidak benar, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Politeknik Elektronika Negeri Surabaya.

Surabaya, 8 Agustus 2017

Yang menyatakan,

(Riyadh Arridha)

HALAMAN KESEPAKATAN PUBLIKASI

Demi pengembangan ilmu pengetahuan, dengan ini saya menyatakan:

1. memberikan persetujuan kepada Politeknik Elektronika Negeri Surabaya untuk menyimpan, mengolah dalam bentuk pangkalan data, merawat, mengalihmedia/formatkan dan mempublikasikan tesis ini selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.
2. tidak melakukan alihmedia/format dan publikasi dalam bentuk makalah ilmiah dari bagian atau keseluruhan tesis ini ke suatu publikasi ilmiah, pada seminar ataupun jurnal, skala nasional ataupun internasional, kecuali ada persetujuan dari saya dan Dosen Pembimbing Utama, dan mencantumkan nama saya, Dosen Pembimbing Utama dan nama-nama lain (jika ada) yang berkontribusi pada makalah.

Surabaya, 8 Agustus 2017

Yang menyatakan,

(Riyadh Arridha)

KATA PENGANTAR

Segala Puji dan syukur ke hadirat Allah SWT atas rahmat, taufik dan hidayah hingga penulis dapat menyelesaikan penyusunan penelitian yang berjudul “Desain dan Implementasi IoT-Big Data Analytic pada Smart Environment Monitoring System”. Sholawat dan salam kepada Rasulullah SAW, utusan Allah SWT yang membawa manusia dari zaman kegelapan menuju zaman terang benderang. Penelitian ini disusun dalam rangka penyusunan Tesis yang menjadi salah satu persyaratan untuk memperoleh gelar Magister Terapan (M.T.) di Politeknik Elektronika Negeri Surabaya.

Dalam penyelesaian tesis ini, penulis banyak mendapatkan bantuan dari berbagai pihak. Untuk itu penulis menyampaikan ucapan terima kasih setulusnya kepada:

1. Bapak Sritrusta Sukaridhoto, S.T., Ph.D., selaku Dosen Pembimbing Utama yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga penyelesaian tesis ini.
2. Bapak Ir. Dadet Pramadihanto, M.Eng., Ph.D., selaku Dosen Pembimbing Pendamping yang dengan sabar telah memberi arahan dan bimbingan dalam proses pembuatan hingga tesis ini.
3. Bapak Dr. Zainal Arief, S.T., M.T., selaku Direktur Politeknik Elektronika Negeri Surabaya atas segala fasilitas yang telah disediakan.
4. Bapak Ir. Mujiaman sebagai Direktur Utama dan Ibu Siti Nurjannah sebagai Manajer Pengendalian Proses PDAM Surya Sembada Surabaya, yang telah memberikan izin dan kesempatan untuk memperoleh data dan informasi yang mendukung tesis ini.
5. Bapak Ali Ridho Barakbah, S.Kom., Ph.D., selaku Ketua Program Pasca Sarjana atas segala fasilitas yang telah disediakan.
6. Bapak Dosen Penguji yang telah memberikan saran dan masukan dalam penyempurnaan dan pengembangan tesis ini, Bapak Amang Sudarsono, S.T., Ph.D., Bapak Dr. Eng. Bima Sena Bayu Dewantara, S.ST., M.T.,

Bapak M. Udin Harun Al Rasyid, S.Kom., Ph.D., Bapak Iwan Syarif, S.Kom., M.Kom., M.Sc., Ph.D., Bapak Tri Harsono, S.Si., M.Kom., Ph.D., Bapak Dr. Sanggar Dewanto, Bapak Dr. Ir. Endra Pitowarno, M.Eng.

7. Bapak dan Ibu dosen pengajar dan staf program Pascasarjana Terapan Politeknik Elektronika Negeri Surabaya yang telah memberikan ilmu dan bantuan selama masa studi, terimakasih banyak atas ilmu dan bantuan yang diberikan selama ini.
8. KEMENRISTEKDIKTI dan LPDP KEMENKEU atas bantuan Pendidikan BPP-DN tahun 2015 hingga 2017 sehingga memudahkan penulis dalam menjalani pendidikan.
9. Tim mahasiswa peneliti Smart Environment Monitoring and Analytic in Realtime (SEMAR) system, Jordan, Andika, Tegar, Atit, RD, Herwin, Buyung, Luhung, atas bantuan, dukungan dan pertemanan selama penelitian dan penyelesaian tesis ini.
10. Teman-teman seperjuangan Magister Terapan Teknik Informatika dan Komputer angkatan 2015: Pak Rengga Asmara, Muh.Subhan, Munsyi, Ubaidillah Umar, Khotibul Umam, Adhe Widianjaya, Irsal Shabirin, Yesta Medya Mahardhika, Galih Hendra Wibowo, Ardiansyah Alfarouq, Angga Rahagiyanto dan Tresna Maulana Fahrudin. Telah memberikan pengalaman dan menjadi teman seperjuangan.
11. Teman-teman Magister Terapan Teknik Elektro angkatan 2015: Indra Ferdiansyah, Safriudin Rifandi, Cindha Riri Pratiwi, Rafina Destiarti Ainul, Ibu Nofriyani, Gezaq Abror, Ocsirendi, Novrian Eka Sandhi, M. Ikrar Yamin, Pak Djoko Santoso dan Pak Justiawan. Terimakasih atas pertemanan yang terjalin selama ini.
12. Anggota laboratorium lantai 9 (Iseng4h 2016/2017) dan anggota EEPIS Robotic Research Center (ER2C) atas bantuan, dukungan dan pertemanan selama penelitian dan penyelesaian tesis ini.
13. Bapak dan Ibu tercinta (Bapak Alm. Drs. Muh. Yakub Halide dan Ibu Hj. Nasmiah Abdullah, S.Pd.I), yang terus mendoakan dan mendukung penulis hingga saat ini. Semoga rahmat Allah SWT selalu tercurahkan kepada beliau.

14. Bapak dan Ibu mertua tercinta (Bapak Muh. Yusuf B dan Ibu Hasna), yang terus mendoakan dan mendukung penulis hingga saat ini. Semoga rahmat Allah SWT selalu tercurahkan kepada beliau.
15. Istri dan anak tercinta dan tersayang (Titin Magfirah, S.Pd. dan Akhtar Asyira Arridha), istri yang selama penulisan menjadi teman yang saling mendukung dan memotivasi dalam menyelesaikan tesis masing-masing. Semoga segala perjuangan dan pengorbanan membawa berkah ke depannya.
16. Kakak saya tercinta dan suami, (Rina Arfina, S.Pd.I dan Jamaluddin Saleh, S.PI.), dan adik saya tersayang (Rizky Arrafiky, S.H.), atas segala dorongan dan motivasi untuk menjadi lebih baik.
17. Semua pihak yang tidak dapat disebutkan satu-persatu atas semua dukungannya.

Semoga Allah SWT senantiasa memberikan berkat dan anugrah-Nya berlimpah bagi beliau-beliau yang tersebut di atas. Sangat disadari dalam tesis ini terdapat banyak kekurangan oleh karena itu semua saran dan kritik penulis terima dengan lapang dada demi kesempurnaan penulisan tesis ini. Akhirnya harapan penulis semoga tesis ini bermanfaat bagi kita semua.

Surabaya, 8 Agustus 2017

Penulis,

ABSTRAK

Sistem monitoring lingkungan air secara *realtime* merupakan hal yang harus dilakukan dalam upaya menjaga ekosistem air pada negara maritim dan kepulauan, seperti contoh Indonesia yang memiliki kekayaan sumber daya air yang sangat besar. Dalam rangka mengintegrasikan sistem monitoring dengan teknologi *big data* untuk analisa *realtime*, kami membangun sebuah sistem yang dinamakan SEMAR (*Smart Environment Monitoring and Analytic in Realtime system*), yang merupakan platform *IoT-Big data* untuk monitoring lingkungan air. Akan tetapi, SEMAR belum memiliki sistem analisa. Sehingga kami melakukan penelitian untuk membangun sistem analisa untuk klasifikasi kualitas air menggunakan metode Indeks Pencemaran, yang mana akan menjadi ekstensi buat sistem SEMAR. Hasil evaluasi menunjukkan algoritma Linear SVM dan Decision Tree yang digunakan memiliki tingkat akurasi di atas 90% dan rata-rata MSE sebesar 0.019075. Sementara untuk waktu pemrosesan sistem SEMAR hanya membutuhkan rata-rata 0.5 detik untuk memproses data yang diterima hingga proses visualisasi.

Kata Kunci

SEMAR, monitoring kualitas air, analisa realtime, IoT, Big Data, klasifikasi, machine learning

ABSTRACT

Monitoring water conditions in real-time is a critical mission to preserve the water ecosystem in maritime and archipelagic countries, such as Indonesia that is relying on the wealth of water resources. To integrate the water monitoring system into the big data technology for real-time analysis, we have engaged in the ongoing project named SEMAR (Smart Environment Monitoring and Analytic in Real-time system), which provides the IoT-Big Data platform for water monitoring. However, SEMAR does not have an analytical system yet. This paper proposes the analytical system for water quality classification using Pollution Index method, which is an extension of SEMAR. Besides, the communication protocol is updated from REST to MQTT. Furthermore, the real-time user interface is implemented for visualisation. The evaluations confirmed that the data analytic function adopting the linear SVM and Decision Tree algorithms achieves more than 90% for the estimation accuracy with 0.019075 for the MSE. The processing time of the SEMAR system only takes an average 0.5 seconds to process the data to be visualized.

Keywords

SEMAR, water condition monitoring, real-time analysis, IoT, big data, classification, machine learning.

DAFTAR ISI

HALAMAN JUDUL	
HALAMAN PENGESAHAN	
HALAMAN PERNYATAAN ORISINALITAS	i
HALAMAN KESEPAKATAN PUBLIKASI	ii
KATA PENGANTAR	iii
ABSTRAK	vi
DAFTAR ISI	viii
DAFTAR GAMBAR	x
DAFTAR TABEL	xiii
1 PENDAHULUAN	1
1.1 LATAR BELAKANG	1
1.2 PERMASALAHAN	3
1.3 TUJUAN	4
1.4 MANFAAT	4
1.5 SISTEMATIKA PENULISAN	4
2 KAJIAN PUSTAKA	6
2.1 TEORI PENUNJANG	6
2.1.1 Sumber Daya Air	6
2.1.2 Parameter Mutu Air	7
2.1.3 Metode Indeks Pencemaran	10
2.1.4 Teknologi <i>Internet of Things & Big Data</i>	12
2.2 PENELITIAN TERKAIT	43
3 DESAIN SISTEM	46
3.1 DESAIN SISTEM	46
3.1.1 <i>Physical Devices & Controllers</i>	48

3.1.2	<i>Connectivity</i>	50
3.1.3	<i>Edge Computing</i>	51
3.1.4	<i>Data Accumulation</i>	51
3.1.5	<i>Data Abstraction</i>	52
3.1.6	<i>Application</i>	53
3.1.7	<i>Collaboration & Processes</i>	61
3.2	DESAIN <i>BIG DATA</i> SERVER	61
3.3	PENGEMBANGAN <i>BIG DATA FRAMEWORK</i>	63
3.3.1	Apache Hadoop	63
3.3.2	Apache Hive	68
3.3.3	Apache Spark	71
4	EKSPERIMEN DAN ANALISIS	72
4.1	PARAMETER EKSPERIMEN	72
4.2	KARAKTERISTIK DATA	73
4.3	TEMPAT UJI COBA	73
4.4	WAKTU UJI COBA	73
4.5	SPESIFIKASI PERALATAN UJI COBA	74
4.6	HASIL EKSPERIMEN	75
4.6.1	Uji Coba Program	75
4.6.2	Eksperimen pada algoritma klasifikasi	82
4.6.3	<i>Processing Time</i>	89
4.7	ANALISIS HASIL EKSPERIMEN	90
4.8	DISKUSI	91
5	PENUTUP	93
5.1	KESIMPULAN	93
5.2	SARAN	94

DAFTAR PUSTAKA

LAMPIRAN

DAFTAR GAMBAR

Gambar 1.1	SEMAR system	2
Gambar 2.1	Dimensi IoT menurut ITU-T Y.2060	13
Gambar 2.2	Model referensi dari IoT menurut ITU-T Y.2060	14
Gambar 2.3	Model referensi dari IoT menurut Cisco	14
Gambar 2.4	Struktur pH meter	16
Gambar 2.5	Konsentrasi DO pada berbagai musim	17
Gambar 2.6	Perbandingan <i>Oxygen Content</i> dengan <i>Temperature</i>	18
Gambar 2.7	Perbandingan <i>Oxygen Content</i> dengan <i>Altitude</i>	18
Gambar 2.8	Nilai EC pada beberapa kasus	20
Gambar 2.9	Perkiraan nilai salinitas pada perairan	21
Gambar 2.10	Skala suhu	21
Gambar 2.11	Raspberry Pi 3 model B	23
Gambar 2.12	USB Modem 4G/LTE DT-100 Plus + Soft AP	24
Gambar 2.13	Alur Pengiriman Pesan Topik pada MQTT	26
Gambar 2.14	Big Data 4V	27
Gambar 2.15	Hadoop Ekosistem	29
Gambar 2.16	Skema Apache Hive dalam sistem	31
Gambar 2.17	Skema Apache Kafka	32
Gambar 2.18	<i>Machine Learning Flow</i>	33
Gambar 2.19	Visualisasi <i>tree</i> dari algoritma decision tree	34
Gambar 2.20	SVM dengan <i>hyperplane</i> dan margin	36
Gambar 2.21	Apache Spark	37
Gambar 2.22	Skema Apache Spark untuk analisa <i>realtime</i>	39
Gambar 2.23	Tipe data abstraksi pada Spark	40
Gambar 2.24	Contoh Penggunaan RDD	40
Gambar 2.25	Contoh Penggunaan Dataframe	41
Gambar 2.26	Contoh Penggunaan Dataset	41
Gambar 2.27	<i>User interface</i> Zeppelin Notebook	42

Gambar 3.1	Gambaran Umum Sistem	46
Gambar 3.2	Desain Sistem	47
Gambar 3.3	Desain sensor kualitas air	48
Gambar 3.4	Lokasi node sensor	49
Gambar 3.5	Algoritma aplikasi pada node sensor.....	50
Gambar 3.6	Data air uji laboratorium PDAM Surya Sembada Surabaya....	53
Gambar 3.7	Data air sensor <i>live</i> PDAM Surya Sembada Surabaya	54
Gambar 3.8	Dataset uji laboratorium setelah <i>preprocessing</i>	56
Gambar 3.9	Dataset sensor <i>live</i> setelah <i>preprocessing</i>	56
Gambar 3.10	Prosedur Learning	57
Gambar 3.11	Algoritma klasifikasi <i>realtime</i>	58
Gambar 3.12	Proses aliran data pada proses visualisasi	59
Gambar 3.13	Tampilan antar muka lokasi node sensor	60
Gambar 3.14	Monitoring dan klasifikasi <i>realtime</i>	60
Gambar 3.15	Proses <i>learning</i> pada Zeppelin dengan Python	61
Gambar 3.16	Desain infrastruktur server	62
Gambar 3.17	Menjalankan <i>service</i> Hadoop	67
Gambar 3.18	Service Hadoop pada node master dan slave	67
Gambar 3.19	Hadoop Web UI.....	67
Gambar 3.20	Informasi Datanode	68
Gambar 3.21	Pembuatan tabel menggunakan Hive	70
Gambar 3.22	Hasil <i>Query</i> pada Apache Hive	70
Gambar 3.23	Service Spark (Worker) pada node master dan slave.....	71
Gambar 4.1	Aplikasi yang berjalan pada node sensor	76
Gambar 4.2	Aplikasi pada MQTT <i>Broker</i>	76
Gambar 4.3	Aplikasi penyimpanan data pada Hadoop HDFS.....	77
Gambar 4.4	Notifikasi <i>load</i> data ke tabel HDFS	77
Gambar 4.5	Data yang tersimpan pada Hadoop HDFS	78
Gambar 4.6	Aliran data menuju klasifikasi <i>realtime</i>	78
Gambar 4.7	Proses pengiriman hasil klasifikasi data air ke Kafka <i>Broker</i> .	79
Gambar 4.8	Data yang diterima oleh <i>backend</i> Node JS	79
Gambar 4.9	Visualisasi lokasi node sensor	80

Gambar 4.10 Visualisasi data sensor dan hasil klasifikasi	80
Gambar 4.11 Informasi <i>job</i> pada Hadoop <i>cluster</i>	81
Gambar 4.12 Akses data pada Hadoop HDFS menggunakan <i>web interface</i>	81
Gambar 4.13 Proses <i>training</i> dataset	82
Gmabar 4.14 Perbandingan skor antar dataset dan algoritma	86
Gambar 4.15 ROC dataset uji laboratorium menggunakan Linear SVM	87
Gambar 4.16 ROC dataset uji laboratorium menggunakan Decision Tree ...	88
Gambar 4.17 ROC dataset sensor <i>live</i> menggunakan Linear SVM	88
Gambar 4.18 ROC dataset sensor <i>live</i> menggunakan Decision Tree.....	89
Gambar 4.19 <i>Processing time</i> dalam mS vs jumlah node sensor.....	89

DAFTAR TABEL

Tabel 2.1	Label Indeks Pencemaran	12
Tabel 2.2	Spesifikasi Raspberry Pi 3 model B	23
Tabel 2.3	Spesifikasi USB Modem 4G/LTE DT-100 Plus + Soft AP	24
Tabel 3.1	Titik Latitude dan Longitude lokasi node sensor	49
Tabel 3.2	Kategori Indeks Pencemaran	54
Tabel 3.3	Penentuan label Indeks Pencemaran	55
Tabel 3.4	Standar Kualitas Air	55
Tabel 3.5	Konfigurasi alamat IP	62
Tabel 4.1	Jadwal Pelaksanaan	74
Tabel 4.2	Spesifikasi <i>Hardware</i>	74
Tabel 4.3	Spesifikasi <i>Software</i>	75
Tabel 4.4	<i>Confusion matrix</i> dataset uji laboratorium menggunakan Linear SVM	83
Tabel 4.5	<i>Confusion matrix</i> dataset uji laboratorium menggunakan Decision Tree	83
Tabel 4.6	<i>Confusion matrix</i> dataset sensor <i>live</i> menggunakan linear SVM	84
Tabel 4.7	<i>Confusion matrix</i> dataset sensor <i>live</i> menggunakan Decision Tree	84
Tabel 4.8	Nilai <i>mislabel</i> , <i>score</i> dan MSE	85
Tabel 4.9	Tabel nilai AUC	87

BAB 1

PENDAHULUAN

1.1 LATAR BELAKANG

Indonesia adalah negara maritim dan negara kepulauan yang terbentang lebih dari 17.000 pulau dari Sabang sampai Merauke. Secara persentase luas laut Indonesia adalah 70% dari luas Indonesia secara keseluruhan. Kekayaan laut Indonesia tidak dapat terbantahkan lagi dan posisinya sangat vital untuk menunjang kelangsungan hidup banyak masyarakatnya. Namun hal tersebut tidak dilandasi dengan kesadaran akan pentingnya menjaga kelangsungan ekosistem laut. Eksploitasi yang tidak bertanggung jawab dan cenderung merusak mengakibatkan hanya sekitar 5% dari terumbu karang di Indonesia yang masih terjaga [1].

Senada dengan lautnya, sebagian besar sungai di Indonesia juga mengalami kerusakan ekosistem akibat pencemaran. Pencemaran tersebut diantaranya bersumber dari limbah industri dan rumah tangga [2]. Pada Kota Surabaya pencemaran air sungainya berada pada level mengkhawatirkan. Sebanyak 62% pencemaran air sungai di Surabaya berasal dari limbah rumah tangga dan sisanya 38% berasal dari limbah industri [3]. Kurangnya kesadaran warga, lemahnya pengawasan pemerintah dan keengganan mereka untuk melakukan penegakan hukum yang benar menjadikan masalah pencemaran sungai menjadi hal yang semakin mengkhawatirkan.

Air merupakan kebutuhan hajat hidup orang banyak sehingga perlu untuk dilindungi agar dapat bermanfaat. Perlu adanya upaya untuk melestarikan, mengendalikan, mengawasi, untuk menjaga kualitas air atau mencapai kualitas air agar tetap dapat dimanfaatkan secara berkelanjutan. Salah satu cara menjaga kualitas air dengan upaya memonitoring kondisi air secara terus menerus yang dapat digunakan oleh pihak terkait untuk mendapatkan penanganan lebih lanjut.

Dalam upaya untuk mengatasi permasalahan tersebut, telah dibangun sistem

SEMAR (*Smart Environment Monitoring and Analytics in Real-time system*)
dimana sistem ini terdiri dari:

Gambar 1.1. SEMAR system

- Sistem monitoring kualitas air menggunakan ROV yang dilengkapi dengan sensor kualitas air untuk mengetahui kondisi air pada sungai. Sistem ini mampu dikendalikan secara jarak jauh dan mampu mengirim data ke pusat data melalui komunikasi kabel dan atau nirkabel.
- Jaringan nirkabel *mesh* yang diadopsi untuk memperluas jangkauan komunikasi antara sensor dan server.
- Sistem monitoring kondisi air portabel dan berbiaya murah.
- Sistem monitoring terumbu karang pada perairan dangkal.
- Pusat data yang menggunakan teknologi *Big Data*.

Pada sistem SEMAR, digunakan teknologi *Big Data* yang berfungsi sebagai pusat data yang berasal dari sensor-sensor yang digunakan. Penggunaan teknologi *Big Data* digunakan untuk mendukung proses monitoring yang dilakukan secara terus menerus dan *realtime* yang akan menghasilkan data yang semakin lama semakin besar. Namun, hingga saat ini SEMAR belum memiliki sistem analisa. Dalam perkembangannya, SEMAR sebagai platform monitoring membutuhkan sebuah sistem yang dapat menganalisa data-data yang diterima oleh server secara *realtime* sebelum disimpan ke dalam *Big Data*. Sekaligus juga sebuah sistem yang dapat digunakan untuk melakukan analisa terhadap data-data yang telah tersimpan di dalam *Big Data*. Sehingga kami melakukan penelitian untuk membangun ekstensi SEMAR untuk melakukan analisa data secara *realtime*. Dengan adanya sistem analisa data air yang *realtime*, data air yang masuk tidak hanya akan sekedar disimpan pada *Big Data*, namun juga dapat menghasilkan informasi baru mengenai kondisi air terkini.

1.2 PERMASALAHAN

Kebutuhan akan sistem monitoring air sungai yang terintegrasi dengan kapasitas penyimpanan yang besar dan mampu melakukan klasifikasi kualitas air secara *realtime* dan akurat sangat diperlukan. Hal ini menjadi landasan sistem SEMAR dibangun dengan mengintegrasikan teknologi *Internet of Things* (IoT) dan *Big Data* dalam melakukan monitoring dan analisa kondisi lingkungan air di Surabaya. Dalam sistem ini, perlu adanya kesinambungan antara node dalam mengirim data, server dalam menerima, analisa, dan visualisasi data tersebut untuk mendukung proses *realtime*. Karena itu dalam penelitian ini akan difokuskan bagaimana penerapan sistem terintegrasi *Internet of Things* (IoT) dan *Big Data* sehingga mampu melakukan pengiriman dan klasifikasi kualitas air secara *realtime* dan akurat. Sehingga permasalahan dalam penelitian ini adalah: Bagaimana membangun sistem pengiriman dan analisa kualitas air sungai secara *realtime* dan akurat?

1.3 TUJUAN

Tujuan dari penelitian tesis ini mendesain dan mengimplementasikan *IoT-Big Data analytic* pada sistem SEMAR yang digunakan untuk melakukan analisa data air secara *realtime* dan akurat. Pengklasifikasian air menggunakan metode Indeks Pencemaran yang membagi tingkat status air ke dalam beberapa kriteria. Hasil dari pengklasifikasian kualitas air pada kelas tertentu dapat digunakan untuk membantu memberikan informasi kepada pemerintah kota Surabaya. Informasi tersebut berupa kondisi air secara *realtime* berdasar pada sensor yang digunakan. Informasi selanjutnya adalah klasifikasi kualitas air yang di monitoring berdasarkan pada metode Indeks Pencemaran (IP).

1.4 MANFAAT

Membangun ekstensi klasifikasi *realtime* berbasis *IoT-Big Data analytic* pada sistem SEMAR. Sistem ini akan memberikan manfaat akan pengimplementasian dari teknologi *Internet of Things* dan teknologi *Big Data* dalam membangun platform monitoring, penyimpanan data *realtime* serta analisa kondisi sebuah lingkungan, utamanya dalam penelitian ini adalah lingkungan air sungai di Kota Surabaya.

1.5 SISTEMATIKA PENULISAN

Adapun sistematika pembahasan dari buku tesis ini meliputi:

Bab 1 Pendahuluan

Bab ini terdiri dari latar belakang, rumusan masalah, tujuan, dan manfaat penelitian. Di latar belakang diuraikan pada hal-hal yang mendasari peneliti untuk melakukan penelitian, penyusunan masalah dalam rumusan masalah, dan tujuan yang ingin dicapai dalam penelitian serta manfaat penelitian.

Bab 2 Kajian Pustaka

Bab ini membahas secara lebih rinci kontribusi penelitian serupa sebelumnya yang telah dilakukan untuk orang lain. Bab ini juga mencakup teori-teori yang berkaitan dengan penyelesaian tesis yang diperoleh dari berbagai buku dan sumber informasi lainnya yang terkait yang berhubungan dengan pembuatan tesis.

Bab 3 Desain Sistem

Bab ini berisi tentang desain sistem yang dibuat termasuk desain aplikasi, desain komunikasi, desain server, desain protokol implementasi, desain skenario implementasi dan skenario pengujian, serta spesifikasi perangkat keras dan perangkat lunak yang digunakan.

Bab 4 Eksperimen dan Analisis

Bab ini berisi tentang parameter pengujian, skenario uji coba, tempat dan waktu eksperimen, spesifikasi alat uji coba. Selain itu dibahas beberapa pengujian yang dilakukan dan menganalisa data pengujian yang sudah didapatkan

Bab 5 Penutup

Bab ini berisi kesimpulan dari hasil eksperimen yang telah dilakukan, baik hasil numerik maupun secara analisis serta saran yang dapat menunjang dalam pengembangan dari tesis ini.

BAB 2

KAJIAN PUSTAKA

2.1 TEORI PENUNJANG

Dalam penelitian ini dibutuhkan beberapa teori penunjang yang digunakan sebagai landasan dalam mengeksplorasi topik penelitian yang dilakukan. Beberapa topik pembahasan yang digunakan dalam penelitian ini meliputi: Sumber Daya Air, Parameter Mutu Air, Metode Indeks Pencemaran, Teknologi *Internet of Things* dan *Big Data*.

2.1.1 Sumber Daya Air

Berdasarkan Undang-Undang Republik Indonesia mengenai Sumber Daya Air No.7 Tahun 2004 menyebutkan bahwa sumber daya air adalah air, sumber air, dan daya air yang terkandung di dalamnya. Air adalah semua air yang terdapat pada, di atas, ataupun di bawah permukaan tanah, termasuk dalam pengertian ini air permukaan, air tanah, air hujan dan air laut yang berada di darat. Sumber Air adalah tempat atau wadah air alami dan/atau buatan yang terdapat pada, di atas, ataupun di bawah permukaan tanah. Daya air adalah potensi yang terkandung dalam air dan/atau pada sumber air yang dapat memberikan manfaat ataupun kerugian bagi kehidupan dan penghidupan manusia serta lingkungannya [4].

Beberapa sumber air dapat diperoleh dari lokasi yang berbeda-beda dan untuk pemanfaatan yang berbeda pula seperti yang tertuang pada acuan Peraturan Pemerintah Republik Indonesia nomor 82 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air tahun 2001, kemudian dilanjutkan dengan Peraturan Daerah Provinsi Jawa Timur nomor 2 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air di Provinsi Jawa Timur tahun 2008, dan Surat Keputusan Peraturan Daerah Kota Surabaya tentang Pengelolaan Kualitas Air dan

Pengendalian Pencemaran Air No. 2 di Jawa Timur Tahun 2004. Penggolongan kriteria air dibagi menjadi empat kelas [5],[6],[7] antara lain :

- a. Kelas satu, air yang peruntukannya dapat digunakan untuk air baku air minum, dan atau peruntukan lain yang mempersyaratkan mutu air yang sama dengan kegunaan tersebut,
- b. Kelas dua, air yang peruntukannya dapat digunakan untuk prasarana/sarana rekreasi air, pembudidayaan ikan air tawar, peternakan, air untuk mengairi pertanaman, dan atau peruntukan lain yang mempersyaratkan mutu air yang sama dengan kegunaan tersebut,
- c. Kelas tiga, air yang peruntukannya dapat digunakan untuk pembudidayaan ikan air tawar, peternakan, air untuk mengairi pertanaman, dan atau peruntukan lain yang mempersyaratkan mutu air yang sama dengan kegunaan tersebut.
- d. Kelas empat, air yang peruntukannya dapat digunakan untuk mengairi pertanaman atau peruntukan lain yang mempersyaratkan mutu air yang sama dengan kegunaan tersebut.

Pembagian penggolongan kategori air ke beberapa kelas di atas tergantung pada nilai kualitas air yang digambarkan pada beberapa parameter-parameter air. Pada penelitian ini digunakan standar baku berdasarkan Peraturan Daerah Kota Surabaya No 02 Tahun 2004, Peraturan Daerah Provinsi Jawa Timur nomor 2 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air di Provinsi Jawa Timur tahun 2008, Peraturan Pemerintah Republik Indonesia No 82 Tahun 2001 tentang peruntukan nilai baku kelas 1 dan untuk nilai kekeruhan menggunakan Peraturan Menteri Kesehatan No 492/Menkes/Per/IV/2010.

2.1.2 Parameter Mutu Air

Dalam menjaga kualitasnya perlu dilakukan pengelolaan kualitas air dengan tujuan untuk menjamin kualitas air yang diinginkan sesuai peruntukannya agar tetap dalam kondisi alamiahnya. Pengelolaan kualitas air adalah upaya pemeliharaan air sehingga tercapai kualitas air yang diinginkan sesuai peruntukannya untuk menjamin agar kualitas air tetap dalam kondisi alamiahnya

[5]. Kualitas air dapat diketahui dengan melakukan pengujian tertentu terhadap air tersebut. Pengujian yang dapat dilakukan menggunakan beberapa parameter yaitu parameter fisika, parameter kimia, parameter mikrobiologi dan parameter radioaktivitas.

Indikator yang sering kali digunakan dalam pemeriksaan pencemaran air adalah pH atau konsentrasi ion hidrogen, oksigen terlarut (*Dissolved Oxygen*, DO), kebutuhan oksigen biokimia (*Biochemiyal Oxygen Demand*, BOD) serta kebutuhan oksigen kimiawi (*Chemical Oxygen Demand*, COD). Namun pada penelitian yang digunakan penulis untuk pemeriksaan pencemaran air menggunakan beberapa parameter yang terdapat pada sensor air 'Atlas Scientific' yaitu *potential of Hydrogen* (pH), *Dissolved Oxygen* (DO), *Temperature*, *Electrical Conductivity* (EC), *Oxidation Reduction Potential* (ORP). Sensor EC telah dikembangkan untuk dapat mengkalkulasi nilai perkiraan dari *Total Dissolved Solids* (TDS), *Salinity*, dan *Specific Gravity* (SG). Sebagai tambahan dalam penelitian ini, penulis menggunakan beberapa sumber data air baku PDAM, yaitu: Sensor air PDAM dengan masa percobaan dari Maret hingga Agustus 2016, Data laporan laboratorium PDAM mulai tahun Januari 2014 hingga Oktober 2016. Untuk sensor air PDAM menggunakan area Ngagel dengan parameter input yang digunakan yaitu *temperature*, *Nephelometric Turbidity Unit* (NTU), *Total Suspended Solids* (TSS), pH, dan DO. Sedangkan data uji laboratorium PDAM mulai tahun Januari 2014 hingga Oktober 2016, menggunakan data laboratorium dengan hanya menggunakan 5 parameter dari 20 parameter yang ada, yaitu suhu, NTU, TSS, pH, dan DO.

Berikut penjelasan untuk masing-masing parameter yang akan digunakan peneliti dalam penelitian ini:

- a. Pengukuran pH, merupakan pengukuran konsentrasi ion Hidrogen pada air sungai. Pada range pH antara 6.5 – 8.5 menunjukkan kondisi alami sungai dalam keadaan belum tercemar. Kondisi sungai yang mengalami pencemaran air, apabila kondisi pH yang dihasilkan lebih rendah dari 6.5 atau lebih tinggi dari 8.5. pH dikatakan dalam kondisi asam apabila nilainya kurang dari 7. Sedangkan pH dikatakan dalam kondisi basa bila nilainya lebih dari 7. Perubahan kondisi air menjadi lebih asam, apabila terdapat kandungan bahan-

bahan organik. Begitu pula sebaliknya, kondisi air bisa menjadi basa apabila terdapat kandungan kapur. Penyebab terjadinya perubahan pada kondisi air sangat tergantung pada kandungan bahan pencemarnya. Terjadinya perubahan nilai pH air memiliki peranan yang penting bagi organisme air, sehingga nilai pH yang terlalu basa ataupun terlalu asam dapat menyebabkan pencemaran.

- b. DO, merupakan oksigen yang terlarut dalam air. Nilai DO biasanya diukur untuk menunjukkan jumlah oksigen dalam air. Semakin besar nilai DO pada airnya mengindikasikan air tersebut memiliki kualitas yang bagus. Sebaliknya jika nilai DO rendah, dapat diketahui bahwa air tersebut telah tercemar. Tanpa adanya oksigen yang terkandung didalamnya maka banyak microorganism dalam air yang tidak dapat hidup karena oksigen yang terlarut digunakan untuk proses degradasi senyawa organik dalam air. Oksigen terlarut akan menurun apabila banyak limbah, terutama limbah organik yang masuk ke sistem perairan.
- c. Suhu, digunakan sebagai indikator dikarenakan perubahan kecil pada temperature dapat mempercepat proses biologis tumbuhan dan hewan bahkan menentukan tingkat kejenuhan oksigen yang terkandung di dalamnya. Suhu air yang relative tinggi ditandai munculnya ikanikan dan hewan air ke permukaan untuk mencari oksigen. Dalam pengukuran DO tanpa adanya indikator suhu airnya maka kurang berguna, karena penurunan tingkat oksigen dihitung dari perbedaan tingkat kejenuhan dan DO terukur tidak dapat ditentukan karena suhu air tidak diketahui.
- d. TSS, residu tersuspensi (*Total Suspended Solid*) adalah semua zat padat (pasir, lumpur, dan tanah liat) atau partikel-partikel yang tersuspensi dalam air dan dapat berupa komponen hidup (biotik) seperti fitoplankton, zooplankton, bakteri, fungi, ataupun komponen mati (abiotik) seperti detritus dan partikel-partikel anorganik. Zat padat tersuspensi merupakan tempat berlangsungnya reaksi-reaksi kimia yang heterogen, dan berfungsi sebagai bahan pembentuk endapan yang paling awal dan dapat menghalangi kemampuan produksi zat organik di suatu perairan. Penetrasi cahaya matahari ke permukaan dan bagian yang lebih dalam tidak berlangsung efektif akibat

terhalang oleh zat padat tersuspensi, sehingga fotosintesis tidak berlangsung sempurna. Sebaran zat padat tersuspensi di laut antara lain dipengaruhi oleh masukan yang berasal dari darat melalui aliran sungai, ataupun dari udara dan perpindahan karena resuspensi endapan akibat pengikisan.

- e. NTU, Kekeruhan menggambarkan sifat optik air yang ditentukan berdasarkan banyaknya penyerapan cahaya dan pancaran cahaya yang berasal dari bahan-bahan yang terdapat di dalam air. Kekeruhan disebabkan oleh adanya bahan organik dan anorganik baik yang tersuspensi ataupun yang terlarut maupun bahan anorganik dan organik yang berupa plankton dan mikroorganisme lainnya. Pada suatu perairan, zat anorganik yang menyebabkan kekeruhan dapat berasal dari pelapukan batuan dan logam, sedangkan zat organik berasal dari lapukan hewan dan tumbuhan. Sedangkan untuk bakteri, dapat dikategorikan sebagai materi organik tersuspensi yang menambah kekeruhan air. Satuan kekeruhan dalam pengukuran nilai kekeruhan dengan pengukuran nephelometer dinyatakan dalam NTU. Pengukuran dengan nephelometer sering digunakan karena ketepatan, sensitifitas dan dapat digunakan dalam rentang turbiditas yang besar. Pada penelitian ini, untuk nilai baku kekeruhan yang digunakan berdasarkan Peraturan Menteri Kesehatan Nomor: 492/Menkes/Per/IV/2010 pada tanggal 19 April 2010, dimana untuk nilai baku yang disarankan pada kekeruhan sebesar 5 [8].

2.1.3 Metode Indeks Pencemaran

Indeks pencemaran (*Pollution Index*) digunakan untuk menentukan tingkat pencemaran relatif terhadap parameter yang diijinkan [9]. Indeks Pencemaran (IP) memiliki konsep yang berlainan dengan Indeks Kualitas Air (*Water Quality Index*). Indeks Pencemaran (IP) ditentukan untuk suatu peruntukan, kemudian dapat dikembangkan untuk beberapa peruntukan bagi seluruh bagian badan air atau sebagian dari suatu sungai.

Perhitungan penentuan status mutu air menggunakan metode Indeks Pencemaran diatur pada Keputusan Menteri Negara Lingkungan Hidup No 115 Tahun 2003 tentang Pedoman Penentuan Status Mutu Air [10].

Pengelolaan kualitas air atas dasar Indeks Pencemaran (IP) dapat memberikan masukan pada pengambil keputusan agar dapat menilai kualitas badan air untuk suatu peruntukan. Kemudian melakukan tindakan untuk memperbaiki kualitas jika terjadi penurunan kualitas akibat kehadiran senyawa pencemar.

Jika L_{ij} merupakan konsentrasi parameter kualitas air yang dicantumkan dalam baku peruntukan air (j), dan C_i dinyatakan sebagai konsentrasi parameter kualitas air (i) yang diperoleh dari hasil analisis dari suatu lokasi pengambilan dari suatu alur sungai, maka IP_j adalah Indeks Pencemaran bagi peruntukan (j) yang merupakan fungsi dari $\frac{C_i}{L_{ij}}$ dan ditentukan dari resultan nilai maksimum (M) dan nilai rerata (R) rasio konsentrai perparamter terhadap nilai baku mutunya. Metode Indeks Pencemaran [10] dapat dihitung menggunakan formula 2.1.

$$IP_j = \sqrt{\frac{(C_i/L_{ij})M^2 + (C_i/L_{ij})R^2}{2}} \quad (2.1)$$

Evaluasi terhadap nilai IP_j dapat menentukan kategori kelas Indeks Pencemaran:

- a. $0 \leq IP \leq 1.0$, memenuhi baku mutu (kondisi baik)
- b. $1.0 \leq IP \leq 5.0$, tercemar ringan
- c. $5.0 \leq IP \leq 10.0$, tercemar sedang
- d. $IP \geq 10.0$, tercemar berat

Pada penelitian ini, penulis menggunakan pembagian kelas berdasarkan pada hasil perhitungan Indeks Pencemaran. Dari pembagian kelas Indeks Pencemaran tersebut akan ditentukan label dari masing-masing kelas. Pelabelan dibutuhkan dalam proses *learning* dari dataset yang digunakan. Dimana label akan menunjukkan kategori indeks pencemaran dari sampel air/ data air yang diterima oleh sistem secara *realtime*. Implementasi skor, keterangan dan label Indeks Pencemaran dapat dilihat pada tabel 2.1. Kategori indeks pencemaran yang terdiri dari 4 kategori ditranslasikan ke dalam 4 label yang masing-masing diwakili oleh angka 0, 1, 2 dan 3.

Tabel 2.1. Label Indeks Pencemaran

Score	Keterangan	Label
$0 \leq IP \leq 1.0$	memenuhi baku mutu (kondisi baik)	0
$1.0 \leq IP \leq 5.0$	tercemar ringan	1
$5.0 \leq IP \leq 10.0$	tercemar sedang	2
$IP \geq 10.0$	Tercemar berat	3

Dari tabel 2.1 diperoleh empat label kelas yang memungkinkan pada dataset yang akan digunakan. Pada label digunakan data numerical untuk memudahkan proses *learning*. Label 0 untuk data yang memenuhi baku mutu, label 1 untuk data yang tercemar ringan, label 2 untuk data yang tercemar sedang, dan label 4 untuk data yang tercemar berat. Nantinya hasil klasifikasi *realtime* akan ditranslasikan kembali ke data kategorikal sesuai dengan keterangan status indeks pencemaran.

2.1.4 Teknologi *Internet of Things* & *Big Data*

Perkembangan *Internet of Things* (IoT), tidak lepas dari konsep dasarnya yaitu untuk menghubungkan obyek yang satu dengan obyek yang lainnya (*Things*) secara bersama-sama, sehingga memungkinkan untuk saling berkomunikasi satu sama lain. Menurut definisi, IoT digambarkan sebagai:

“a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies (ICT)”. [11]

Melalui penerapan konsep teknologi IoT dengan interkoneksi antara dunia fisik dengan dunia maya, baik melalui eksploitasi identifikasi, pengambilan data, pengolahan data, dan kemampuan dalam berkomunikasi, membuka peluang baru dalam dimensi IoT untuk mengakses apapun, setiap saat dan dari tempat manapun. Seperti yang digambarkan pada Gambar 2.1 mengenai dimensi IoT menurut ITU-T Y.2060.

Sumber: Recommendation ITU-T Y.2060, 2012 [11]

Gambar 2.1. Dimensi IoT menurut ITU-T Y.2060

Berikut adalah karakteristik dasar dari IoT, yang memberikan gambaran yang lebih jelas mengenai *Internet of Things*:

- Interkonektivitas, di dalam IoT, apapun (*things*) bisa saling berhubungan dengan informasi global dan juga infrastruktur komunikasi.
- Layanan yang berkaitan pada objek (*things*), dimana IoT mampu memberikan layanan terkait dengan objek-objek yang saling berhubungan termasuk dalam hal keterbatasan seperti perlindungan privasi maupun konsistensi semantik baik teknologi informasi di dunia fisik maupun dunia maya.
- Heterogenitas, meliputi interaksi perangkat IoT yang heterogen karena berdasarkan platform hardware yang berbeda dan jaringan yang digunakan.
- Perubahan dinamis, dimana keadaan perangkat berubah secara dinamis misalkan saat perangkat terhubung atau terputus maupun kondisi “*sleep*” serta kondisi “*waking up*”. Selain itu, dapat ditinjau dari jumlah perangkat dapat berubah secara dinamis.
- Skala “*Enormous*”: Jumlah perangkat yang perlu dikelola dan berkomunikasi satu sama lainnya dapat berkembang lebih besar dengan perangkat yang terhubung ke internet saat ini. Bahkan yang lebih penting, pengelolaan data yang dihasilkan dan interpretasi untuk tujuan aplikasi [11].

ITU-T Y 2060, juga telah mendefinisikan model referensi untuk IoT, seperti yang digambarkan pada Gambar 2.2 mengenai model referensi dari IoT.

Sumber: Recommendation ITU-T Y.2060, 2012 [11]

Gambar 2.2. Model referensi dari IoT menurut ITU-T Y.2060

Selanjutnya model referensi dari *Internet of Things* dikembangkan oleh Cisco dengan menerbitkan Cisco White Paper mengenai *The Internet of Things Reference Model* pada tahun 2014. Dalam dokumen tersebut di definisikan peningkatan referensi dari IoT dengan 7 lapisan [12], sebagaimana digambarkan pada gambar 2.3.

Internet of Things Reference Model

Sumber: Cisco, 2014.

Gambar 2.3. Model referensi dari IoT menurut Cisco

Model referensi dari *Internet of Things* yang digunakan dalam sistem ini terbagi menjadi 7 lapisan sebagaimana yang dirumuskan oleh Cisco, yaitu:

a. *Physical Devices and Controllers*

Lapisan ini merupakan perangkat fisik dan pengendali yang merupakan perangkat *endpoint* yang mengirim dan menerima informasi. Saat ini perangkat pada lapisan ini sudah sangat banyak dan akan senantiasa meningkat seiring dengan tren IoT yang semakin meningkat.

1) Sensor Air

Dalam penelitian ini digunakan sensor air sebagai perangkat untuk melakukan monitoring air sungai secara *realtime*. Perangkat sensor air yang digunakan adalah ‘Atlas Scientific’. Sensor ‘Atlas Scientific’ terdiri dari sensor *Potential of Hydrogen (pH)*, *Oxidation Reduction Potential (ORP)*, *Dissolved Oxygen (DO)*, *Electrical Conductivity (EC)*, dan *Temperature*.

a) *Potential of Hydrogen (pH)*

pH merupakan sebuah larutan untuk takaran dari konsentrasi molar pada ion-ion hidrogen di dalam larutan dan begitu pun untuk takaran dari sebuah *acidity* atau *basicity* dari sebuah larutan. Satuan untuk pH dan nilai di definisikan minus 10 logaritma dari konsentrasi molar pada ion – ion hydrogen sebagaimana pada formula 2.2.

$$pH = -\log_{10}[H^+] \quad (2.2)$$

Pengukuran pH dari sebuah sampel dapat diselesaikan oleh pengukuran potensial *cell* dari sampel tersebut ke dalam standar referensi elektroda hidrogen, semua itu merupakan prosedur yang diterima untuk standar pengukuran potensial elektroda. Prosedur ini akan di peroleh sebuah nilai dari 0 untuk sebuah 1 larutan

molar pada ion-ion H^+ , jadi semua itu definisi 0 dari skala pH. Potensial *cell* untuk nilai yang lain dari konsentrasi H^+ dapat di *obtain* dengan menggunakan Nerst equation. Untuk sebuah larutan pada suhu $25^\circ C$ seperti pada formula 2.3.

$$E_{cell} = -0.0592 \log_{10}[H^+] \quad \text{atau} \quad \frac{pH=E_{cell}}{0.0592} \quad (2.3)$$

Untuk formula 2.3, merupakan sebuah perubahan dasar dari log dasar pada 10 algoritma yang telah dibuat dalam persamaan nerst [13].

Dalam praktiknya, pH biasanya tidak diukur dengan cara ini, karena memerlukan gas hidrogen pada tekanan standar, dan platinum elektroda digunakan di dalam standar elektroda hidrogen yang mudah dicemari oleh pengaruh bahan-bahan lain di dalam sebuah larutan. Untungnya, konfigurasi elektroda yang lain dapat dikalibrasi untuk membaca konsentrasi ion H^+ . pH meter sering dibuat dengan sebuah elektroda kaca yang terdiri dari perak dilapisi kawat dengan perak klorida direndam di dalam larutan asam klorida. Larutan elektroda yang dipisahkan dari larutan untuk diukur dengan membran kaca yang tipis. Potensi yang berkembang di seluruh membran kaca cukup proporsional yang dapat memperlihatkan konsentrasi ion hidrogen pada dua permukaan. Di dalam instrumen pengukuran, *cell* dibuat dengan elektroda lain yang umum untuk menjadi merkuri-merkuri elektroda klorida. Potensi *cell* yang linier cukup proporsional untuk pH dan alat ukur dapat dikalibrasi untuk membaca pH secara langsung. Gambar 2.4 memperlihatkan bentuk struktur pH sensor.

Sumber: Nave, 2016 [13]

Gambar 2.4. Struktur pH meter

b) *Dissolved Oxygen (DO)*

Oksigen terlarut (*dissolved oxygen*, disingkat DO) atau sering juga disebut dengan kebutuhan oksigen (*oxygen demand*) merupakan salah satu parameter penting dalam analisis kualitas air [14]. Nilai DO yang biasanya diukur dalam bentuk konsentrasi yang menunjukkan jumlah oksigen (O₂) yang tersedia dalam suatu badan air. Semakin besar nilai DO pada air, mengindikasikan air tersebut memiliki kualitas yang bagus. Sebaliknya jika nilai DO rendah, dapat diketahui bahwa air tersebut telah tercemar. Pengukuran DO juga bertujuan melihat sejauh mana badan air mampu menampung biota air seperti ikan dan mikroorganisme. Selain itu kemampuan air untuk membersihkan pencemaran juga ditentukan oleh banyaknya oksigen dalam air. Oleh sebab itu pengukuran parameter ini sangat dianjurkan di samping parameter lain seperti kob dan kod.

Tipikal level DO, konsentrasi DO berdampak secara konstan oleh difusi dan aerasi, fotosintesis, respirasi dan dekomposisi. Sementara kesetimbangan air terhadap 100% kejenuhan udara, tingkat oksigen yang terlarut juga akan berubah-ubah dengan suhu, salinitas dan perubahan tekanan. Dengan demikian, tingkat oksigen yang terlarut dapat berkisar dari kurang dari 1 mg/l hingga lebih dari 20 mg/L tergantung pada bagaimana semua dari faktor-faktor ini berinteraksi. Dalam sistem air tawar seperti danau, sungai dan aliran air, konsentrasi oksigen yang terlarut akan berbeda-beda oleh lokasi dan kedalaman air [15], [16].

Sumber: Kemker, 2016 [15]

Gambar 2.5. Konsentrasi DO pada berbagai musim

Kelarutan oksigen (DO) menurun seiring meningkatnya suhu. DO akan berkurang secara eksponensial seiring meningkatnya kadar garam (salinitas). Dan jika tekanan meningkat maka DO juga akan meningkat [17]. Itulah mengapa pada suhu dan tekanan yang sama air asin menampung 20% lebih sedikit DO daripada air tawar [18].

Sumber: Kemker, 2016 [18]

Gambar 2.6. Perbandingan *Oxygen Content* dengan *Temperature*

Sumber: Kemker, 2016[18]

Gambar 2.7. Perbandingan *Oxygen Content* dengan dengan *Altitude*

Cara penentuan oksigen terlarut dengan metoda elektrokimia adalah cara langsung untuk menentukan oksigen terlarut dengan alat DO meter. Prinsip kerjanya adalah menggunakan probe oksigen yang terdiri dari katoda dan anoda yang direndam dalam larutan elektrolit. Pada alat DO meter, probe ini biasanya menggunakan katoda perak (Ag) dan anoda timbal (Pb). Secara keseluruhan,

elektroda ini dilapisi dengan membran plastik yang bersifat semi permeable terhadap oksigen. Reaksi kimia yang akan terjadi berupa reaksi katoda dan anoda sebagaimana pada persamaan 2.4.

c) *Oxidation Reduction Potential (ORP)*

Hasil penelitian teliti di *Recirculating Aquaculture systems (RAS-C)* menunjukkan adanya dampak jangka panjang yang disebabkan oleh pengaruh dari *Oxidation Reduction Potential (ORP)* terhadap kehidupan biota laut yang di akibatkan oleh infeksi bakteri, memunculkan pentingnya memonitoring ORP pada air [18].

Adanya parameter ORP dilatarbelakangi oleh kenyataan bahwa suatu sistem, dalam hal ini larutan, mengalami proses oksidasi sehingga terjadi perubahan yang terus-menerus dari perbandingan (rasio) antara bentuk teroksidasi dan tereduksi. Potensi reduksi mempengaruhi proses kimia yang terjadi diperairan. Pada penentuan potensi reduksi, parameter yang diukur adalah elektroda hidrogen. ORP dinyatakan dengan persamaan 2.5.

$$E = E_0 - x \log 10 \quad (2.5)$$

Keterangan:

E_0 = potensial oksidasi

E = jumlah elektron pada proses reaksi

Reaksi yang terjadi pada kondisi aerob memiliki nilai ORP > 200 mV, sedangkan reaksi pada kondisi anaerob memiliki nilai ORP < 50 mV. Perairan dengan kadar oksigen jenuh pH 7 dan kondisi suhu 25° C memiliki nilai ORP 0,80 volt. Perairan alami biasanya memiliki nilai ORP berkisar antara 0,45 – 0,52 volt.

Nilai ORP sedikit dipengaruhi oleh suhu, namun sangat dipengaruhi oleh kadar oksigen. Pada lapisan hipolimnium, nilai ORP dapat mencapai nol. Pada

lumpur dasar perairan yang memiliki kondisi anaerob, nilai ORP dapat mencapai - 0,1 volt.

Potensi redoks (reduksi dan oksidasi) atau *oxidation reduction potential* (ORP) yang menggambarkan aktivitas elektron (e) di perairan adalah potensi larutan untuk mentransfer elektron dari suatu oksidan kepada reduktan. Suatu bahan dikatakan mengalami oksidasi jika kehilangan elektron dan dikatakan mengalami reduksi jika menerima elektron. Adapun contoh proses oksidasi reduksi ditunjukkan dalam persamaan 2.6.

d) *Electrical Conductivity* (EC)

Secara umum, sumber air tawar memiliki konduktivitas yang lebih rendah dibandingkan dengan konduktivitas air laut. Sejauh ini, tidak ada aturan secara global mengenai standar konduktivitas pada air. Namun, beberapa organisasi dan wilayah tertentu telah menentukan batasan pada total *dissolved solid* untuk tubuh dan air. Hal ini dikarenakan konduktivitas dan salinitas dapat berbeda-beda tidak hanya di antara lautan dan air tawar, tetapi juga di setiap sungai yang berdekatan. Jika geologi di sekitarnya yang cukup berbeda, atau jika salah satu sumber memiliki arus yang terpisah, nilai konduktivitas pada kandungan di setiap aliran air sungai akan bernilai berbeda. Meskipun tidak adanya standar dan efek dari lingkungan sekitar pada konduktivitas, terdapat kira-kira nilai yang dapat diharapkan berdasarkan sumber air [19].

uS/cm	
DISTILLED WATER	0.5 - 3
MELTED SNOW	2 - 42
TAP WATER	50 - 800
POTABLE WATER IN THE US	30 - 1500
FRESHWATER STREAMS	100 - 2000
INDUSTRIAL WASTEWATER	10000
SEAWATER	55000

Sumber: Kemker, 2016 [19]

Gambar 2.8. Nilai EC pada beberapa kasus

ppt

FRESHWATER	< 0.5
BRACKISH / ESTUARY	0.5 - 17
BLACK SEA	16
OCEAN RANGE	32 - 37
OCEAN AVERAGE	35

Sumber: Kemker, 2016 [19]

Gambar 2. 9. Perkiraan nilai salinitas pada perairan

e) *Temperature*

Sebagai satuan suhu terkait energi, skala yang dikembangkan untuk memberi nilai suhu yang relatif ke beberapa nilai yang ada. Sekarang ini suhu dibedakan menjadi tiga yakni celcius, fahrenheit dan kelvin. Yang sering digunakan sebagai standar suhu adalah skala celcius.

(Sumber: http://www.fondriest.com/environmental-measurements/wp-content/uploads/2015/03/temperature_realtime.jpg)

Gambar 2.10. Skala suhu

Suhu pada derajat celcius dapat dikonversi ke dalam *Fahrenheit* menggunakan formula 2.7.

$$^{\circ}\text{F} = (1.8 \times ^{\circ}\text{C}) + 32 \quad (2.7)$$

Sementara untuk memperoleh data suhu dengan skala kelvin dapat dikonversi dengan menggunakan formula 2.8.

$$K = ^\circ C + 273.15 \quad (2.8)$$

Dengan formula 2.7 dan 2.8 dapat dengan mudah diperoleh hasil nilai yang dapat disesuaikan dengan kebutuhan output yang diharapkan.

2) *Embedded System*

Sistem benam yang digunakan untuk penelitian ini adalah Raspberry Pi versi 3 model B [20]. Pada sistem benam ini diinstall sistem operasi Raspbian yang merupakan Linux distro Debian yang telah dimodifikasi untuk dapat berjalan pada sistem benam. Sistem benam ini akan menjadi *controller* bagi *physical devices*, dalam hal ini adalah sensor air. Pada sistem benam ini akan dibangun sebuah aplikasi untuk menerima data dari sensor, mengirim data tersebut ke server, dan menyimpan data tersebut ke media penyimpanan lokal sebagai backup data. Aplikasi yang dibangun pada sistem benam ini menggunakan Bahasa pemrograman python dengan menggunakan beberapa library python yaitu: 1) *pySerial*, untuk komunikasi serial antara sistem benam dan mikrokontroller dalam mengirim dan menerima data sensor air; 2) *time*, untuk mengambil waktu sistem yang akan dikirim bersamaan dengan data air yang digunakan sebagai penunjuk waktu data diambil; dan 3) *paho-mqtt-client*, sebagai *library* pengiriman data dari sistem benam ke server menggunakan protokol MQTT. Dalam skema ini aplikasi pada sistem benam akan berperan sebagai MQTT *Publisher*. Sumber daya pada siste benam menggunakan power bank yang dapat mengisi daya sambil mengeluarkan arus ke raspberry. Kabel yang menghubungkan sensor dan sistem benam adalah USB-to-Serial. Sistem benam ini memiliki gambar seperti yang terlihat pada Gambar 2.11. Sedangkan untuk spesifikasi Raspberry Pi 3 model B yang digunakan dijelaskan pada Tabel 2.2.

(Sumber: https://www.element14.com/community/dtss-images/uploads/devtool/diagram/large/Raspberry_Pi_3_Starter_Kit_.png)

Gambar 2.11. Raspberry Pi 3 model B

Tabel 2.2. Spesifikasi Raspberry Pi 3 model B

Spesifikasi	
Prosesor	A 1.2 GHz Quad Core ARMv8
RAM	1 GB
Catu Daya	5 V
Sistem Operasi	Raspbian
Bahasa Pemrograman	Python
Interface	
USB	4 port
GPIO	40 pin (UART)
HDMI	1
Ethernet	100baseTx
Audio	3.5 mm jack output
Camera Interface (CSI)	1
Display Interface	1
Micro SD Slot	1
VGA	VideoCore IV 3D

(Sumber: <https://www.raspberrypi.org/?s=raspberry+pi+3+model+b>)

b. *Connectivity*

Komunikasi data yang digunakan untuk menghubungkan perangkat *Internet of Things* dengan environment server *Big Data* adalah komunikasi 4G LTE. Dimana perangkat yang akan digunakan adalah USB Modem 4G/LTE DT-100 Advance Jetz Plus Soft AP [21] seperti yang digambarkan pada Gambar 2.12. Sedangkan untuk spesifikasi USB Modem 4G/LTE DT-100 Plus + Soft AP dijelaskan pada Tabel 2.3.

(Sumber: <https://alnect.net/product/7633/Page-Modem-GSM-4GLTE-Advance-Jetz-DT100-Plus-Soft-AP>)

Gambar 2.12. USB Modem 4G/LTE DT-100 Plus + Soft AP

Tabel 2.3. Spesifikasi USB Modem 4G/LTE DT-100 Plus + Soft AP

Spesifikasi	
-	USB Dongle Modem dan menggunakan USB 2.0 Plug and Play
-	Mendukung data jaringan LTE/DC-HSPA+/HSUPA/HSDPA/UMTS/EDGE/GPRS
-	Mendukung internet data downlink dengan kecepatan sampai 100 Mbps (4G/LTE/WIMAX) dan kecepatan upload sampai 50Mbps
-	Sudah terintegrasi soft AP (Wifi Tethering)
-	MicroSD (TF-Card) slot untuk penyimpanan data
-	Mendukung sampai 5 koneksi pengguna dalam waktu bersamaan
-	Mendukung OS PC Windows 2000/2003/XP/Vista/7/8/10 dan Mac OS X v10.4 ke atas, serta Linux.

(Sumber: <https://alnect.net/product/7633/Page-Modem-GSM-4GLTE-Advance-Jetz-DT100-Plus-Soft-AP>)

Data yang diambil dari sensor perlu dikirim ke data center melalui komunikasi data 4G menggunakan internet. Protokol yang digunakan untuk mengirimkan data sensor menuju ke data center menggunakan protokol MQTT.

c. *Edge Computing*

Untuk pentransferan data yang digunakan dalam sistem ini sehingga data yang ditransfer cukup efisien, menggunakan protokol MQTT (*Message Queue Telemetry Transport Protocol*) [22]. Protokol MQTT adalah protokol jaringan yang menggunakan konsep *publisher/subscriber* dalam pengiriman datanya, biasanya untuk pengiriman pesan antar perangkat IoT. Pada mekanismenya, konsep yang menggunakan cara *publisher/subscribe* pesan dengan menerapkan topik yang sama di dalamnya, sekilas prinsipnya menyerupai client-server. MQTT ini sebenarnya diterapkan pada lintas *stack* protokol TCP/IP yang memiliki ukuran paket data dengan *header*-nya lebih kecil sehingga sumber daya yang diperlukan relatif kecil. *Header* pada MQTT lebih sederhana dibandingkan protokol TCP/IP pada penggunaan HTTP. Jenis data yang dikirimkan menggunakan protokol MQTT ini bisa berupa data *binary*, teks, bahkan XML. Salah satu platform yang mengimplementasikan MQTT adalah Mosquitto [23], platform ini yang nantinya digunakan sebagai MQTT broker.

MQTT menggunakan topik dalam mem-*publish* ataupun men-*subscribe* pada pengiriman pesan. Dikarenakan terhubung pada berbagai perangkat, maka komunikasi yang terjadi antar perangkat dilakukan dengan pengiriman pesan, dimana setiap pesan selalu memiliki topik yang nantinya digunakan sebagai kata kunci yang berupa string atau autentifikasi user, layaknya password. Sama halnya dengan paradigma client dan server, dalam komunikasi di dalam jaringan pada MQTT digunakan istilah MQTT *Publisher* dan MQTT *Subscriber* sebagai perangkat-perangkat yang ingin berkomunikasi satu sama lain, istilah dalam jaringan sebagai client nya. Sedangkan MQTT Broker bertindak sebagai pihak yang mengatur dan meneruskan pesan-pesan yang diterima bahkan bisa mempertahankan pesan pada setiap topik yang dikirimkan oleh MQTT *Publisher* [24]. Alur pengiriman pesan topik pada MQTT digambarkan pada Gambar 2.13.

Gambar 2.13. Alur Pengiriman Pesan Topik pada MQTT

Pada gambar di atas terlihat Publisher1 ketika akan mengirimkan pesan tertentu menggunakan topik “X” dan ketika ada permintaan dari *Subscriber* untuk mengakses pesan pada Publisher1 harus mengetahui topik tertentu dalam hal ini topik “X” sehingga yang mendapatkan pesan dari publisher1 hanyalah Subscriber1 dan Subscriber2 yang memiliki topik sama yaitu topik “X” sedangkan untuk *subscriber* yang tidak memiliki topik sama tidak mendapatkan akses kepada pesan tersebut.

MQTT *Publisher* memiliki peran yang memberikan suatu pesan kepada topik tertentu. Pada penelitian ini, perangkat Raspberry Pi 3 berperan sebagai MQTT *Publisher*, yang akan memberikan pesan berupa data sensor air kepada subscriber apabila memiliki topik yang sama. Dalam menggunakan MQTT ini, *publisher* menggunakan library paho-python. Pada penelitian ini telah dibuat aplikasi yang dapat mengambil data sensor kemudian mengirimkan dengan protokol MQTT, aplikasi ini diberi nama “watermonitoring”.

MQTT *Subscriber* dianalogikan sebagai client yang *men-subscribe* suatu topik, sehingga ketika *publiser* mengirimkan pesan dengan topik tertentu misalkan topik “watermonitoring”, maka *subscriber* yang memiliki topik yang sama akan menerima pesan tersebut. Perangkat yang berperan sebagai *subscriber* pada penelitian ini adalah aplikasi untuk penyimpanan data pada *Big Data* dan aplikasi untuk analisa *realtime*.

MQTT *Broker*, memiliki fungsi sebagai pihak yang mengatur sekaligus sebagai perantara yang menghubungkan antara *publisher* dengan *subscriber*, yang

selanjutnya akan meneruskan pesan dari *publisher* ke *subscriber*. Platform yang menggunakan protokol MQTT sangat banyak, namun yang digunakan sebagai MQTT *Broker* pada penelitian ini adalah Mosquitto-server.

Topik yang digunakan dalam MQTT penggunaannya hampir sama dengan pengiriman pesan pada *chatting broadcast* tetapi lebih sederhana dan memiliki fungsi sebagai autentifikasi user dalam pengiriman pesan agar MQTT *Broker* dengan mudah menghubungkan dan meneruskan pesan dari *publisher* ke *subscriber*.

d. *Data Accumulation*

Pada lapisan *Data Accumulation* digunakan teknologi *Big Data* untuk manajemen penyimpanan data yang berasal dari node sensor. Menurut Min Cen, dkk, 2014. Secara umum *Big Data* dapat diartikan sebagai sekumpulan data yang ditinjau dari ukurannya yang sangat besar (*volume*), sangat cepat perkembangan/pertumbuhannya (*velocity*), data yang beragam dalam berbagai bentuk/format (*variety*), serta memiliki nilai tertentu (*value*) [25].

(Sumber: <http://www.looiconsulting.com/wp-content/uploads/2014/08/Four-Vs.png>)

Gambar 2.14. Big Data 4V

Dalam infrastruktur *Big Data*, data akuisisi menunjukkan acuan aliran data yang memiliki kecepatan tinggi dan ragam yang bervariasi, infrastruktur *Big Data* mendukung akuisisi data yang besar sehingga apabila diproses dengan cepat dan

dalam lingkungan terdistribusi maka struktur data yang dinamis tersebut dapat menghasilkan prediksi yang lebih baik.

1) Apache Hadoop

Hadoop [26] menjadi sebuah pilihan alternatif dalam mengelola data yang cukup besar karena merupakan perangkat *open source* yang handal, terukur dan komputasi terdistribusi. Kepopuleran Hadoop dalam rentan beberapa tahun terakhir ini, menjadikan Hadoop sebagai model pemrograman sederhana pada *distributed parallel computing* dalam mengolah dataset yang besar. Hadoop merupakan implementasi *open source* yang mengadopsi dari Google-MapReduce karena dianggap memberikan banyak keuntungan yang signifikan dibandingkan database paralel. Sudah banyak perusahaan yang berhasil dalam mengembangkan perusahaannya melalui penggunaan Hadoop. Secara gambaran umum Apache Hadoop adalah sebuah perangkat lunak *open source* yang memungkinkan pemrosesan dataset yang besar secara terdistribusi dalam kluster server. Hadoop dirancang untuk dapat bekerja pada server tunggal hingga banyak server, dengan toleransi kesalahan yang sangat tinggi.

Hadoop mendukung pembuatan *software open-source* yang menyediakan sebuah framework untuk membangun aplikasi *distributed computing* dengan skalabilitas yang tinggi. Apache Hadoop memiliki dua komponen utama yaitu:

- a) MapReduce, framework yang membagi pekerjaan ke node-node dalam proses pengklasteringan Hadoop. MapReduce merupakan model programing untuk pengolahan data. MapReduce dibangun menggunakan konsep *divide and conquer*, dengan cara membagi pekerjaan yang besar menjadi bagian-bagian kecil dan memprosesnya secara paralel. Penanganan pada MapReduce meliputi pendistribusian data, pengkomputasian secara paralel serta otomatisasi dalam penanganan kegagalan. Dua bagian utama dalam MapReduce, antara lain pertama, proses Map. Pada proses Map terjadi pembacaan data dari sekumpulan *record* pada sebuah input file, kemudian menjalankan *filtering* dan transformasi serta memberikan output berupa sekumpulan data rekam menengah ke dalam bentuk pasangan nilainya.

Setelah itu nilai pada data rekam menengah menjadi inputan data pada proses kedua yaitu *Reduce*. Pada proses *Reduce* terjadi penerimaan data nilai rekam menengah sebagai input dan menggabungkan nilai-nilainya menjadi sebuah *key* yang utuh sehingga menjadi nilai ringkasan sebagai outputnya.

- b) HDFS, istilah singkat dari *Hadoop Distributed Filesystem*. Dimana sebuah file system digunakan untuk menyimpan file yang besar dengan membagi dan menyimpan lalu mendistribusikan ke dalam banyak node yang salaung berhubungan. HDFS merupakan perancangan filesystem yang menggunakan MapReduce dalam pembacaan data, mengolah dan menuliskan output keluarannya. Untuk menjaga reliabilitas, data dalam HDFS akan diduplikasi ke dalam beberapa node. Dua bagian utama dalam HDFS yaitu Namenode dan Datanode. Namenote digunakan dalam pengaturan metadata *filesystem*, sedangkan Datanode merupakan blok penyimpanan.

Di dalam Apache Hadoop terdapat beberapa aplikasi seperti yang digambarkan dalam Hadoop ekosistem, dimana beberapa diantaranya mendukung fungsi *Stream Processing* dan *Machine Learning* yang dapat diimplementasikan menggunakan Spark. Hal ini yang jika dikombinasikan dapat menghasilkan aplikasi analisa *realtime*. Hadoop ekosistem selengkapnya terlihat pada Gambar 2.15.

(Sumber: <https://biguru.files.wordpress.com/2014/04/cdh.png>)

Gambar 2.15. Hadoop Ekosistem

Setelah data disimpan tahap selanjutnya adalah penganalisaan data yang besar dan beraneka ragam pada lingkungan terdistribusi dengan menggunakan analisa data mining maupun analisa statistik. Ada dua jenis analisa dalam *Big Data* ditinjau dari pendekatan *machine learning* yaitu:

- a) Analisa penggambaran data yang tersimpan di *Big Data* dan menampilkan data tanpa memerlukan algoritma *Artificial Intelligence*.
- b) Analisa dalam pencarian informasi yang tersembunyi dari data yang tersimpan di *Big Data*. Pada analisa ini, tidak hanya menampilkan data yang lama namun menampilkan data yang baru yang dihasilkan dari proses analisa *Big Data*, dimana sudah menggunakan *Artificial Intelligence* atau *Machine Learning*. Sifat umumnya biasanya diikuti adanya proses *training* untuk meningkatkan keakurasian data.

Big Data menawarkan berbagai solusi yang biasa dikategorikan sebagai berikut:

- a) *Social data analysis* yang dikembangkan dalam pemrosesan data dari sosial media, yang bisa dikembangkan untuk sentiment analisis.
- b) *Historical data analysis*, misalkan pemrosesan analisa pada masa lalu sehingga bisa memberikan gambaran di masa lalu.
- c) *Prediktif analysis*, penggabungan data historical analysis dengan kombinasi *Artificial Intelligence* dalam memprediksi kejadian yang akan datang sehingga tindakan antisipasi dapat dilakukan.

2) Apache Hive

Apache hive [27] adalah tool lain untuk membentuk program MapReduce. Apache Hive pertama kali dikembangkan oleh Facebook untuk melakukan *data*

warehouse pada klaster Hadoop mereka yang sangat banyak. Selanjutnya Hive disumbangkan ke Apache Foundation untuk dikembangkan oleh komunitas *open source*. Hive lebih ditujukan untuk proses *data warehouse* di atas HDFS. Pada Apache Hive proses MapReduce dituliskan dengan gaya yang sangat mirip dengan SQL yang pada umumnya ada di RDBMS. Contoh Script Hive yang menghitung kemunculan huruf,

```

CREATE TABLE word_text(word STRING) COMMENT 'This is the word table'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '
LINES TERMINATED BY '\n';
LOAD DATA INPATH 'hdfs:/user/hue/word_count_text.txt' INTO TABLE
word_text;
SELECT word, count(*) as count FROM word_text GROUP BY word;

```

Seperti terlihat pada script Hive, tampak sekali kemiripan antara Script Hive dengan SQL. Script Hive di atas terdiri dari tiga statement yang masing-masing diakhiri dengan semi-colon (;). Bagian pertama adalah membentuk tabel yang akan menampung semua kata. Bagian kedua adalah menarik semua kata dari file di HDFS ke dalam tabel. Sedangkan bagian ketiga adalah SQL query yang bisa dilakukan terhadap data yang sudah dimasukkan ke tabel. Kita bisa melakukan berbagai *query* seperti SQL pada tabel yang sudah kita bentuk sehingga tidak hanya terbatas pada satu *query* saja. Output dari *query* tersebut bisa langsung ke layar, bisa ke file atau ke sistem eksternal menggunakan tool tambahan seperti Apache Thrift atau Apache Avro. Hive cukup modular sehingga bisa digabungkan dengan banyak tool lain seperti Spring, Apache Thrift, Apache Avro dan Pig. Skema apache Hive dalam hubungannya dengan Apache Hadoop dapat dilihat pada gambar 2.16.

(Sumber: <http://blog.cloudera.com/wp-content/uploads/2013/07/hiveserver1.png>)

Gambar 2.16. Skema Apache Hive dalam sistem

e. *Data Abstraction*

Pada lapisan *Data Abstraction* digunakan Apache Kafka [28]. Kafka adalah sebuah message bus yang mempublikasikan dan dirancang untuk pengiriman data stream. Model arsitektur Kafka seperti pendistribusian commit log, dan Kafka juga mendukung pemberian *resource isolation* antara menghasilkan data dan kegiatan penggunaan data itu sendiri. Kafka sering digunakan sebagai pusat penyimpanan dari stream, di mana setiap event disimpan dalam Kafka untuk jangka waktu menengah sampai nantinya diarahkan ke klaster data untuk diproses lebih lanjut dan analisis. Skema apache kafka dapat dilihat pada gambar 2.17.

(Sumber: <https://kafka.apache.org/0110/images/kafka-apis.png>)

Gambar 2.17. Skema Apache Kafka

f. *Application*

Lapisan ini merupakan lapisan yang menangani pemrosesan data masukan ke server untuk diolah guna menghasilkan informasi baru yang berguna. Pada penelitian ini dilakukan analisa berbasis *artificial intelligence*. Analisa tersebut menggunakan algoritma klasifikasi yang dalam beberapa penelitian terkait dalam pengklasifikasian mutu air memiliki performa yang baik. Dalam hal ini ada dua

algoritma klasifikasi yang akan digunakan dan dibandingkan yaitu Decision Tree dan Support Vector Machine. Dimana kedua algoritma ini telah diimplementasikan pada Apache Spark sebagai tool *machine learning* yang akan digunakan dalam penelitian ini. Sementara untuk reporting digunakan platform Node JS guna memaksimalkan aliran data *realtime* menuju ke *front end*. Sementara untuk mengatur aliran data dari dan menuju aplikasi *machine learning* digunakan Apache Spark.

Pembelajaran mesin (*Machine Learning*) [29] merupakan salah satu cabang dari kecerdasan buatan yang membahas mengenai pembangunan sistem yang didapat berdasarkan pada pembelajaran data, atau sebuah studi yang mempelajari cara untuk memprogram sebuah komputer untuk belajar. Inti dari pembelajaran mesin adalah representasi dan generalisasi.

Gambar 2.18. Machine Learning Flow

Pada Gambar 2.18 menunjukkan pembelajaran mesin secara umum, dimana input merupakan representasi data dan pemilihan ekstraksi fitur-fitur tertentu, dimana sekumpulan fitur-fitur yang memberikan deskripsi sebuah objek dinyatakan dengan $X = [X1, X2, ..., Xn]$. Dalam mengenali sebuah objek sehingga mendapatkan informasi data dari pembagian area fitur-fitur tersebut, maka diperlukan model $w = [X, y]$ tertentu yang digunakan dan berfungsi sebagai model pembelajaran mesin yang menggunakan algoritma-algoritma tertentu sehingga diperoleh Output yang berupa informasi dari data yang telah diproses, sehingga menghasilkan klasifikasi data, regresi, ataupun pengklasteran data.

1) Decision Tree

Decision Tree (pohon keputusan) adalah sebuah pohon yang digunakan sebagai prosedur penalaran untuk mendapatkan jawaban dari permasalahan yang dimasukkan. Fleksibilitas dari metode decision tree membuat metode ini atraktif, terutama karena memberikan kelebihan berupa visualisasi saran untuk membuat prosedur prediksi yang mudah diamati [30]. Decision tree sering digunakan untuk menyelesaikan kasus pengambilan keputusan seperti pada bidang medis (diagnosis penyakit pasien), komputer sains (struktur data), psikologi (teori pengambilan keputusan), dan sebagainya. Hasil visualisasi dari decision tree dapat dilihat pada gambar 2.19.

Gambar 2.19. Visualisasi *tree* dari algoritma decision tree

Karakteristik dari algoritma decision tree diilustrasikan pada gambar 2.19 yang terdiri dari beberapa elemen berikut ini:

- Root node*, atau node akar, node ini tidak mempunyai cabang input tetapi mempunyai banyak cabang output.
- Internal node*, atau node internal, setiap node internal tidak dapat menjadi node daun, dimana node internal mempunyai hanya satu cabang input dan mempunyai dua atau lebih cabang output. Node ini mengekspresikan pengujian yang didasarkan pada nilai fitur.
- Branch*, atau cabang, setiap cabang mengekspresikan nilai hasil pengujian di dalam node.

- d) *Leaf node*, atau node daun, node ini mempunyai satu cabang input dan tidak mempunyai cabang output. Node ini mengekspresikan label kelas.

Panjang atau pendeknya sebuah aturan pohon keputusan yang dihasilkan bergantung tipe algoritma decision tree yang digunakan. Terdapat beberapa tipe algoritma decision tree yang populer dan sering digunakan oleh beberapa peneliti yakni ID3, C4.5, SPRINT, SLIQ, C5.0 and CART [31]. Pada penelitian ini digunakan CART, yang dikenal dengan singkatan dari classification and regression trees [32]. CART dapat menghandel data numerik maupun kategorikal. CART bekerja dengan menghitung level impurity dari data yang memenuhi dan membangun pohon keputusan dimana tiap node menghasilkan dua kelas untuk atribut yang memenuhi. Gini indeks dihitung untuk tiap atribut, atribut dengan nilai gini terkecil dipilih sebagai atribut *breaker* [33]. Proses tersebut terus berulang hingga pohon keputusan terbentuk. Gini indeks dihitung berdasar pada probabilitas kelas ke-i untuk kelas target c.

$$Gini\ Index = 1 - \sum_{i=1}^c P_i^2 \quad (2.9)$$

2) Support Vector Machine

Support Vector Machine (SVM) [34] adalah salah satu metode dalam *machine learning* yang sangat populer saat ini. SVM dikembangkan oleh Boser, Guyon, Vapnik, dan pertama kali dipresentasikan pada tahun 1992 di Annual Workshop on Computational Learning Theory. Konsep dasar SVM sebenarnya merupakan kombinasi harmonis dari teori komputasi yang telah ada puluhan tahun sebelumnya, seperti *margin hyperplane* oleh Vapnik tahun 1964, kernel diperkenalkan oleh Aronszajn tahun 1950, dan demikian juga dengan konsep-konsep pendukung yang lain. Akan tetapi hingga tahun 1992, belum pernah ada upaya merangkaikan komponen-komponen tersebut. Prinsip dasar SVM adalah *linear classifier*, dan selanjutnya dikembangkan agar dapat bekerja pada problem *non-linear*, dengan memasukkan konsep kernel trik pada ruang kerja yang berdimensi tinggi.

SVM juga dikenal sebagai teknik pembelajaran mesin paling mutakhir setelah pembelajaran mesin sebelumnya yang dikenal Neural Network (NN). Kedua Metode tersebut menunjukkan keberhasilan dalam penggunaan pengklasifikasian dan pengenalan pola. Proses pembelajaran dilakukan dengan menggunakan pasangan data input dan data output sebagai sasaran yang diinginkan. Proses pembelajaran ini dikenal dengan *Supervised Learning*. Dari proses inilah akan diperoleh fungsi yang menggambarkan ketergantungan input dan outputnya. Selanjutnya, diharapkan fungsi yang dihasilkan dapat menggeneralisasi data dengan baik. Sehingga fungsi yang digunakan nantinya untuk data input diluar data pembelajaran.

Konsep SVM digambarkan secara sederhana sebagai upaya mencari *hyperplane* dengan menggunakan margin yang maksimal agar memberikan generalisasi yang lebih baik pada metode klasifikasi. *Hyperplane* berfungsi sebagai pemisah dua set data dari dua kelas yang berbeda. *Hyperplane* terbaik antara kedua kelas dapat ditemukan dengan mengukur margin *hyperplane* tersebut dan mencari titik maksimalnya. Margin adalah jarak antara *hyperplane* tersebut dengan data terdekat dari masing-masing kelas. Data yang paling dekat ini disebut sebagai *Support Vector*. Hal ini dapat digambarkan pada Gambar 2.20 dimana terlihat lingkaran ganda. Usaha untuk mencari lokasi *hyperplane* merupakan inti dari proses pelatihan pada SVM. Contoh implementasi SVM dapat dilihat pada Gambar 2.20, SVM dengan *hyperplane* dan margin.

Gambar 2.20. SVM dengan *hyperplane* dan margin

3) Apache Spark

Apache Spark [35] adalah sebuah mesin pengolahan data hadoop memiliki fungsi analisis data yang kompleks dan fungsi machine learning serta algoritma grafik. Apache Spark bisa menjalankan program 100 kali lebih cepat dalam memori dan 10 kali lebih cepat pada disk dibandingkan MapReduce. Apache Spark dapat berjalan beriringan dengan sistem penyimpanan Hadoop dan HDFS. Aplikasi pada Apache Spark dapat dilihat pada Gambar 2.21.

(Sumber: <https://spark.apache.org/images/spark-stack.png>)

Gambar 2.21. Apache Spark

Ada empat komponen utama pada Apache Spark yang dapat digunakan untuk mendukung proses analisa dalam *Big Data*, yaitu:

a) Spark-SQL

Spark SQL adalah komponen di atas Spark yang memperkenalkan abstraksi data baru yang disebut *DataFrame*, yang menyediakan dukungan untuk data terstruktur dan semi-terstruktur. Spark SQL menyediakan bahasa domain-spesifik untuk memanipulasi *DataFrame* menggunakan Scala, Java, atau Python. Spark SQL menyediakan dukungan pada bahasa SQL sebagai antarmuka baris perintah dan juga menyediakan dukungan terhadap server ODBC / JDBC.

b) Spark Streaming

Spark Streaming memanfaatkan penjadwalan cepat untuk melakukan analisis streaming. Dibutuhkan untuk menganalisis data secara *realtime*.

Adapun Sumber data bagi Spark dengan menggunakan skema streaming dapat berasal dari Kafka, Flume, Twitter, ZeroMQ, Kinesis, dan TCP/IP Socket.

c) MLlib (Machine Learning)

Spark MLlib adalah kerangka mesin pembelajaran dan statistik terdistribusi di atas Spark. Dukungan Spark MLlib sebagai algoritma *machine learning* dan statistik meliputi banyak hal yaitu:

- (1) Summary Statistics, Correlation, Stratified Sampling, Hypothesis Testing, Random Data Generation.
- (2) Klasifikasi dan Regresi: Support Vector Machine, Logistic Regression, Decision Trees, naive Bayes Classification.
- (3) Teknik Collaborative Filtering termasuk Alternating Least Squares (ALS).
- (4) Metode Cluster Analysis termasuk K-Means dan Latent Dirichlet Allocation (LDA)
- (5) Teknik Dimensionality Reduction seperti Singular Value Decomposition (SVD), dan Principal Component Analysis (PCA)
- (6) Ekstraksi Fitur dan Fungsi Transformasi
- (7) Algoritma Optimasi seperti Stochastic Gradient Descent, Limited-memory BFGS (-L-BFGS)

d) GraphX (graph)

GraphX adalah kerangka pengolahan grafik terdistribusi di atas Apache Spark. Karena didasarkan pada RDDs, yang berubah, grafik yang berubah dan dengan demikian GraphX tidak cocok untuk grafik yang perlu diperbarui, apalagi dengan cara transaksional seperti database grafik. GraphX menyediakan dua API terpisah untuk pelaksanaan secara besar-besaran algoritma paralel (seperti PageRank): abstraksi Pregel, dan gaya MapReduce API yang lebih umum. Tidak

seperti pendahulunya Bagel, yang secara resmi tidak digunakan dalam Spark 1.6, GraphX memiliki dukungan penuh untuk grafik. GraphX dapat dilihat sebagai basis memory dari Apache Giraph, yang dimanfaatkan Hadoop MapReduce berbasis disk. Seperti Apache Spark, GraphX awalnya dimulai sebagai sebuah proyek penelitian di UC Berkeley AMPLab dan Databricks, dan kemudian disumbangkan ke Apache Software Foundation dan proyek Spark.

Dalam sistem yang dibangun ini, untuk analisa realtime menggunakan Spark dengan memadukan Spark Streaming, Spark MLlib dan Spark-SQL, sebagaimana pada gambar 2.22.

Gambar 2.22. Skema Apache Spark untuk analisa *realtime*

Spark dalam proses analisa data telah beberapa kali melakukan perubahan tipe data yang digunakan. Tipe data yang digunakan dalam Spark berbeda dengan tipe data yang biasa digunakan pada tool yang lain. Selain itu untuk melakukan pemrograman python, Spark memiliki dukungan dengan meluncurkan library pyspark. Library spark untuk para programmer python. Berikut ulasan kedua hal tersebut:

a) Tipe data abstraksi pada Spark

Pada Spark versi 2.0 terdapat 3 tipe data abstraksi yang disediakan Spark dan dapat digunakan, yaitu *RDD (Resilient Distribution Dataset)*, *DataFrame* dan


```
In [4]: from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)

df = sqlContext.read.format('com.databricks.spark.csv').options(header='true').load('data/datasensordb-sparksvm.csv')
df

Out[4]: DataFrame[output: string, suhu: string, ph: string, do: string, sal: string]

In [5]: df.registerTempTable("sensordb")
df.printSchema()
df.show()
```

```
root
|-- output: string (nullable = true)
|-- suhu: string (nullable = true)
|-- ph: string (nullable = true)
|-- do: string (nullable = true)
|-- sal: string (nullable = true)
```

output	suhu	ph	do	sal
1	29.69	8.15	-0.02	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.02	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.17	0.0	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.16	-0.03	0.28
1	29.01	8.15	0.22	0.28

Gambar 2.25. Contoh penggunaan DataFrame

Dataset merupakan dataframe yang sudah di pilah menjadi suatu baris baru. Spark tidak menyediakan fungsi khusus untuk python dalam mengakses suatu dataset, namun struktur data python sudah dapat langsung mengakses dataframe sehingga dapat dibentuk menjadi dataset. Contoh penggunaan dataset dapat dilihat pada Gambar 2.26.

```
In [8]: #Python dataframe as dataset
df[((df['ph']>8.15)&(df['output']=='1'))].show()
```

output	suhu	ph	do	sal
1	29.01	8.17	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.02	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.17	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.16	-0.01	0.28
1	29.01	8.17	0.0	0.28

Gambar 2.26. Contoh penggunaan Dataset

Spark menggunakan RDD mulai versi awal, pada tahun 2011 dikembangkan tipe data DataFrame yang mulai digunakan pada Spark versi 1.3, dan mulai Spark 1.6 dan terbaru dikenalkan tipe data Dataset.

b) pySpark

pySpark [35] merupakan salah satu modul python yang digunakan untuk apache Spark. PySpark adalah Python API untuk aplikasi Spark. pySpark menggunakan bahasa pemrograman python dalam mengimplementasikan machine learning yang digunakan untuk pengklasifikasian.

Dalam penelitian ini pySpark diintegrasikan dengan Zeppelin untuk dapat menjalankan *script* python secara interaktif melalui web user interface.

g. *Collaboration & Processes*

Lapisan ini menekankan pada interaksi antara analis dengan sistem. Dikarenakan untuk dapat melakukan analisa atau membangun model klasifikasi dibutuhkan media yang dapat menghubungkan analis dan sistem. Pada platform ini digunakan Apache Zeppelin untuk memfasilitasi interaksi antara analis dan sistem. Zeppelin [36] adalah aplikasi antar muka berbasis web yang dapat digunakan untuk membuat program analisis yang dapat diintegrasikan dengan Spark. Zeppelin juga dapat membuat program analisis yang menggunakan metode *machine learning* sehingga dapat memproduksi grafik analisa.

Zeppelin ini dapat diintegrasikan juga dengan librari pemrograman python Spark (pySpark). User interface dari zeppelin dapat dilihat pada gambar 2.27.

(Sumber: <https://zeppelin.apache.org/assets/themes/zeppelin/img/notebook.png>)

Gambar 2.27. User interface Zeppelin Notebook

2.2 PENELITIAN TERKAIT

Berikut ini mengenai penelitian terkait yang mencoba untuk menyelesaikan permasalahan dalam penelitian ini antara lain:

SEMAR terdiri dari: 1) *ROV Water Quality Monitoring System* [37], ide tentang *ROV Water Quality Monitoring System* berdasarkan pada pengalaman Badan Lingkungan pemerintah yang biasanya mengambil sampel untuk memonitor kondisi air sungai. Pada penelitian ini kami mengajukan cara yang berbeda dengan mengkombinasikan *Remotely Operated Vehicle* (ROV) atau robot air kecil yang dapat dikontrol dengan sensor kualitas air. Dengan alat ini, petugas tidak perlu mengambil sampel air secara manual. Dan hasil dari sensor dapat langsung dikirim ke server melalui internet; 2) *Wireless Mesh Network* [38], dalam SEMAR kami menggunakan teknologi *wireless mesh* untuk memperlebar jangkauan komunikasi antara sensor dan server; 3) *Portable Water Quality Monitoring System* [39], kami mengembangkan sebuah alat portable dan murah untuk pemantauan kualitas air yang hasilnya dapat langsung dikirim ke server; 4) *Coral Reef Monitoring System* [40], sistem ini dibangun untuk melakukan monitoring terhadap kondisi karang pada perairan dangkal dengan menggunakan kamera aktif dan datanya langsung dikirim ke server; 5) *Big Data Storage Architecture* [41], data dari sensor dikumpulkan dan disimpan pada Hadoop server dengan dukungan HDFS, Yarn, dan MapReduce.

Modaresi et. al. [42], telah melakukan studi mengenai klasifikasi kualitas air menggunakan metode Indeks Kualitas Air CCME (Canadian Council of Minister of the Environment) dengan 2 parameter yaitu Nitrate and Chloride. Dalam penelitian ini menggunakan 3 algoritma yaitu Support Vector Machines, Probabilistic Neural Networks, and K-Nearest Neighbor. Hasil penelitian tersebut menunjukkan algoritma SVM menampilkan best performance dengan tidak ada error pada proses kalibrasi dan validasi. Ladjal et al. [43], juga telah melakukan studi mengenai klasifikasi kualitas air menggunakan Dempster-Shafer Theory. Dalam penelitian ini menggunakan 4 parameter yaitu *Temperature*, pH, *Conductivity*, dan *Turbidity*. Algoritma yang digunakan adalah ANN dan SVM. Hasil dari penelitian ini menunjukkan algoritma SVM memiliki performa yang

lebih baik dibandingkan menggunakan ANN. Jaloree et al. [44], juga telah melakukan studi mengenai klasifikasi kualitas air menggunakan algoritma Decision Tree. Parameter yang digunakan dalam penentuan kualitas air adalah pH, DO, BOD, No3_N, dan NH3_N. Dimana hasil dari proses *training* menunjukkan tingkat akurasi 95.4545%. Saghebian et al. [45] telah melakukan study tentang klasifikasi kualitas air menggunakan Decision Tree. Hasil study menunjukkan Decision Tree mampu memproses dataset yang digunakan. Hasil studi juga menunjukkan rata-rata CCI (*Correctly Classified Instances*) dan Statistik Kappa untuk prediksi kualitas air berada pada angka 0.88 dan 0.83%. Keempat penelitian di atas dalam pengaplikasiannya belum mendukung proses klasifikasi *realtime* dan belum terintegrasi dengan teknologi *Big Data*.

Fazio et al. [46], telah melakukan studi mengenai implementasi *big data* sebagai media penyimpanan untuk *Smart Environment Monitoring*. Dalam penelitian tersebut digambarkan secara umum mengenai *System in Cloud environment for Advanced Multi-risk Management* (SIGMA) yang merupakan bagian dari *Italian National Operative Program* (PON). Project tersebut diharapkan dapat mengakomodir segala macam data yang bersumber dari berbagai environment. Dalam system tersebut belum terlihat gambaran adanya proses *Big Data Analytic*.

Meng et al. [47], telah melakukan studi mengenai performa antara Mahout dan MLLib Spark v1.1 dan v1.4 menggunakan Amazon Reviews dataset yang dijalankan pada 16 node. Hasil penelitian tersebut menunjukkan penjadwalan MapReduce pada Mahout menjadi *overhead* and *lack* pada dataset ukuran menengah. Hal yang berbeda pada Spark MLLib yang mengalami kinerja yang sangat baik dan bahkan dapat menangani dataset dengan ukuran lebih besar. Sementara itu Richter et al. [48], telah melakukan studi komparasi beberapa toolkit untuk *machine learning* pada *Big Data* yaitu Mahout MapReduce, Mahout Samsara, MLLib Spark, H2O, dan SAMOA. Dari penelitian tersebut secara rata-rata MLLib Spark dan H2O memiliki performa lebih baik dibanding toolkit yang lain dalam hal *Extensibility*, *Scalability*, *Usability*, *Fault Tolerance* and *Speed*. Namun dari sisi implementasi algoritma, MLLib Spark merupakan tool dengan implementasi terbanyak dengan total sebanyak 17 algoritma dibandingkan dengan

tool yang lain seperti Mahout MapReduce sebanyak 13 algoritma, H2O 10 algoritma, Mahout Samsara 7 algoritma and SAMOA 3 algoritma. Kelebihan lain dari MLlib Spark dibandingkan yang lain adalah kemampuannya yang dapat menangani proses *batch* maupun *stream*. Berbeda dengan Mahout dan H2O yang hanya menangani proses *batch*, dan SAMOA yang hanya menangani proses *stream*.

BAB 3

DESAIN SISTEM

3.1 DESAIN SISTEM

Dalam penelitian ini digunakan teknologi *Internet of Things* dan *Big Data* dalam mengembangkan dan mendesain sistem. Secara umum desain sistem didasarkan pada tujuh lapisan pada teknologi *Internet of Things* yang dapat dilihat pada gambar 3.1. Pengadopsian teknologi *Big Data* akan diintegrasikan ke dalam beberapa lapisan dari IoT.

Gambar 3.1. Gambaran Umum Sistem

Integrasi dari keseluruhan layer akan menghasilkan sebuah platform yang digunakan untuk tujuan tertentu seperti platform untuk monitoring lingkungan, kesehatan, lalu lintas, dll. Dalam penelitian ini akan dihasilkan sebuah platform yang dapat digunakan untuk monitoring lingkungan, utamanya lingkungan air. Gambar 3.2 menunjukkan detail desain sistem yang dibangun dalam penelitian ini yang didasarkan pada tujuh lapisan *Internet of Things*.

Gambar 3.2. Desain sistem

Desain sistem terdiri dari 7 tahapan sesuai pada lapisan dari *Internet of Things* yang menjadi dasar sistem dalam penelitian ini dibangun meliputi:

3.1.1 *Physical Devices & Controllers*

Physical Devices & Controllers merupakan node-node yang telah dilengkapi dengan sensor kualitas air yang tersebar di beberapa titik sepanjang sungai yang mengalir kota Surabaya.

Sensor kualitas air yang digunakan pada penelitian ini adalah ‘Atlas Scientific’ *kit* sensor yang terdiri dari sensor *Potential of Hydrogen* (pH), *Oxidation Reduction Potential* (ORP), *Dissolved Oxygen* (DO), *Electrical Conductivity* (EC) dan *Temperature*.

Gambar 3.3. Desain sensor kualitas air

Untuk kontroler digunakan sistem benam jenis Raspberry Pi tipe 3. Sistem benam ini menggunakan sistem operasi Raspbian, dimana sistem operasi ini berbasis Debian GNU/Linux yang telah dimodifikasi untuk dapat berjalan pada sistem benam Raspberry Pi. Dalam menghubungkan antara sensor dan kontroler digunakan kabel USB-to-serial. Pada penelitian ini terdapat 7 titik lokasi yang tersebar di sepanjang sungai kota Surabaya dan akan mengambil data dari sensor air secara periodik yaitu tiap 5 detik. Titik-titik lokasi tersebut dapat dilihat pada gambar 3.4.

Gambar 3.4. Lokasi node sensor

Adapun titik latitude dan longitude dari tiap lokasi dapat dilihat pada tabel 3.1 yang tersebar di sepanjang sungai di Kota Surabaya.

Tabel 3.1. Titik Latitude dan Longitude lokasi node sensor

No.	Sensor ID	Latitude	Longitude
1	Joyoboyo	-7.299487	112.734737
2	Ngagel	-7.297746	112.741606
3	Pintu Air Jagir	-7.301354	112.743381
4	Prapatan Prapen	-7.305796	112.761288
5	Stikom	-7.310150	112.783255
6	Wonorejo	-7.305158	112.806950
7	Karang Pilang	-7.349155	112.680496

Pada sistem benam dibangun sebuah aplikasi yang digunakan untuk melakukan penerimaan data dari sensor, pengiriman data ke server, dan penyimpanan data ke penyimpanan lokal sistem benam sebagai *backup*. Algoritma aplikasi node sensor dapat dilihat pada gambar 3.5.

Algorithm 1 Physic Sensor App Procedure

```
1: Begin
2: sensorInfo ← sensorID, latitude, longitude,
3: loop:
4:   time ← system time
5:   sensorData ← pH,DO,EC,ORP,Temp,TDS,Sal,Sg
6:   Send sensorInfo, time, sensorData to MQTT Server
7:   Save local
8:   Sleep (5)
9:   Goto loop.
10: End
```

Gambar 3.5. Algoritma aplikasi pada node sensor

Dari gambar 3.5 terdapat beberapa proses yaitu proses pengambilan info sensor berupa sensor ID, latitude dan longitude dari lokasi node sensor. Setelah itu dilakukan proses perulangan untuk melakukan pengambilan waktu sistem, pengambilan data sensor dengan menggunakan pySerial. Keseluruhan data kemudian dikirim menggunakan protokol MQTT dan selanjutnya disimpan secara lokal sebagai data *backup*. Data yang dikirim berbasis teks dengan data antar sensor dipisahkan dengan koma dengan urutan: ID Sensor, *Latitude*, *Longitude*, *Date*, *Time*, ORP, pH, EC, TDS, Sal, SG, DO dan *Temperature*. Dalam protokol MQTT node sensor bertindak sebagai MQTT *Publisher*, merupakan pihak yang mengirim data ke MQTT *Broker* (server). Komunikasi MQTT menggunakan port 1883. Pengiriman data menggunakan topik ‘watermonitoring’. Aplikasi pada node sensor ini dibangun menggunakan bahasa pemrograman Python.

3.1.2 Connectivity

Dalam penelitian ini sebagai media komunikasi antara node sensor yang dipasang di sepanjang sungai di Kota Surabaya dan server terpusat penerima data digunakan modem 4G. Modem 4G yang digunakan menggunakan protokol IEEE 802.11n. Sementara untuk konektivitas antara *embedded system* dengan modem menggunakan jaringan Wifi. Sistem pengalamatan IP pada *embedded system* yang digunakan, menggunakan DHCP (Dynamic Host Configuration Protocol) yang dikontrol melalui modem.

Modem USB 4G bekerja sebagai router yang menghubungkan koneksi WAN dengan menggunakan LTE dengan jaringan LAN yang terhubung melalui *wireless* LAN. Dalam pengujian didapatkan throughput sekitar 20 Mbps.

3.1.3 *Edge Computing*

Bagian ini membahas proses penerimaan data dari node sensor oleh server. Pada sisi server data diterima oleh MQTT *Broker*. Sesuai dengan skema dari komunikasi MQTT yang menggunakan node perantara (*Broker*). Pada penelitian ini digunakan ‘Mosquitto’ sebagai MQTT *Broker*. Data yang diterima oleh server memiliki basis yang sama dengan data yang dikirim oleh node sensor dan disimpan pada topik ‘watermonitoing’. Nantinya aplikasi yang membutuhkan data tersebut akan melakukan *consume/subscribe* ke *Broker* melalui topik dan port yang sama.

3.1.4 *Data Accumulation*

Data Accumulation merupakan proses penyimpanan data yang diterima dari node-node ke Hadoop HDFS. Sebelum melakukan proses penyimpanan, data pada MQTT *Broker* (Mosquitto) didistribusikan ke Kafka *Broker*. Jadi pada sisi server, proses selanjutnya MQTT *Broker* akan digantikan oleh Kafka *Broker*. Proses pendistribusian data dari Mosquitto ke Kafka *Broker* membutuhkan sebuah kode program, karena secara default keduanya belum saling mendukung satu sama lain. Kode yang digunakan yaitu MQTTKafkaBridge [49], yang ditulis menggunakan Bahasa pemrograman Java. Cara kerja dari MQTTKafkaBridge adalah mendistribusikan data secara langsung data yang diterima oleh MQTT *Broker* ke Kafka *Broker* dan disimpan pada topik yang sama. Hal ini menyebabkan data pada Kafka *Broker* juga memiliki topik yang sama dengan data pada Mosquitto yaitu ‘watermonitoring’.

Apache kafka digunakan untuk memudahkan pendistribusian dan manajemen data pada sistem yang sedang berjalan. Apache kafka juga mendukung *stream processing* untuk aplikasi yang membutuhkan ketersediaan data *stream*. Apache kafka juga memiliki kapasitas *cache* yang lebih besar dibanding Mosquitto, sehingga data yang berasal dari sensor tidak akan hilang jika terjadi masalah pada aplikasi di server. Dengan menggunakan Kafka juga akan memudahkan penyediaan data antar lapisan pada IoT saat dikombinasikan dengan Big Data.

Data yang ada pada Kafka Broker selanjutnya secara *stream* dibaca oleh Spark dan menghasilkan tipe data RDD yang kemudian di-*buffer* selama 10 detik. Setiap 10 detik, data yang di-*buffer* tersebut akan di-load ke Hadoop HDFS menggunakan *query* Hive. *Query* Hive dilakukan menggunakan Spark SQL. Data tersebut disimpan ke dalam tabel ‘watermonitoringku’.

Aplikasi yang dibangun pada proses ini terpisah dengan proses klasifikasi *realtime*. Hal ini untuk memangkas waktu proses klasifikasi agar lebih cepat, dikarenakan pada proses penyimpanan data ke Hadoop HDFS terdapat *delay* waktu untuk proses MapReduce.

Penggunaan Hive untuk *query* penyimpanan data dengan pertimbangan kemudahan dalam penggunaan karena menggunakan *query* yang mirip dengan *query* SQL. Selain itu dalam sistem ini tidak terdapat kebutuhan akan akses *realtime* terhadap data yang disimpan. Data-data hasil pengukuran air yang tersimpan pada Hadoop HDFS, selanjutnya jika dibutuhkan dapat di-load secara *batch* dengan menggunakan *query* Hive.

3.1.5 Data Abstraction

Data Abstraction merupakan proses pengaksesan data yang telah diterima oleh server. Pada tahap ini, digunakan Kafka *Broker* untuk melakukan pengambilan data dari MQTT *Broker* (Mosquitto) dan untuk diteruskan kepada lapisan di atasnya. Pendistribusian data yang dilakukan secara langsung dari Mosquitto bertujuan untuk memangkas waktu pemrosesan data. Selanjutnya data pada Kafka Broker akan di-*subscribe*/di-*consume* oleh aplikasi yang membutuhkan, baik itu untuk proses analisa maupun untuk proses visualisasi. Dalam pengimplementasiannya, Kafka identik dengan MQTT hanya berbeda pada penggunaan istilah. Pada Kafka aplikasi yang mengirim data disebut Kafka *Producer*, aplikasi yang menerima disebut Kafka *Consumer*, sementara perantara keduanya disebut Kafka *Broker*. Data-data yang diterima juga disimpan dalam topik tertentu. Dalam penelitian ini data yang berasal dari node sensor diberi nama ‘watermonitoring’, sementara data hasil analisa nantinya akan diberi nama ‘wateranalytic’. Data pada lapisan ini yang akan di-*consume* oleh aplikasi analisa *realtime*.

3.1.6 Application

Proses pembangunan bagian *application* terdiri dari 3 sub yaitu; a) Proses *learning*, yang merupakan proses pembangunan model klasifikasi; b) Klasifikasi *realtime*, yang merupakan proses pembangunan aplikasi klasifikasi *realtime*; dan c) Visualisasi, yang merupakan proses visualisasi pada *front-end web interface*.

a. Proses Learning

Proses *learning* ini merupakan proses pembangunan model klasifikasi yang digunakan dalam sistem. Proses ini dilakukan sebelum klasifikasi *realtime* dilakukan. Performa dari proses learning sangat mempengaruhi hasil klasifikasi *realtime*.

Pada penelitian ini digunakan data yang diperoleh dari PDAM Surya Sembada Surabaya. Yaitu data uji laboratorium dan data live sensor. Data uji laboratorium yang merupakan uji laboratorium harian dari tahun 2014 hingga tahun 2016 dengan 1347 sampel dan 20 atribut yang terdiri dari Tanggal, Suhu, Kekeruhan, Warna, TSS, pH, Alkalinitas, CO2 Bebas, DO, Nitrit, Amonia, Tembaga, Fosfat, Sulfida, Besi, Kromium Hexavalen, Mangan, Seng, Timbal dan COD. Data live sensor merupakan data yang berasal dari sensor live PDAM yang dipasang pada pintu air Ngagel. Data tersebut diambil dari bulan Maret hingga bulan Agustus 2016 dengan 205.720 sampel dan 6 atribut yaitu Tanggal, Jam, Kekeruhan, TSS, pH, DO, dan Suhu. Gambar 3.6 menunjukkan data air dari uji laboratorium PDAM Surya Sembada Surabaya.

No	SAMPLE STATION ID	WAKTU	POINT ID	YEAR	HARAH																							
					Subu	Kekuhuan	Pm-Ca	Pm-Co	Pm-Cu	Pm-Fe	Pm-Mn	Pm-Ni	Pm-Pb	Pm-Sb	Pm-Ti	Pm-Zn	COD	Dissolved	HARAH									
																			NTU	Vm-Ca	Vm-Co	Vm-Cu	Vm-Fe	Vm-Mn	Vm-Ni	Vm-Pb	Vm-Sb	Vm-Ti
		PP-02/2001			Subu-Uda+30C			50	6.4					16 (ppm)	0.06	0.5	0.02	0.2	0.002	0.3	0.05	0.1	0.06	0.04	10	200		
1	Nagel	Januari	2014	2	25.4	66.4	242.75	76	7.9	106.1	4.7	4.64	0.09	0.555256	0.163974	0.146282	0.167032	0.460574	0.10666677						16	70.54020		
2	Nagel	Januari	2014	3	25	54.4	242.75	76	7.9	106.1	4.7	4.64	0.09	0.555256	0.163974	0.146282	0.167032	0.460574	0.10666677						16	70.54020		
3	Nagel	Januari	2014	4	25.3	78.0	481.525	95.4	8.9	133.3	2.30	4.38	0.08	1.2	0.38	0.08	1.2	0.38	0.08	1.2	0.38	0.08	1.2	0.38	0.08	1.2	0.38	0.08
4	Nagel	Januari	2014	7	25.1	244	211.723	45	7.75	160.89	2.74	3.93	0.07	1.147004	0.262145	0.167101	1.529672	1.523233	0.1570425									
5	Nagel	Januari	2014	8	25.3	474	553.362	625	7.8	144.32	2.62	4.36	0.16	0.084098	0.235775	0.220076	0.899992	0.2344620	0.1232771						16	50.37383		
6	Nagel	Januari	2014	9	25.1	163	190.391	348	7.85	165.65	3.34	3.22	0.08	0.562431	0.12	0.1037785	0.5904	0.3855856	0.1026573									
7	Nagel	Januari	2014	10	25.3	144.5	224.947	75	7.9	106.1	4.7	4.64	0.09	0.555256	0.163974	0.146282	0.167032	0.460574	0.10666677									
8	Nagel	Januari	2014	12	25.3	228.90	355.221	389	8.9	203.92	4.080	2.92	0.209	0.185	0.227	0.17	0.27	0.313	0.054									

Gambar 3.6. Data air uji laboratorium PDAM Surya Sembada Surabaya.

Gambar 3.7 menunjukkan data air dari live sensor PDAM pada pintu air Ngagel pada PDAM Surya Sembada Surabaya.

No.	Timestamp	Sensor ID	Measurement	Value	Unit/Label
1	2016-03-10 12:23:37	S01	Measuring,VisoTurb700IQ,09250704,AIR BAKU	91	NTU,Turb,, ,0x0
2	2016-03-10 12:23:37	S03	Measuring,MIQIC2 REC1,09381117,AIR PRO IP2,OFL,ppm,Chlor	0.01	mA,0x0
3	2016-03-10 12:23:37	S04	Measuring,ViSolid700IQ,12490907,DIS.TOR IP2,125.5,mg/l,TSS,860,mg/l#,0x20		
4	2016-03-10 12:23:37	S05	Measuring,SensoLyt700IQ,12390797,AIR BAKU NG	7.19	,pH,30.2, °C,0x0
5	2016-03-10 12:23:37	S06	Measuring,SC FDO 700,12100970,AIR BAKU NG	2.29	mg/l,O2,30.1, °C,0x0
6	2016-03-10 12:23:37	S07	Measuring,VisoTurb700IQ,12311119,AIR_PRO_IP2,1.05,NTU,Turb,, ,0x0		

Gambar 3.7. Data air sensor *live* PDAM Surya Sembada Surabaya.

Dalam menentukan label dari data air digunakan metode Indeks Pencemaran. Indeks Pencemaran merupakan salah satu dari dua metode yang diakui di Indonesia dalam menentukan kualitas air. Rumus yang digunakan dalam menentukan Pollution Index:

$$PI_j = \sqrt{\frac{(C_i/L_{ij})_M^2 + (C_i/L_{ij})_R^2}{2}} \quad (3.1) \quad (1)$$

L_{ij} merupakan konsentrasi parameter kualitas air sesuai dengan nilai baku mutu yang digunakan pada air (j), and C_i merupakan konsentrasi parameter kualitas air (i) yang diambil pada lokasi atau pada aliran sungai tertentu, kemudian PI_j adalah Pollution Index untuk air (j) yang merupakan fungsi dari $\frac{C_i}{L_{ij}}$ dan ditentukan dari resultan nilai maksimum (M) dan nilai rerata (R) dari tiap parameter terhadap nilai baku mutunya. Selanjutnya nilai PI_j dievaluasi dan dapat ditentukan kategori Indeks Pencemaran sebagaimana pada tabel 3.2:

Tabel 3.2. Kategori Indeks Pencemaran

No.	Indeks Pencemaran	Kategori
1	$0 \leq PI_j \leq 1.0$	Memenuhi baku mutu
2	$1.0 < PI_j \leq 5.0$	Tercemar Ringan
3	$5.0 < PI_j \leq 10.0$	Tercemar Sedang
4	$PI_j > 10.0$	Tercemar Berat

Dari data uji laboratorium dan data live sensor, parameter yang digunakan dalam menentukan Indeks Pencemaran adalah kekeruhan, TSS, pH, DO dan Suhu. Kelima parameter ini merupakan parameter standar yang digunakan PDAM Surya Sembada Surabaya dalam *me-monitoring* kondisi air baku sebelum diproses menjadi air produksi. Dari kategori Indeks Pencemarannya maka ditentukan empat label yaitu:

Tabel 3.3. Penentuan label Indeks Pencemaran

Indeks Pencemaran	Keterangan	Label
$0 \leq IP \leq 1.0$	memenuhi baku mutu	0
$1.0 \leq IP \leq 5.0$	tercemar ringan	1
$5.0 \leq IP \leq 10.0$	tercemar sedang	2
$IP \geq 10.0$	Tercemar berat	3

Adapun untuk standar kualitas air yang digunakan sebagai pembanding dalam menentukan Indeks Pencemaran adalah Peraturan Pemerintah Nomor 82 tahun 2001 tentang penentuan standar kualitas air, Peraturan Daerah Provinsi Jawa Timur nomor 2 tahun 2008 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air Di Provinsi Jawa Timur, Peraturan Daerah Kota Surabaya nomor 02 tahun 2004 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air, dan peraturan Menteri Kesehatan nomor 492/MENKES/PER/IV/2010 tentang Persyaratan Kualitas Air Minum sebagaimana yang dapat dilihat pada tabel 3.4.

Tabel 3.4. Standar Kualitas Air

No	Parameter	Standar
1	Turbidity	5
2	TSS	10
3	pH	6-9
4	DO	Minimum 6
5	Temperature	$25^{\circ}\text{C} \pm 3^{\circ}\text{C}$

Data yang telah dilakukan *preprocessing* data, selanjutnya dilakukan proses training dataset. Spark MLlib telah memiliki dukungan terhadap beberapa algoritma klasifikasi, klastering dan regresi. Sebagaimana dari penelitian yang telah dilakukan peneliti-peneliti sebelumnya, maka digunakan dua algoritma klasifikasi yang kemudian hasilnya dibandingkan yaitu Support Vector Machine dan Decision Tree. Dataset uji laboratorium setelah *preprocessing* dapat dilihat pada gambar 3.8, dimana data yang dihasilkan sudah tidak memiliki missing/corrupt value, inkonsistensi nilai, dll.

Suhu	Kekeruhan	SS	pH	DO	Class
25.40	66.40	86.00	7.900	4.64	1
25.00	54.40	75.00	8.000	4.33	1
25.30	786.00	954.00	7.720	4.38	2
25.10	244.00	425.00	7.750	3.83	2
25.30	474.00	625.00	7.800	4.36	2
25.10	143.00	248.00	7.850	3.22	2
25.00	64.30	75.00	7.600	2.57	1
26.30	276.00	389.00	7.900	2.92	2
25.20	88.60	125.00	7.850	3.57	2
25.10	106.00	136.00	7.800	3.79	2
26.00	78.00	94.00	8.000	3.53	2
25.30	57.60	78.00	7.800	3.28	1
25.00	37.70	62.00	7.700	3.92	1
25.40	45.50	85.00	7.900	4.09	1

Gambar 3.8. Dataset uji laboratorium setelah *preprocessing*

Gambar 3.9 menunjukkan dataset *live* sensor setelah *preprocessing*, dimana data-data yang dihasilkan sudah tidak memiliki missing/corrupt value, inkonsistensi nilai, dll.

Time	Turb	TSS	pH	DO	Temp	Class
3/10/2016 12:23	91	125.5	7.19	2.29	30.1	2
3/10/2016 12:28	91	171.6	7.2	2.28	30.1	2
3/10/2016 12:29	91	246.7	7.19	2.28	30.1	2
3/10/2016 12:30	91	264.2	7.2	2.28	30.2	2
3/10/2016 12:31	90	215.9	7.18	2.28	30.2	2
3/10/2016 12:32	91	123.3	7.19	2.28	30.2	2
3/10/2016 12:33	90	164.2	7.19	2.27	30.2	2
3/10/2016 12:34	90	165.5	7.19	2.27	30.2	2
3/10/2016 12:35	89	113.8	7.21	2.27	30.2	2
3/10/2016 12:36	89	123.2	7.19	2.27	30.2	2
3/10/2016 12:37	89	156.1	7.18	2.26	30.2	2

Gambar 3.9. Dataset sensor *live* setelah *preprocessing*

Untuk mengevaluasi model yang dihasilkan digunakan metode Hold-out. Metode ini membagi dataset ke dalam dua bagian, yaitu *training* set dan *test* set. *Training* set digunakan untuk membangun model klasifikasi dan *test* set digunakan untuk menguji performa dari model. Dataset akan dibagi dengan perbandingan 70:30, 70% digunakan sebagai *training* set dan 30% digunakan sebagai *test* set. Aplikasi pada proses learning ini dapat dilihat pada gambar 3.10.

```

1: method ← Linear SVM, Decision Tree
2: Begin
3:   Retrieve Dataset
4:   Split Dataset (70,30)
5:   Machine Learning Training (method)
6:   Machine Learning Testing (method)
7:   Calculate MSE, Mislabel, Accuracy
8:   Save Model
9: End

```

Gambar 3.10. Prosedur Learning

Pada gambar 3.10 terdapat dua metode klasifikasi yang diinisialisasi yaitu Linear SVM dan Decision Tree. Dataset yang telah di-load selanjutnya dibagi dengan perbandingan 70% untuk training dan 30% untuk testing. Selanjutnya menggunakan Spark MLlib dilakukan proses *training* dan *testing*. Hasil proses tersebut kemudian dikalkulasi untuk menghasilkan *Mean Squared Error* (MSE), *Mislabel*, dan *Accuracy*. Setelah itu model yang dihasilkan disimpan ke dalam file sistem untuk digunakan pada klasifikasi *realtime*.

b. Klasifikasi *realtime*

Klasifikasi *realtime* menggunakan teknologi *Big Data Analytic* yang salah satunya diwakili oleh Apache Spark. Pada proses ini digunakan Spark MLlib dan menggunakan model yang dihasilkan dari langkah pembangunan model klasifikasi pada bagian proses *learning*. Hasil klasifikasi akan disimpan pada variabel 'label' yang bertipe *numeric*. Selanjutnya akan ditranslasikan ke dalam kategori Indeks Pencemaran (*Fulfil Standard*, *Lightly Polluted*, *Polluted*, dan *Heavy Polluted*) dan disimpan ke dalam variabel 'criteria'.

Data air dan data hasil klasifikasi selanjutnya akan diloat ke Kafka *Broker* menggunakan kafka *Producer* pada port 9092 dengan topik ‘wateranalytic’. Topik ini juga yang akan digunakan untuk proses visualisasi. Data yang dikirim oleh aplikasi analisa *realtime* menggunakan format JSON dengan susunan: Sensor ID, Latitude, Longitude, Date, Time, ORP, pH, EC, TDS, Sal, SG, DO, Temperature, Label, Criteria. Aplikasi yang dibangun dalam bagian ini sesuai gambar 3.11.

```

1: function parsing(data)
2:   parse  $\leftarrow$  param1, param2, ..., param n
3:   return parse
4: end function
5: function realtimeClassification(parse)
6:   classify  $\leftarrow$  Model.predict
7:   return classify
8: end function
9: function sendToKafka(data,classify)
10:  Send data, classify to Kafka Broker
11: end function
12: Begin
13:  Spark initialization
14:  Load Model
15:  data  $\leftarrow$  pH,DO,EC,ORP,Temp,TDS,Sal,Sg from Kafka Broker
16:  parsing (data)
17:  realtimeClassification (parse)
18:  SendToKafka(data,classify)
19: End

```

Gambar 3.11. Algoritma klasifikasi *realtime*

Pada gambar 3.11 terdapat beberapa proses. Proses pertama yaitu inisialisasi Spark sebagai tool yang akan digunakan untuk melakukan klasifikasi, selanjutnya me-load model yang dihasilkan dari proses *learning*. Setelah model di-load, selanjutnya proses pengambilan data dari Kafka *Broker* menggunakan Kafka *Consumer* yang dalam penelitian ini Kafka *Consumer* menggunakan salah satu modul Spark yaitu Spark Streaming. Selanjutnya data yang diterima tersebut dimasukkan ke dalam fungsi ‘parsing’ yang bertujuan untuk memisahkan data berdasarkan parameter sensor yang dibutuhkan untuk proses klasifikasi. Kemudian data hasil parsing tersebut selanjutnya digunakan oleh fungsi ‘realtimeClassification’ untuk memprediksi kualitas air pada data tersebut (yang saat ini diterima oleh aplikasi). Selanjutnya data hasil proses klasifikasi dengan menggunakan fungsi ‘SendToKafka’, dikirim ke Kafka *Broker* menggunakan metode Kafka *Producer*. Pada langkah ini topik yang digunakan dalam mengirim data air dan hasil klasifikasinya ke Kafka *Broker* adalah ‘wateranalytic’.

c. Visualisasi

Node JS selanjutnya akan men-*subscribe* data hasil klasifikasi pada Kafka Broker menggunakan Kafka *Consumer* dengan topik 'wateranalytic'. Selanjutnya Node JS dengan menggunakan socket.io untuk mengirim data ke *front-end* untuk proses visualisasi pada *web user interface*. Alasan penggunaan Node JS untuk komunikasi data dari *backend* ke *front-end* adalah:

- 1) Kemampuan Node JS untuk membuat aplikasi yang *realtime* dan menggunakan teknologi *push* melalui *websocket*. Hal ini tentu berbeda dengan web server konvensional yang hanya berkomunikasi dengan user hanya ketika ada *request* dari user (melalui browser). Sehingga pembangunan aplikasi yang *realtime* sulit diwujudkan.
- 2) Menggunakan teknologi *event-driven* dan *non-blocking* I/O agar tetap ringan dan efisien untuk aplikasi *data-intensive* secara *realtime* dan dijalankan pada berbagai *device*.

Node JS digunakan untuk mewujudkan aliran data *realtime*. Selanjutnya pada *front-end* akan digunakan web server Apache, PHP, dll untuk kepentingan visualisasi data agar lebih menarik. Secara spesifik digunakan Google Map API untuk visualisasi lokasi node sensor. Highcharts digunakan untuk memvisualisasi data air, highcharts telah mendukung *realtime* visualization untuk data *realtime*. Pada tampilan visualisasi juga ditampilkan informasi indeks pencemaran. Proses aliran data pada proses visualisasi dapat dilihat pada gambar 3.12.

Gambar 3.12. Proses aliran data pada proses visualisasi

Adapun lokasi dari node sensor yang berada pada 7 titik dapat dilihat pada tampilan front end sesuai pada gambar 3.13. Titik pusat dari peta yang ditampilkan adalah (-7.348195, 112.681339).

Gambar 3.13. Tampilan antar muka lokasi node sensor

Sementara tampilan antar muka untuk *realtime* monitoring dan klasifikasi dapat dilihat pada gambar 3.14. Pada gambar 3.14 dapat dilihat grafik data air yang diterima oleh server per 5 detiknya. Data-data air tersebut berasal dari beberapa titik lokasi yang dapat dipilih. Selain itu, juga terdapat informasi Indeks Pencemaran terhadap data-data yang diterima dan telah dianalisa saat ini.

Gambar 3.14. Monitoring dan klasifikasi *realtime*

3.1.7 Collaboration & Processes

Proses ini digunakan untuk menjembatani interaksi antara analis dan sistem. Proses ini digunakan sebagai analis untuk membangun model klasifikasi secara online dan interaktif. Selain itu, melalui bagian ini, juga dapat dilakukan pengaksesan terhadap data air yang tersimpan pada Hadoop HDFS untuk kebutuhan analisa lebih lanjut. Sebagai *interface* dalam bagian ini digunakan Zeppelin. Zeppelin disetup terintegrasi dengan Hadoop, Hive, dan Spark yang telah dibangun sebelumnya. Gambar 3.15 menunjukkan proses pembentukan model klasifikasi menggunakan zeppelin.

Gambar 3.15. Proses *learning* pada Zeppelin menggunakan Python

3.2 DESAIN BIG DATA SERVER

Dalam membangun platform *Big Data* digunakan tiga buah komputer server untuk membangun server *Big Data* dengan skema *multi node cluster*. Master node akan berperan sebagai titik sentral, yang mengatur proses pendistribusian data, pengalokasian cpu dan memory untuk job-job yang akan dijalankan. Dengan implementasi *multi node cluster* permasalahan akan terbatasnya ruang penyimpanan, cpu, maupun memory dapat teratasi. Nantinya jika ketiga server masih kurang, dapat ditingkatkan hanya dengan menambahkan node pada sisi

slave sehingga tidak akan mengganggu sistem secara keseluruhan. Dengan implemementasi *multi node cluster* ini, kemungkinan akan adanya data yang rusak dapat teratasi karena sifat dari Hadoop yang melakukan replikasi data pada semua node. Sementara Hadoop *single-node* diimplementasikan pada satu mesin. Mesin tersebut didesain menjadi master tetapi dapat bekerja juga sebagai slave dan semua proses distribusi dilakukan dalam satu mesin tersebut. Pada Hadoop terbagi menjadi dua layer yaitu layer HDFS yang menjalankan Namenode dan Datanode sedangkan layer MapReduce yang menjalankan Jobtracker dan Tasktracker. Kedua layer ini sangat penting terutama Namenode dan Jobtracker, karena apabila dua bagian ini tidak berjalan maka kerja HDFS dan Mapreduce tidak bisa dijalankan. Pada mesin single node, Datanode dan Tasktracker hanya ada satu, jika memiliki mesin yang banyak maka kedua bagian ini terbentuk pada setiap mesin (*multinode*). Gambar 3.16 menunjukkan desain infrastruktur server dengan skema *multinode cluster*.

Gambar 3.16. Desain infrastruktur server

Ketiga server menggunakan IP Public sehingga dapat diakses dari manapun, konfigurasi alamat IP pada ketiga server dapat dilihat pada tabel 3.5. Throughput masing-masing node ke node lainnya adalah sekitar 1 Gbps.

Tabel 3.5. Konfigurasi alamat IP

Komputer Server	IP Address	Netmask	Gateway
Master Node	202.182.58.21	255.255.255.0	202.182.58.1
Slave Node 1	202.182.58.22	255.255.255.0	202.182.58.1
Slave Node 2	202.182.58.23	255.255.255.0	202.182.58.1

3.3. PENGEMBANGAN *BIG DATA FRAMEWORK*

Dalam rangka memenuhi kebutuhan sistem akan teknologi Big Data, maka dilakukan pembangunan *Big Data Framework*. Beberapa teknologi Big Data yang digunakan adalah Apache Hadoop sebagai komponen utama dalam manajemen data pada *Big Data* yang terdiri dari HDFS, Yarn, dan MapReduce. Apache Hive untuk memaksimalkan dan memudahkan penyimpanan data ke dalam Hadoop HDFS menggunakan *SQL Query Language*, Apache Spark sebagai tool *machine learning* yang handal, baik untuk analisa *batch* maupun *stream*. Apache Kafka untuk proses distribusi data antar lapisan pada *Big Data*.

3.3.1 Apache Hadoop

Apache Hadoop merupakan salah satu distribusi *big data* yang dikembangkan oleh Apache Software Foundation. Apache Hadoop ini dibangun dengan lisensi *free* dan *open source*. Dalam penelitian ini digunakan Apache Hadoop sebagai framework untuk Big Data dengan versi 2.6.0.

Instalasi dan konfigurasi dilakukan pertama kali pada komputer yang bertindak sebagai hadoop-master (single node), untuk menjadikan *multinode cluster* akan dilakukan sinkronisasi dengan komputer slave dan dengan melakukan perubahan konfigurasi yang minimal pada node.

Secara garis besar tahapan instalasi dan konfigurasi Big Data Hadoop adalah sebagai berikut:

- a. Persiapkan komputer baik yang akan dijadikan sebagai node master dan node slave
- b. Install dan konfigurasi sistem Operasi (Debian 8)
- c. Konfigurasi jaringan komputer.
- d. Install Java Development Kit (JDK), OpenSSH-Server, Rsync
- e. Buat user dan group untuk hadoop
- f. Mapping node-node yang ada.
- g. Konfigurasi SSH agar mendukung autologin baik pada master maupun slave.
- h. Instalasi Hadoop dan konfigurasi Hadoop Environment pada node master dan slave.

Untuk dapat menggunakan Hadoop sesuai kebutuhan maka ada beberapa property dasar yang perlu diatur. Diantaranya adalah yarn-site.xml, core-site.xml, hdfs-site.xml, mapred-site.xml dan file bashrc pada user home.

a. yarn-site.xml

```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
  <value>riyadh:8025</value>
</property>
<property>
  <name>yarn.resourcemanager.scheduler.address</name>
  <value>riyadh:8030</value>
</property>
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>true</value>
</property>
<property>
  <name>yarn.nodemanager.remote-app-log-dir</name>
  <value>/tmp/logs</value>
</property>
<property>
  <name>yarn.log-aggregation.retain-seconds</name>
  <value>259200</value>
</property>
<property>
  <name>yarn.log-aggregation.retain-check-interval-seconds</name>
  <value>3600</value>
</property>
<property>
  <name>yarn.log.server.url</name>
  <value>http://202.182.58.21:19888/jobhistory/logs/</value>
</property>
</configuration>
```

b. core-site.xml

```
<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hduser/SEMAR_Framework/hadoop_tmp</value>
</property>
<property>
  <name>fs.default.name</name>
  <value>hdfs://riyadh:54310</value>
</property>
<property>
  <name>hadoop.proxyuser.hduser.hosts</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.proxyuser.hduser.groups</name>
  <value>*</value>
</property>
</configuration>
```

b. hdfs-site.xml

```
<configuration>
<property>
  <name>dfs.replication</name>
  <value>3</value>
</property>
<property>
  <name>dfs.permissions.enabled</name>
  <value>>false</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/home/hduser/SEMAR_Framework/hadoop_store/hdfs/
    namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/home/hduser/SEMAR_Framework/hadoop_store/hdfs/
    datanode</value>
</property>
</configuration>
```

c. mapred-site.xml

```
<configuration>
<property>
  <name>mapred.job.tracker</name>
  <value>riyadh:54311</value>
</property>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
<property>
  <name>mapreduce.jobhistory.webapp.address</name>
  <value>202.182.58.21:19888</value>
</property>
<property>
  <name>mapreduce.jobhistory.address</name>
  <value>riyadh:10020</value>
</property>
<property>
  <name>mapreduce.jobhistory.intermediate-done-dir</name>
  <value>/mr-history/tmp</value>
</property>
<property>
  <name>mapreduce.jobhistory.done-dir</name>
  <value>/mr-history/done</value>
</property>
</configuration>
```

d. Hadoop Environment (~/.bashrc)

```
export JAVA_HOME=/home/hduser/SEMAR_Framework/javasun
export PATH=$PATH:$JAVA_HOME/bin
export HADOOP_INSTALL=/home/hduser/SEMAR_Framework/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
export HADOOP_CONF_DIR=$HADOOP_INSTALL/etc/hadoop
export HADOOP_HOME=$HADOOP_INSTALL
export SCALA_HOME=/home/hduser/SEMAR_Framework/scala
export PATH=$PATH:$SCALA_HOME/bin
```


e. Jalankan service Hadoop.

```
hduser@HadoopMaster:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [hadoopmaster]
hadoopmaster: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hduser-namenode-HadoopMaster.out
HadoopMaster: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-datanode-HadoopMaster.out
HadoopSlave1: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-datanode-HadoopSlave1.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hduser-secondarynamenode-HadoopMaster.out
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hduser-resourcemanager-HadoopMaster.out
HadoopMaster: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-nodemanager-HadoopMaster.out
HadoopSlave1: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-nodemanager-HadoopSlave1.out
```

Gambar 3.17. Menjalankan service Hadoop

Gambar 3.18 menunjukkan service Hadoop yang berjalan pada node master (kanan) dan slave (kiri).

```
hduser@HadoopMaster:~$ jps
4085 NodeManager
3448 NameNode
3798 SecondaryNameNode
4433 Jps
3580 DataNode
3953 ResourceManager

hduser@HadoopSlave1:~$ jps
3201 NodeManager
3557 Jps
3061 DataNode
hduser@HadoopSlave1:~$
```

Gambar 3.18. Service Hadoop pada node master dan slave

Gambar 3.19 menunjukkan informasi Hadoop yang dapat diakses melalui web. Halaman web ini berisi informasi tentang status cluster Hadoop yang telah dipasang, informasi datanode, maupun data-data yang tersimpan pada Hadoop HDFS. Hadoop web UI ini dapat diakses melalui port 50070.

Overview 'HadoopMaster:9000' (active)	
Started:	Thu Jul 21 10:11:26 WIB 2016
Version:	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
Compiled:	2014-11-13T21:10Z by jenkins from (detached from c349649)
Cluster ID:	CID-4f1b72ce-4539-46b3-b42c-780ba93c5f31
Block Pool ID:	BP-1926484997-192.168.1.1-1466665282599

Gambar 3.19. Hadoop Web UI

Gambar 3.20 menunjukkan informasi tentang node-node slave pada cluster Hadoop.

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
HadoopSlave1 (192.168.1.2:50010)	0	In Service	47.04 GB	15.37 MB	9.66 GB	37.36 GB	1076	15.37 MB (0.03%)	0	2.6.0
HadoopMaster (192.168.1.1:50010)	1	In Service	191.17 GB	15.43 MB	17.48 GB	173.67 GB	1076	15.43 MB (0.01%)	0	2.6.0

Gambar 3.20. Informasi Datanode

3.3.2 Apache Hive

Apache Hive merupakan salah satu bagian dari proyek *Big Data* Apache Software Foundation yang lisensi *free* dan *open source* yang merupakan hibah dari Facebook, Inc. Apache Hive ini dikembangkan untuk memudahkan penanganan terhadap data yang terstruktur dan semi-terstruktur. Dibanding dengan penanganan data terstruktur biasa yang disimpan dalam bentuk file teks, dengan Apache Hive dapat dilakukan penyimpanan data terstruktur dengan menggunakan struktur database RDBMS pada umumnya. Proses penyimpanan dan pengambilan datanya pun menggunakan perintah-perintah SQL yang sudah familiar bagi pengguna database RDBMS.

Secara garis besar instalasi Apache Hive adalah sebagai berikut:

- Pastikan instalasi Java Development Kit (JDK) berhasil
- Pastikan instalasi Apache Hadoop berhasil
- Instalasi dan Konfigurasi Environment Apache Hive
Hive environment (`~/bashrc`)

```
export HIVE_HOME=/home/hduser/SEMAR_Framework/hive
export HIVE_CONF_DIR=/home/hduser/SEMAR_Framework/hive/conf
export PATH=$PATH:$HIVE_HOME/bin
export HADOOP_USER_CLASSPATH_FIRST=true
export CLASSPATH=$CLASSPATH:/home/hduser/SEMAR_Framework/hadoop/
lib/*:.
export CLASSPATH=$CLASSPATH:/home/hduser/SEMAR_Framework/hive/lib/*:.
```

- d. Instalasi Postgre-SQL
- e. Buat tabel pada Postgre-SQL menggunakan skema hive dan berikan otoritas pada user hive untuk mengakses secara penuh.
- f. Konfigurasi Metastore Hive menggunakan Postgre-SQL

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:postgresql://riyadh/metastore</value>
</property>
<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.postgresql.Driver</value>
</property>
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>riyadharridha</value>
</property>
<property>
  <name>hive.metastore.warehouse.dir</name>
  <value>/user/hive/warehouse</value>
</property>
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://riyadh:9083</value>
</property>
<property>
  <name>datanucleus.autoCreateSchema</name>
  <value>>false</value>
</property>
<property>
  <name>hive.execution.engine</name>
  <value>mr</value>
</property>
```

- g. Pastikan Instalasi Apache Hive telah berhasil
- h. Jalankan service Apache Hive (hive hiveserver2 dan hive metastore)

Pembuatan table pada Apache Hive untuk menampung data sensor air. *Query* hive mayoritas sama dengan *query* SQL konvensional. Hal ini menyebabkan banyak pengguna yang baru bermigrasi ke Hadoop merasa nyaman menggunakan Hive. Gambar 3.21 menunjukkan *query* pada hive dalam membuat tabel untuk menampung data air dari sensor.

```
hive> CREATE EXTERNAL TABLE projectiotwater(
> tanggal STRING,
> jam STRING,
> orp STRING,
> ph STRING,
> ec STRING,
> tds STRING,
> sal STRING,
> sg STRING,
> do STRING,
> temp STRING)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY '\n'
> LINES TERMINATED BY '\n'
> STORED AS TEXTFILE;
```

Gambar 3.21. Pembuatan tabel menggunakan Hive

Untuk mengakses data yang tersimpan pada Hadoop HDFS juga dapat menggunakan *query* hive, yang sama dengan *query* pada database konvensional. Query tabel pada Apache Hive untuk melihat data sensor air terlihat pada gambar 3.22.

```
hive> select * from projectiotwater limit 100;
OK
2016-06-24 16:13:56 418.2 8.108 304.2 164 0.14 1.000 5.98 22.50
2016-06-24 16:16:58 460.3 7.420 304.7 164 0.14 1.000 5.85 22.50
2016-06-24 16:23:04 429.0 6.481 304.2 164 0.14 1.000 6.66 22.25
2016-06-24 17:08:07 462.5 7.276 302.2 163 0.14 1.000 6.05 22.25
2016-06-25 00:38:35 483.6 7.060 338.4 182 0.16 1.000 6.58 22.25
2016-06-25 00:39:05 483.7 7.115 338.4 182 0.16 1.000 6.58 22.25
2016-06-25 00:39:35 483.7 7.115 338.4 182 0.16 1.000 6.56 22.25
2016-06-25 00:40:06 484.1 7.398 338.4 182 0.16 1.000 6.51 22.50
2016-06-25 00:40:36 484.0 7.658 338.4 182 0.16 1.000 6.51 22.50
2016-06-25 00:41:06 484.0 7.658 338.4 182 0.16 1.000 6.51 22.50
2016-06-25 00:41:36 484.0 7.778 338.4 182 0.16 1.000 6.49 22.50
2016-06-25 00:42:06 484.1 7.775 338.4 182 0.16 1.000 6.52 22.50
2016-06-25 00:42:36 484.1 7.773 338.3 182 0.16 1.000 6.52 22.50
2016-06-25 00:43:06 484.6 7.688 338.3 182 0.16 1.000 6.47 22.25
2016-06-24 17:08:37 462.3 7.296 302.2 163 0.14 1.000 6.09 22.50
2016-06-25 00:43:36 484.5 7.538 338.4 182 0.16 1.000 6.42 22.25
2016-06-25 00:44:06 484.6 7.421 338.5 182 0.16 1.000 6.42 22.50
2016-06-25 00:44:36 484.8 7.282 338.5 182 0.16 1.000 6.41 22.25
2016-06-25 00:45:06 485.2 7.188 338.5 182 0.16 1.000 6.37 22.50
2016-06-25 00:45:36 485.7 7.188 338.5 182 0.16 1.000 6.44 22.50
2016-06-25 00:46:06 485.5 7.190 338.5 182 0.16 1.000 6.37 22.50
2016-06-25 00:46:36 485.4 7.196 338.5 182 0.16 1.000 6.30 22.25
2016-06-25 00:47:06 485.3 7.214 338.5 182 0.16 1.000 6.30 22.50
2016-06-25 00:47:36 485.1 7.231 338.4 182 0.16 1.000 6.34 22.50
2016-06-25 00:48:06 485.1 7.291 338.4 182 0.16 1.000 6.26 22.50
2016-06-24 17:09:07 462.4 7.464 302.2 163 0.14 1.000 6.14 22.50
2016-06-25 00:48:36 485.2 7.476 338.4 182 0.16 1.000 6.21 22.25
2016-06-25 00:49:06 485.5 7.647 338.5 182 0.16 1.000 6.28 22.50
2016-06-25 00:49:36 485.8 7.815 338.5 182 0.16 1.000 6.19 22.50
2016-06-25 00:50:06 485.9 7.876 338.6 182 0.16 1.000 6.26 22.25
2016-06-25 00:50:36 486.0 7.877 338.7 182 0.16 1.000 6.21 22.50
2016-06-25 00:51:06 486.5 7.872 338.6 182 0.16 1.000 6.19 22.50
2016-06-25 00:51:36 486.3 7.858 338.6 182 0.16 1.000 6.15 22.50
```

Gambar 3.22. Hasil Query pada Apache Hive

3.3.3 Apache Spark

Apache Spark merupakan salah satu bagian dari proyek *Big Data* Apache Software Foundation yang menangani analisis data pada *Big Data*.

Secara garis besar instalasi Apache Spark adalah sebagai berikut:

- Pastikan keberadaan instalasi Java Development Kit
- Install dan konfigurasi environment Scala
- Install dan konfigurasi environment Apache Spark

Konfigurasi environment Scala dan Apache Spark (~/.bashrc)

```
export SPARK_HOME=/home/hduser/SEMAR_Framework/spark
export PATH=$PATH:$SPARK_HOME/bin
```

- Konfigurasi untuk spark-default.conf

```
spark.blockManager.port      38000
spark.broadcast.port          38001
spark.driver.port             38002
spark.executor.port           38003
spark.fileserver.port         38004
spark.replClassServer.port    38005
```

- Setup Apache Spark pada cluster
- Jalankan service Apache Spark

Service Spark pada Node Master (kanan) dan node slave (kiri) dapat dilihat pada Gambar 3.23.

```
hduser@ubuntu1:~$ jps
2282 ResourceManager
1921 NameNode
2111 SecondaryNameNode
2805 Worker
2908 Jps

hduser@ubuntu2:~$ jps
2166 Worker
1936 NodeManager
2230 Jps
1792 DataNode
```

Gambar 3.23. Service Spark (Worker) pada node master dan slave

BAB 4

EKSPERIMEN DAN ANALISIS

Pada bab ini akan dijabarkan mengenai parameter eksperimen, skenario ujicoba, tempat dan waktu eksperimen, spesifikasi peralatan uji coba dan analisa hasil eksperimen.

4.1 PARAMETER EKSPERIMEN

Dalam penelitian ini terdapat beberapa parameter eksperimen, diantaranya: pengujian performa klasifikasi air dengan metode Indeks Pencemaran, pengujian performa algoritma klasifikasi yang digunakan adalah Decision Tree dan Support Vector Machine, pengujian performa sistem SEMAR ada proses *realtime*.

Input dari penelitian ini adalah data-data hasil uji laboratorium, sensor *live* dari PDAM Surya Sembada Surabaya, dan data sensor yang dipasang di sepanjang sungai di Kota Surabaya. Sementara output nya adalah model klasifikasi, hasil klasifikasi *realtime*, dan waktu pemrosesan data oleh sistem.

Parameter yang digunakan dalam algoritma Decision Tree dalam penelitian ini adalah:

- a. Menggunakan algoritma CART (Classification and Regression Tree)
- b. Kriteria untuk mengukur kualitas split tree menggunakan gini impurity.
- c. Tidak ada batasan kedalaman pohon keputusan yang dibuat.

Parameter yang digunakan pada algoritma Support Vector Machine dalam penelitian ini adalah:

- a. Menggunakan kernel linear
- b. Optimasi parameter menggunakan algoritma *grid search* pada SVM (Linear).C menggunakan *grid/range* minimum 0, maksimum 100, step 100 dan skala linear.
- c. Hasil optimasi yang digunakan, parameter $C = 25$ untuk dataset uji laboratorium, dan $C = 11$ untuk dataset sensor *live*.

4.2 KARAKTERISTIK DATA

Data yang digunakan untuk melakukan proses klasifikasi berasal dari PDAM Swasembada Surabaya yang terdiri dari data uji laboratorium dan data *live* sensor. Data uji laboratorium yang merupakan uji laboratorium harian dari tahun 2014 hingga tahun 2016 dengan 1347 sampel dan 20 atribut yang terdiri dari Tanggal, Suhu, Kekeruhan, Warna, TSS, pH, Alkalinitas, CO₂ Bebas, DO, Nitrit, Amonia, Tembaga, Fosfat, Sulfida, Besi, Kromium Hexavalen, Mangan, Seng, Timbal dan COD. Data live sensor merupakan data yang berasal dari sensor live PDAM yang dipasang pada pintu air Ngagel. Data tersebut diambil dari bulan Maret hingga bulan Agustus 2016 dengan 205.720 sampel dan 6 atribut yaitu Tanggal, Jam, Kekeruhan, TSS, pH, DO, dan Suhu.

Sementara untuk data yang digunakan dalam pengujian klasifikasi *realtime* merupakan data yang berasal dari node-node sensor yang dipasang pada sepanjang sungai di Kota Surabaya.

4.3 TEMPAT UJI COBA

Eksperimen dilaksanakan di laboratorium mahasiswa lantai 9 gedung pascasarjana Politenik Elektronika Negeri Surabaya untuk proses pembangunan aplikasi, implementasi server, dan *remote* server. Di kantor PDAM Surya Sembada Kota Surabaya untuk mendapatkan data air uji laboratorium dan data sensor *live*. Di kantor Kementerian Lingkungan Hidup Kota Surabaya untuk mendapatkan data pengukuran kualitas air sungai di Kota Surabaya. Tujuh titik lokasi node sensor pada sungai di sepanjang sungai Surabaya.

4.4 WAKTU UJI COBA

Penelitian berlangsung selama 22 bulan. Jadwal pelaksanaan kegiatan penelitian dijelaskan pada table 4.1 berikut:

Tabel 4.1. Jadwal Pelaksanaan

Mulai: Agustus 2015

Berakhir: Juni 2017

No	Rincian Kegiatan	2015		2016				2017	
		K3	K4	K1	K2	K3	K4	K1	K2
1	Studi Pustaka								
2	Pengumpulan Data								
3	Perancangan dan Desain Sistem								
4	Integrasi Sistem								
5	Pengujian Sistem								
6	Pelaporan dan Publikasi								

4.5 SPESIFIKASI PERALATAN UJI COBA

Pada penelitian ini digunakan beberapa peralatan yaitu: 3 buah server untuk membangun *Big Data* dengan skema *multi node cluster*, dan 1 laptop untuk remote server dan pembangunan aplikasi. Selain itu juga digunakan beberapa perangkat lunak yang mendukung selama proses penelitian. Perangkat lunak yang digunakan mayoritas diimplementasikan pada komputer server. Perangkat lunak yang digunakan pada komputer server maupun pada laptop berbasis *free software* dan *open source*. Detail spesifikasi *hardware* dan *software* yang digunakan dalam penelitian ini dapat dilihat masing-masing pada tabel 4.2 dan 4.3.

Tabel 4.2. Spesifikasi Hardware

@Server		Laptop	
Hardware			
CPU	Intel Xeon E3-1220 v3	CPU	Intel Core i3-
	3.1 GHz		3217U
Core	4	Core	4
RAM	16 GB	RAM	8 GB
HDD	1 TB	HDD	720 GB

Tabel 4.3. Spesifikasi Software

@Server		Laptop	
Software			
OS	Debian GNU/Linux 8.6 (Jessie)	OS	Ubuntu GNU/Linux 14.04 (Trusty Tahr)
Hadoop	2.6.0	Eclipse	4.3 (Kepler)
Hive	1.2.1	OpenSSH-client	1:6.6p1-2ubuntu1
Kafka	2.10-0.10.1.0		
Spark	1.6.1		
Mosquitto	1.6		
Zeppelin	0.6.0		
Node JS	6.9.4		
JDK	1.7.0 versi SUN		
OpenSSH-server	1:6.7p1-5+deb8u3		

4.6 HASIL EKSPERIMEN

Pada subbab ini akan dijabarkan hasil eksperimen yang dilakukan berdasarkan beberapa parameter yang telah dikemukakan. Beberapa eksperimen yang dilakukan adalah uji coba program yang telah dibangun pada tiap lapisan dari *Internet of Things* dan penentuan performa dari algoritma klasifikasi yang digunakan pada *Big Data Analytic* yaitu Decision Tree dan Support Vector Machine.

4.6.1 Uji coba program

Pada tahap ini akan dilakukan uji coba menjalankan program pada tiap lapisan dari platform *Internet of Things* yang telah dibuat. Program-program yang dijalankan mewakili dari tiap lapisan dari 7 lapisan platform IoT yang dibangun.

a. Lapisan *Physical Devices & Controllers*

Gambar 4.1 menunjukkan aplikasi yang berjalan pada node sensor. Data yang ditampilkan merupakan data gabungan yang akan dikirim ke server menggunakan protokol MQTT. Gambar 4.1 mewakili aplikasi yang dibangun pada lapisan ke-1 dari platform IoT yang dibangun.

```
2017-04-04,00:12:02,226.6,6.635,508.3,274,0.24,1,2.92,28.39,-7.299487,112.734737,joyoboyo
2017-04-04,00:12:07,226.5,6.863,507.8,274,0.24,1,2.92,28.15,-7.299487,112.734737,joyoboyo
2017-04-04,00:12:12,227.2,7.075,507.5,274,0.24,1,2.93,28.39,-7.299487,112.734737,joyoboyo
2017-04-04,00:12:17,229.2,7.247,507.5,274,0.24,1,2.92,28.39,-7.299487,112.734737,joyoboyo
2017-04-04,00:12:22,231.1,7.279,507.2,273,0.24,1,2.91,28.15,-7.299487,112.734737,joyoboyo
```

Gambar 4.1. Aplikasi yang berjalan pada node sensor

b. Lapisan *Edge Computing*

Gambar 4.2 menunjukkan data yang diterima oleh MQTT *Broker* di sisi komputer server. Dari gambar 4.2 terlihat bahwa data masuk dari beberapa titik node sensor yang berbeda yaitu Ngagel, Pintu Air Jagir, Prapatan Prapen, Stikom, Joyoboyo, Wonorejo dan Karang Pilang. Gambar 4.2 mewakili aplikasi yang berjalan pada lapisan ke-3 dari platform IoT yang dibangun.

```
hduser@riyadh:~/SEMAR_Framework/appSEMAR/server$ python mqttBridge.py
log4j:WARN No appenders could be found for logger (kafka.utils.VerifiableProperties).
log4j:WARN Please initialize the log4j system properly.
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
2017-04-04,00:24:06,290.6,5.818,513.5,277,0.24,1,3.39,28.39,-7.297746,112.741606,ngagel
2017-04-04,00:24:07,256.2,5.988,509.3,275,0.24,1,3.74,28.15,-7.301354,112.743381,pintuairjagir
2017-04-04,00:24:08,312.4,6.607,505.3,272,0.24,1,3.77,28.64,-7.305796,112.761288,prapatanprapen
2017-04-04,00:24:09,270.3,6.132,517.6,279,0.25,1,4.17,28.39,-7.310150,112.783255,stikom
2017-04-04,00:24:10,219.4,6.686,513.9,277,0.24,1,2.92,28.39,-7.299487,112.734737,joyoboyo
2017-04-04,00:24:10,431.3,6.49,569.7,307,0.27,1,3.89,29.38,-7.305158,112.806950,wonorejo
2017-04-04,00:24:11,294.5,5.818,513.4,277,0.24,1,3.35,28.39,-7.297746,112.741606,ngagel
```

Gambar 4.2. Aplikasi pada MQTT *Broker*.

c. Lapisan *Data Accumulation*

Gambar 4.3 menunjukkan proses penyimpanan data pada Hadoop HDFS menggunakan *query* Hive. *Query* Hive dieksekusi menggunakan Spark SQL. Info yang ditampilkan menunjukkan data yang diterima dari Kafka *Broker* dan

kemudian disimpan pada HDFS. Gambar 4.3, 4.4, dan 4.5 mewakili aplikasi yang berjalan pada lapisan ke-4 dari platform IoT yang dibangun.

```
hduser@riyadh:~/SEMAR_Framework/appSEMAR/server$ spark-submit --jars ~/SEMAR_Framework/spark/lib/spark-streaming-mqtt-assembly_2.10-1.6.1.jar --master yarn-client mqttToHive.py
****LOAD DATA INTO HDFS****
=====
=====RESET LOG FILE=====
Komunikasi MQTT Berhasil
=====
2017-04-04 01:14:03 =====
2017-04-04 01:14:03 =====
2017-04-04 01:14:04 =====
2017-04-04 01:14:04 =====
2017-04-04 01:14:05 =====
2017-04-04 01:14:05 =====
2017-04-04 01:14:06 =====
2017-04-04,01:14:05,263.6.883,498.2,269,0.24,1,2.28,27.9,-7.299487,112.734737,joyoboyo
2017-04-04,01:14:05,446.5,6.681,491.5,265,0.23,1,3.71,28.39,-7.305796,112.761288,prapatanprapen
2017-04-04,01:14:05,387.9,6.825,568.9,307,0.27,1,3.09,28.88,-7.305158,112.806950,wonorejo
2017-04-04,01:14:05,513.9,7.856,513.2,277,0.24,1,2.87,28.39,-7.297746,112.741606,ngagel
2017-04-04,01:14:05,298.5,7.276,520,280,0.25,1,3.22,27.9,-7.301354,112.743381,pintuairjagir
2017-04-04,01:14:05,449.5,5.859,514.5,277,0.24,1,3.32,28.15,-7.310150,112.783255,stikom
2017-04-04 01:14:06 =====
2017-04-04 01:14:07 =====
2017-04-04 01:14:07 =====
2017-04-04 01:14:08 =====
2017-04-04 01:14:08 =====
2017-04-04 01:14:09 =====
2017-04-04 01:14:09 =====
2017-04-04 01:14:10 =====
2017-04-04 01:14:10 =====
2017-04-04 01:14:11 =====
2017-04-04,01:14:10,263.6.883,498.2,269,0.24,1,2.35,27.9,-7.299487,112.734737,joyoboyo
2017-04-04,01:14:10,446.8,6.681,491.5,265,0.23,1,3.68,28.39,-7.305796,112.761288,prapatanprapen
2017-04-04,01:14:10,387.8,6.726,568.9,307,0.27,1,3.09,28.88,-7.305158,112.806950,wonorejo
2017-04-04,01:14:10,514.1,7.759,513.3,277,0.24,1,2.85,28.39,-7.297746,112.741606,ngagel
2017-04-04,01:14:10,298.9,7.29,519.8,280,0.25,1,3.23,27.9,-7.301354,112.743381,pintuairjagir
2017-04-04,01:14:10,448.7,5.962,514.5,277,0.24,1,3.31,28.15,-7.310150,112.783255,stikom
2017-04-04 01:14:11 =====
2017-04-04 01:14:12 =====
2017-04-04 01:14:12 =====
2017-04-04 01:14:13 =====
2017-04-04 01:14:13 =====
2017-04-04 01:14:14 =====
2017-04-04 01:14:14 =====
2017-04-04 01:14:15 =====
2017-04-04 01:14:15 =====
2017-04-04 01:14:16 =====
2017-04-04,01:14:15,263.8,6.891,498.2,269,0.24,1,2.32,27.9,-7.299487,112.734737,joyoboyo
2017-04-04,01:14:15,446.8,6.683,491.5,265,0.23,1,3.74,28.39,-7.305796,112.761288,prapatanprapen
2017-04-04,01:14:15,387.9,6.54,569,307,0.27,1,3.05,28.88,-7.305158,112.806950,wonorejo
2017-04-04,01:14:15,514.4,7.649,513.3,277,0.24,1,2.88,28.39,-7.297746,112.741606,ngagel
```

Gambar 4.3. Aplikasi penyimpanan data pada Hadoop HDFS

Gambar 4.4 merupakan informasi tambahan pada proses *background* tentang keberhasilan *load* data ke dalam Hadoop HDFS

```
Loading data to table default.watermonitoringku
Table default.watermonitoringku stats: [numFiles=3638, totalSize=362866162]
OK
Loading data to table default.watermonitoringku
Table default.watermonitoringku stats: [numFiles=3639, totalSize=362867788]
OK
Loading data to table default.watermonitoringku
Table default.watermonitoringku stats: [numFiles=3640, totalSize=362869423]
OK
```

Gambar 4.4. Notifikasi *load* data ke tabel HDFS

Gambar 4.5 merupakan proses query data secara manual menggunakan *Command Line Interface* Hive untuk melihat data yang tersimpan pada Hadoop HDFS. Proses ini dilakukan secara *batch*. Untuk memasuki mode Hive CLI cukup dengan mengetikkan command 'hive' pada terminal linux.

```
hive> select * from watermonitoringku limit 30;
OK
2017-01-30 13:47:55 297.6 7.255 519.3 280 0.25 1 3.15 28.15 -7.301354 112.743381 pintuairjagir
2017-01-30 13:47:55 382.9 5.864 568.4 306 0.27 1 3.51 28.64 -7.305158 112.806950 wonorejo
2017-01-30 13:47:55 447.1 6.076 514.7 277 0.24 1 3.4 28.15 -7.310150 112.783255 stikom
2017-01-30 13:47:55 508.7 7.43 513.2 277 0.24 1 2.9 28.39 -7.297746 112.741606 ngagel
2017-01-30 13:47:55 438.4 6.7 491.4 265 0.23 1 3.59 28.39 -7.305796 112.761288 prapatanprapen
2017-01-30 14:05:17 390.1 6.179 568.1 306 0.27 1 2.51 28.64 -7.305158 112.806950 wonorejo
2017-01-30 14:05:17 271.9 7.675 499.2 269 0.24 1 2.78 28.15 -7.299487 112.734737 joyoboyo
2017-01-30 14:05:17 287.1 6.924 526.4 284 0.25 1 3.06 28.15 -7.301354 112.743381 pintuairjagir
2017-01-30 14:05:18 381.2 7.268 515.9 278 0.24 1 3.67 28.64 -7.310150 112.783255 stikom
2017-01-30 14:05:18 481.4 5.888 514.9 278 0.24 1 2.86 28.39 -7.297746 112.741606 ngagel
2017-01-30 14:05:18 418.1 7.624 495.5 267 0.23 1 3.94 28.15 -7.305796 112.761288 prapatanprapen
2017-01-30 14:05:18 388.7 6.213 568 306 0.27 1 2.51 28.64 -7.305158 112.806950 wonorejo
2017-01-30 14:05:18 264.3 7.791 499.2 269 0.24 1 2.72 28.15 -7.299487 112.734737 joyoboyo
2017-01-30 14:05:18 287.4 6.925 526.5 284 0.25 1 3.13 27.9 -7.301354 112.743381 pintuairjagir
2017-01-30 14:05:19 381.7 7.268 515.7 278 0.24 1 3.65 28.64 -7.310150 112.783255 stikom
2017-01-30 14:05:19 417.7 7.697 495.4 267 0.23 1 3.88 28.15 -7.305796 112.761288 prapatanprapen
2017-01-30 14:05:19 481.6 5.948 514.9 278 0.24 1 2.83 28.15 -7.297746 112.741606 ngagel
2017-01-30 14:05:19 386.9 6.245 568 306 0.27 1 2.53 28.64 -7.305158 112.806950 wonorejo
2017-01-30 14:05:19 261.3 7.886 499.2 269 0.24 1 2.72 28.15 -7.299487 112.734737 joyoboyo
2017-01-30 14:05:19 207.7 6.999 526.3 284 0.25 1 3.13 27.9 -7.301354 112.743381 pintuairjagir
2017-01-30 14:05:20 381.8 7.417 515.7 278 0.24 1 3.65 28.39 -7.310150 112.783255 stikom
2017-01-30 14:05:20 417.8 7.762 495.5 267 0.23 1 3.94 28.39 -7.305796 112.761288 prapatanprapen
2017-01-30 14:05:20 481.9 6.008 514.9 278 0.24 1 2.82 28.39 -7.297746 112.741606 ngagel
2017-01-30 14:05:20 386.7 6.274 568.1 306 0.27 1 2.57 28.88 -7.305158 112.806950 wonorejo
2017-01-30 14:05:20 261.3 7.886 499.2 269 0.24 1 2.75 28.15 -7.299487 112.734737 joyoboyo
2017-01-30 14:05:20 287.8 6.927 526.3 284 0.25 1 3.11 28.15 -7.301354 112.743381 pintuairjagir
2017-01-30 14:05:21 381.8 7.561 515.7 278 0.24 1 3.68 28.39 -7.310150 112.783255 stikom
2017-01-30 14:05:21 418.4 7.811 495.4 267 0.23 1 3.97 28.39 -7.305796 112.761288 prapatanprapen
2017-01-30 14:05:21 482.1 6.06 514.9 278 0.24 1 2.83 28.39 -7.297746 112.741606 ngagel
2017-01-30 14:05:21 386.7 6.233 568.1 306 0.27 1 2.56 28.64 -7.305158 112.806950 wonorejo
Time taken: 0.145 seconds, Fetched: 30 row(s)
```

Gambar 4.5. Data yang tersimpan pada Hadoop HDFS

d. Lapisan *Data Abstraction*

Gambar 4.6 merupakan proses aliran data menuju ke proses klasifikasi *realtime*. Pada gambar 4.6 terlihat aliran data yang diterima dari node yang akan didistribusikan ke klasifikasi *realtime*. Gambar 4.6 mewakili aplikasi yang dibangun pada lapisan ke-5 dari platform IoT yang dibangun.

```
log4j:WARN No appenders could be found for logger (kafka.utils.VerifiableProperties).
log4j:WARN Please initialize the log4j system properly.
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
2017-04-04,00:24:06,290.6,5.818,513.5,277,0.24,1,3.39,28.39,-7.297746,112.741606,ngagel
2017-04-04,00:24:07,256.2,5.988,509.3,275,0.24,1,3.74,28.15,-7.301354,112.743381,pintuairjagir
2017-04-04,00:24:08,312.4,6.607,505.3,272,0.24,1,3.77,28.64,-7.305796,112.761288,prapatanprapen
2017-04-04,00:24:09,270.3,6.132,517.6,279,0.25,1,4.17,28.39,-7.310150,112.783255,stikom
2017-04-04,00:24:10,219.4,6.686,513.9,277,0.24,1,2.92,28.39,-7.299487,112.734737,joyoboyo
2017-04-04,00:24:10,431.3,6.49,569.7,307,0.27,1,3.89,29.38,-7.305158,112.806950,wonorejo
2017-04-04,00:24:11,294,5.818,513.4,277,0.24,1,3.35,28.39,-7.297746,112.741606,ngagel
2017-04-04,00:24:12,256.2,5.988,509.3,275,0.24,1,3.75,28.15,-7.301354,112.743381,pintuairjagir
2017-04-04,00:24:13,313.1,6.773,505.1,272,0.24,1,3.74,28.39,-7.305796,112.761288,prapatanprapen
```

Gambar 4.6. Aliran data menuju klasifikasi *realtime*

e. Lapisan *Application*

Gambar 4.7 merupakan hasil dari klasifikasi *realtime*. Info yang ditampilkan adalah data air yang telah diklasifikasi yang kemudian dikirim ke Kafka *Broker*

menggunakan Kafka *Producer*. Gambar ini mewakili proses klasifikasi *realtime* yang terdapat pada lapisan ke-6 dari platform IoT yang dibangun.

```
hduser@riyadh:~/SEMAR_Framework/appSEMAR/server$ spark-submit --jars ~/SEMAR_Framework/spark/1
ib/spark-streaming-kafka-assembly_2.10-1.6.1.jar --master yarn-client serverBigDataAnalytic.py
===== 2017-04-04 01:42:31 =====
===== 2017-04-04 01:42:32 =====
Data Sent: {'orp': '464.8', 'do': '3.05', 'sal': '0.25', 'ec': '516.8', 'tds': '279', 'tanggal':
1': '2017-04-04', 'lat': '-7.297746', 'loc': 'ngagel', 'temp': '28.39', 'long': '112.741606',
'label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '6.622', 'sg': '1', 'result': '2'}
Data Sent: {'orp': '421.3', 'do': '3.8', 'sal': '0.23', 'ec': '496', 'tds': '267', 'tanggal':
'2017-04-04', 'lat': '-7.305796', 'loc': 'prapatanprapen', 'temp': '28.15', 'long': '112.7612
88', 'label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '7.084', 'sg': '1', 'result': '2'}
Data Sent: {'orp': '385.7', 'do': '3.65', 'sal': '0.24', 'ec': '514.8', 'tds': '278', 'tanggal
1': '2017-04-04', 'lat': '-7.310150', 'loc': 'stikom', 'temp': '28.88', 'long': '112.783255',
'label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '7.579', 'sg': '1', 'result': '2'}
Data Sent: {'orp': '402.3', 'do': '3.23', 'sal': '0.27', 'ec': '568.3', 'tds': '306', 'tanggal
1': '2017-04-04', 'lat': '-7.305158', 'loc': 'wonorejo', 'temp': '28.88', 'long': '112.806950',
'label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '6.71', 'sg': '1', 'result': '2'}
Data Sent: {'orp': '256.3', 'do': '2.6', 'sal': '0.24', 'ec': '500', 'tds': '270', 'tanggal':
'2017-04-04', 'lat': '-7.299487', 'loc': 'joyoboyo', 'temp': '28.39', 'long': '112.734737', '
label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '7.009', 'sg': '1', 'result': '2'}
===== 2017-04-04 01:42:33 =====
Data Sent: {'orp': '328.2', 'do': '2.98', 'sal': '0.24', 'ec': '505.9', 'tds': '273', 'tanggal
1': '2017-04-04', 'lat': '-7.301354', 'loc': 'pintuairjagir', 'temp': '28.15', 'long': '112.74
3381', 'label': 'Lightly Polluted', 'jam': '01:42:31', 'ph': '6.39', 'sg': '1', 'result': '2'}
===== 2017-04-04 01:42:34 =====
===== 2017-04-04 01:42:35 =====
===== 2017-04-04 01:42:36 =====
===== 2017-04-04 01:42:37 =====
Data Sent: {'orp': '465', 'do': '3.01', 'sal': '0.25', 'ec': '516.8', 'tds': '279', 'tanggal'
```

Gambar 4.7. Proses pengiriman hasil klasifikasi data air ke Kafka *Broker*.

Gambar 4.8 merupakan data yang diterima oleh Node JS yang akan diproses pada *front end* untuk divisualisasi. Gambar 4.8, 4.9, 4.10, 4.11 mewakili proses visualisasi data yang terdapat pada lapisan ke-6 (*Application*) dari platform IoT yang dibangun.

```
hduser@riyadh:~/SEMAR_Framework/webAppSEMAR$ node app.js
listening on *:3001
{ topic: 'wateranalytic',
  value: '{"orp": "241.5", "do": "2.47", "sal": "0.24", "ec": "499.4", "tds": "26
9", "tanggal": "2017-04-04", "lat": "-7.299487", "loc": "joyoboyo", "temp": "28.1
5", "long": "112.734737", "label": "Lightly Polluted", "jam": "01:47:36", "ph": "
7.431", "sg": "1", "result": "2"}',
  offset: 402,
  partition: 0,
  key: -1 }
{ topic: 'wateranalytic',
  value: '{"orp": "477.3", "do": "2.95", "sal": "0.24", "ec": "515.3", "tds": "27
8", "tanggal": "2017-04-04", "lat": "-7.297746", "loc": "ngagel", "temp": "28.15",
"long": "112.741606", "label": "Lightly Polluted", "jam": "01:47:36", "ph": "5.
875", "sg": "1", "result": "2"}',
  offset: 403,
  partition: 0,
  key: -1 }
{ topic: 'wateranalytic',
  value: '{"orp": "420.8", "do": "3.71", "sal": "0.23", "ec": "495.7", "tds": "26
7", "tanggal": "2017-04-04", "lat": "-7.305796", "loc": "prapatanprapen", "temp":
"28.39", "long": "112.761288", "label": "Lightly Polluted", "jam": "01:47:37", "
ph": "6.71", "sg": "1", "result": "2"}',
  offset: 404,
```

Gambar 4.8. Data yang diterima oleh *backend* Node JS

Gambar 4.9 menunjukkan hasil visualisasi pada *front end*. Port yang digunakan adalah port standar http yaitu 80. Pada gambar 4.9 terlihat posisi node sensor yang berada di beberapa titik lokasi sepanjang sungai di Kota Surabaya.

Gambar 4.9. Visualisasi lokasi node sensor

Gambar 4.10 menunjukkan chart dari data sensor dan hasil klasifikasi *realtime* dari sistem.

Gambar 4.10. Visualisasi data sensor dan hasil klasifikasi

Gambar 4.11 menunjukkan informasi node dan aplikasi yang berjalan di atas Hadoop dengan *web user interface*. Halaman ini dapat diakses pada port 8088.

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
3	0	3	0	9	12 GB	12 GB	0 B	9	8	0	1	0	0	0	0

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1489693380176_0003	hduser	RealTimeAnalyticsBigData	SPARK	default	Thu, 16 Mar 2017 20:02:48 GMT	N/A	RUNNING	UNDEFINED	<div></div>	ApplicationMaster
application_1489693380176_0002	hduser	mqttToHive	SPARK	default	Thu, 16 Mar 2017 20:01:47 GMT	N/A	RUNNING	UNDEFINED	<div></div>	ApplicationMaster
application_1489693380176_0001	hduser	Zeppelin	SPARK	default	Thu, 16 Mar 2017 19:48:51 GMT	N/A	RUNNING	UNDEFINED	<div></div>	ApplicationMaster

Showing 1 to 3 of 3 entries

Gambar 4.11 Informasi *job* pada Hadoop cluster

f. Lapisan *Collaboration & Processes*

Gambar 4.12 menunjukkan proses interaksi antara analis dan sistem. Informasi yang ditunjukkan berupa data air yang tersimpan pada Hadoop HDFS yang dapat diakses untuk analisa lebih lanjut. Pengaksesan data yang tersimpan pada Hadoop HDFS dapat dilakukan secara langsung. Zeppelin pada penelitian ini diakses melalui port 3000. Gambar 4.12 dan 4.13 ini mewakili proses pada lapisan ke-7 dari platform IoT yang dibangun.

Zeppelin Notebook

Load data Hive

```

sqlContext = HiveContext(sc)
pc = "ryjadh"
portHive = 10000
authHive = "PLAIN"
userHive = "hduser"
passHive = "magfirran"
database = "default"

##### koneksi ke Hive
con = pyHive.connect(host=pc, port=portHive, authMechanism=authHive, user=userHive, password=passHive, database=database)
cur = con.cursor()

print cur.execute("select * from watermonitoringku limit 1000")
print cur.fetchall()
for i in cur.fetchall():
    print i

None
[('comment': None, 'columnName': 'watermonitoringku.tanggal', 'type': 'DATE_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.jam', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.org', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.ph', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.es', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.tss', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.sal', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.sg', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.do', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.temp', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.long', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.lat', 'type': 'STRING_TYPE'), ('comment': None, 'columnName': 'watermonitoringku.loc', 'type': 'STRING_TYPE')]
[('2017-01-30', '13:47:55', '297.6', '7.255', '519.3', '280', '0.25', '1', '3.15', '28.15', '-7.301354', '112.743381', 'pintuairjagir')]
[('2017-01-30', '13:47:55', '382.9', '5.804', '508.4', '280', '0.27', '1', '3.51', '28.64', '-7.305158', '112.800950', 'nonorejo')]
[('2017-01-30', '13:47:55', '467.1', '6.076', '514.7', '277', '0.28', '1', '3.4', '28.15', '-7.310150', '112.783255', 'stikom')]
[('2017-01-30', '13:47:55', '508.7', '7.43', '513.2', '277', '0.24', '1', '2.9', '28.39', '-7.297746', '112.741686', 'ngagel')]
[('2017-01-30', '13:47:55', '438.4', '6.7', '491.4', '269', '0.23', '1', '3.59', '28.39', '-7.305796', '112.761288', 'prapatanoropen')]
[('2017-01-30', '14:05:17', '390.1', '6.179', '508.1', '280', '0.27', '1', '2.51', '28.64', '-7.305158', '112.800950', 'nonorejo')]
[('2017-01-30', '14:05:17', '271.9', '7.675', '499.2', '269', '0.24', '1', '2.70', '28.15', '-7.299487', '112.734737', 'joyoyooy')]
[('2017-01-30', '14:05:17', '287.1', '6.924', '526.4', '284', '0.25', '1', '3.06', '28.15', '-7.301354', '112.743381', 'pintuairjagir')]
[('2017-01-30', '14:05:18', '381.2', '7.268', '515.9', '278', '0.24', '1', '3.67', '28.64', '-7.310150', '112.783255', 'stikom')]
[('2017-01-30', '14:05:18', '481.4', '5.888', '514.9', '278', '0.24', '1', '2.66', '28.39', '-7.297746', '112.741686', 'ngagel')]

```

Gambar 4.12. Akses data pada Hadoop HDFS menggunakan *web interface*

Gambar 4.13 menunjukkan proses training dataset secara online dan interaktif. Proses training data pada proses ini juga dapat menghasilkan visualisasi dari proses *training*. Pada gambar 4.13 menunjukkan kode program python yang dijalankan untuk melakukan proses *training* dataset. Gambar 4.13 ini sekaligus mewakili aplikasi yang dibangun untuk proses pada layer ke-6 (lapisan Application), sub bagian proses *learning*.

Gambar 4.13. Proses *training* dataset

4.6.2 Eksperimen pada algoritma klasifikasi

Pada subbab ini akan dijabarkan hasil eksperimen pembentukan model klasifikasi metode Indeks Pencemaran menggunakan algoritma Decision Tree dan Support Vector Machine. Hasil eksperimen tersebut dijabarkan dalam beberapa bagian yaitu: 1) *Confusion Matrix*, yang menampilkan data hasil *learning* yang berada pada kategori *True Positive* (TP), *False Positive* (FP), *True Negative* (TN) dan *False Negative* (FN); 2) Menentukan *Mean Squared Error* (MSE), *Mislabel*, dan *Score* dari data-data yang diperoleh dari *Confusion matrix*; 3) Grafik ROC yang menampilkan grafik perbandingan antara *True Positive Rate* dan *False Positive Rate*, yang bertujuan untuk memvalidasi model klasifikasi yang telah dibangun; dan 4) *Processing time*, yang menghitung waktu rata-rata yang

dibutuhkan server dalam mengolah, menganalisa, dan menampilkan data sensor air yang diterima dari node.

a. *Confusion Matrix*

Tabel 4.4 menunjukkan *confusion matrix* dari dataset uji laboratorium menggunakan Linear Support Vector Machine.

Tabel 4.4. *Confusion Matrix* dataset uji laboratorium menggunakan Linear SVM

		Predicted Class			
		0	1	2	3
Actual Class	0	0	0	0	0
	1	0	206	6	0
	2	0	19	159	0
	3	0	0	0	0

Tabel 4.5 menunjukkan *confusion matrix* dari dataset uji laboratorium menggunakan Decision Tree.

Tabel 4.5. *Confusion matrix* dataset uji laboratorium menggunakan Decision Tree

		Predicted Class			
		0	1	2	3
Actual Class	0	0	0	0	0
	1	0	212	0	0
	2	0	2	176	0
	3	0	0	0	0

Tabel 4.6 menunjukkan *confusion matrix* dari dataset sensor *live* menggunakan Linear Support Vector Machine.

Tabel 4.6. *Confusion matrix* dataset sensor *live* menggunakan linear SVM

		Predicted Class			
		0	1	2	3
Actual Class	0	0	0	0	0
	1	0	19770	107	0
	2	0	182	25338	0
	3	0	0	0	0

Tabel 4.7 menunjukkan *confusion matrix* dari dataset sensor *live* menggunakan Decision Tree.

Tabel 4.7. *Confusion matrix* dataset sensor *live* menggunakan Decision Tree

		Predicted Class			
		0	1	2	3
Actual Class	0	0	0	0	0
	1	0	19859	18	0
	2	0	16	25504	0
	3	0	0	0	0

b. MSE, *Mislabel*, dan *Score*

Dalam melakukan klasifikasi, sebuah sistem diharapkan dapat melakukan klasifikasi semua set data dengan benar. Sebenarnya, kinerja suatu sistem klasifikasi tidak bisa bekerja secara 100% benar, namun kinerja sistem klasifikasi dapat diukur. Dalam mengukur kinerja sistem klasifikasi dapat menggunakan *confusion matrix*.

Dari *confusion matrix*, kita dapat menghitung *mislabel*, *score* dan *mean squared error* (MSE) pada tiap algoritma pada tiap dataset. *Mislabel* adalah hasil jumlah data yang diprediksi secara salah terhadap jumlah dataset yang akan diprediksi setelah dilakukan training data. *Score* adalah nilai akurasi hasil prediksi untuk mengetahui jumlah data yang diklasifikasikan secara benar. Untuk menghitung akurasi digunakan persamaan 4.1.

$$Score = \frac{Jumlah\ data\ yang\ diprediksi\ benar}{Jumlah\ data\ prediksi\ total} \quad (4.1)$$

MSE adalah nilai *error rate* yang digunakan untuk mengetahui jumlah data yang diklasifikasikan secara salah sehingga mengetahui laju *error* pada prediksi yang dilakukan. Untuk menghitung nilai *error rate* digunakan persamaan 4.2.

$$MSE = \frac{Jumlah\ data\ yang\ diprediksi\ salah}{Jumlah\ data\ prediksi\ total} \quad (4.2)$$

Berdasarkan hasil eksperimen yang dilakukan terhadap semua dataset dengan perbandingan data *training* dan data *testing* sebesar 70%:30%, diperoleh tabel nilai *mislabel*, *score* dan MSE yang dapat dilihat pada tabel 4.8.

Tabel 4.8. Nilai *mislabel*, *score* dan MSE

Features	Dataset	Algorithm	Mislabel	Score	MSE
pH, TSS, DO, Temp, Turbidity	Laboratory test	Linear Support Vector Machine	24 / 390	0.938462	0.0615
		Decision Tree	2 / 390	0.994872	0.0051
	Live Sensor	Linear Support Vector Machine	286 / 45397	0.993700	0.0063
		Decision Tree	34 / 45397	0.999251	0.0007

Pada tabel 4.8 terlihat jika secara umum performa algoritma Decision Tree lebih baik dibandingkan dengan algoritma Support Vector Machine pada penelitian ini. Algoritma Decision Tree memiliki rata-rata tingkat akurasi sebesar 0.9970615 (99.7%) dan MSE rata-rata 0.0029. Support Vector Machine memiliki rata-rata tingkat akurasi sebesar 0.966081 (96.6%) dan MSE rata-rata 0.0339.

Perbandingan skor dari tiap dataset dan algoritma juga dapat dilihat pada gambar 4.14. Pada gambar 4.14 terlihat bahwa dataset sensor *live* memiliki akurasi yang lebih baik dibandingkan dengan dataset uji laboratorium. Selain itu dari kedua algoritma yang dibandingkan algoritma Decision Tree memiliki akurasi yang lebih baik.

Gambar 4.14. Perbandingan skor antar dataset dan algoritma

c. ROC

Kurva ROC merupakan salah satu cara melakukan analisa terhadap model klasifikasi yang telah dibuat dalam menentukan parameter model yang diinginkan sesuai dengan karakteristik dari model klasifikasi yang diinginkan. Penggunaan kurva ROC seringkali digunakan dalam mengevaluasi proses klasifikasi, dikarenakan mempunyai kemampuan evaluasi secara menyeluruh dan cukup baik. Misalkan pada table yang terdiri dari dua buah kelas data yaitu data kelas yang dihasilkan dari *classifier* (*Predicted Class*) dan data kelas asli yang telah diketahui (*Actual Class*) dimana data *Predicted Class* yang sama dengan *Actual Class* maka termasuk *True Positive* (TP), sedangkan data *Predicted Class* yang tidak sama dengan *Actual Class* tapi termasuk dari data hasil klasifikasi maka termasuk *False Positive* (FP).

Kurva ROC digunakan sebagai grafik perbandingan antara *True Positive Rate* (TPR) pada sumbu vertikal dengan *False Positive Rate* (FPR) pada sumbu horisontal. TPR merupakan proporsi data yang digunakan positif yang teridentifikasi dengan benar antara data *Predicted Class* dengan *Actual Class* sedangkan FPR merupakan proporsi data negatif yang teridentifikasi salah sebagai positif pada suatu model klasifikasi. *True Positive Rate* dan *False Positive Rate*, dapat dihitung menggunakan persamaan:

$$TPR = \frac{TP}{TP+FP} \text{ dan}$$

$$FPR = \frac{FP}{TP+FP} \quad (4.3)$$

Pada kurva ROC, luas area dibawah kurva dikenal dengan AUC (*Area Under the ROC Curve*), dimana nilai AUC berkisaran antara 0 sampai dengan 1, semakin mendekati 1 maka semakin baik nilai uji pada karakteristik klasifikasi tersebut. Nilai kategori AUC [26], sebagai berikut:

Tabel 4.9. Tabel nilai AUC

Range Nilai AUC	Keterangan
0.5 – 0.6	Fail
0.6 – 0.7	Poor
0.7 – 0.8	Fair
0.8 – 0.9	Good
0.9 – 1.0	Excellent

Gambar 4.15 menunjukkan grafik ROC dari dataset uji laboratorium menggunakan Linear SVM.

Gambar 4.15. ROC dataset uji laboratorium menggunakan Linear SVM

Gambar 4.16 menunjukkan grafik ROC dari dataset uji laboratorium menggunakan Decision Tree.

Gambar 4.16. ROC dataset uji laboratorium menggunakan Decision Tree

Gambar 4.17 menunjukkan grafik ROC pada dataset sensor *live* menggunakan Linear SVM.

Gambar 4.17. ROC dataset sensor *live* menggunakan linear SVM

Gambar 4.18 menunjukkan grafik ROC pada dataset sensor *live* menggunakan Decision Tree.

Gambar 4.18. ROC dataset sensor *live* menggunakan Decision Tree

4.6.3 Processing Time

Dalam menentukan *processing time* dihitung sejak data tersebut diterima oleh server hingga data tersebut divisualisasi pada *front end*. Selain itu dalam perhitungan *processing time* ini digunakan skenario jumlah node yang mengirim data secara bersamaan yang berbeda-beda. Gambar 4.18 menunjukkan rata-rata waktu pemrosesan data pada server.

Gambar 4.19. Processing time dalam mS vs jumlah node sensor

4.7 ANALISIS HASIL EKSPERIMEN

Dari *confusion matrix* yang dihasilkan pada percobaan algoritma klasifikasi diperoleh hasil yang memperlihatkan tidak adanya sampel data yang diperoleh baik dari uji laboratorium PDAM maupun dari sensor live PDAM yang memenuhi standar mutu baik air minum. Di samping juga tidak ditemukan sampel data yang memenuhi standar tersemar berat. Hal ini menunjukkan bahwa sungai-sungai di sepanjang sungai di Kota Surabaya berada pada tingkat tercemar ringan dan tercemar sedang. Hal ini merupakan informasi awal yang dapat dijadikan acuan untuk meningkatkan kualitas air sungai di Kota Surabaya atau malah membiarkan hingga air sungai di Kota Surabaya berada pada level tercemar berat.

Hasil learning antara algoritma Linear SVM dan Decision Tree menunjukkan performa yang baik dimana tingkat akurasi berada pada level $> 90\%$ dan MSE sekitar 0.019075. Hal ini menunjukkan dataset yang digunakan tidak mengalami fluktuasi angka yang signifikan. Namun jika kedua algoritma dibandingkan maka algoritma Decision Tree menghasilkan akurasi yang lebih baik yaitu dengan score 0.999251 untuk dataset sensor *live* dan 0.994872 untuk dataset uji laboratorium. Salah satu penyebab performa Linear SVM yang kurang baik dibanding dengan Decision Tree adalah rapatnya jarak antar data sensor yang ada, hal ini menyebabkan algoritma Linear SVM kesulitan dalam menemukan garis *hyperplane* yang paling optimal.

Dari hasil grafik ROC menunjukkan bahwa kinerja sistem klasifikasi yang dibangun memiliki performa yang 'excellent'. Dimana status 'excellent' tersebut diperoleh dari nilai grafik ROC yang diperoleh > 0.9 dan bahkan mendekati atau sama dengan 1.

Spark merupakan aplikasi machine learning pada Big Data yang menggunakan skema pemrosesan dalam memori. Hal ini mengakibatkan kebutuhan penggunaan memori yang besar oleh Spark. Dalam menjalankan aplikasi Spark pada Big Data platform, terkadang ditemukan kendala *error* yaitu *Out of Memory Error*. Hal ini dapat diatasi dengan meningkatkan alokasi Java Heap pada sistem. Dalam percobaan ini, kami menggunakan Yarn untuk

manajemen cpu dan memori pada *job* Spark. Pada yarn, kami alokasikan 4 GB memori yang dialokasikan pada tiap *job* Spark yang berjalan.

Server rata-rata hanya membutuhkan waktu 508 miliseconds untuk memproses data yang berasal dari semua node. Waktu pemrosesan tersebut adalah waktu di saat data diterima oleh server dan berakhir saat data telah divisualisasi. Hal ini bisa terjadi karena skema *direct flow* yang digunakan dalam sistem ini. Hal ini didukung oleh kemampuan Spark dalam memproses data dengan skema pemrosesan dalam memori dan kemampuan Node JS yang dapat mendukung pemrosesan data *realtime* hingga divisualisasi. *Processing time* di sini tidak termasuk waktu transmisi data dari node sensor ke server. Dari percobaan sebelumnya [36], diperoleh hasil jika rata-rata waktu pengiriman data dari node ke server adalah 1 detik. Hal ini menunjukkan bahwa sistem yang dibangun efektif.

4.8 DISKUSI

Penelitian ini berangkat dari sebuah ide untuk menemukan sebuah metode untuk menyelesaikan sebuah persoalan yaitu mendeteksi keberadaan kontaminan tertentu (berbahaya) dengan menggunakan sensor konvensional.

Hal ini merupakan ide awal dari penelitian yang dilaksanakan. Yakni dengan hanya menggunakan sensor yang saat ini digunakan (sensor fisika) dapat mendeteksi kontaminan tertentu (berbahaya/kimia) yang kemungkinan terkandung dalam air. Dari ide tersebut rencananya dikembangkan sebuah metode model klasifikasi yang dapat digunakan untuk mendeteksi keberadaan kontaminan tersebut. Sejauh ini masalah utama yang ditemukan penulis adalah ketidaksiapan dataset yang dapat mendukung proses pembelajaran pada *machine learning*.

Dataset yang diperoleh dari PDAM Surya Sembada Surabaya, Kementerian Lingkungan Hidup Kota Surabaya, maupun dari berbagai sumber di internet, salah satunya adalah *An inorganic water chemistry dataset (1972-2011) of rivers, dams and lakes in South Africa* [50], tidak dapat menemukan keunikan reaksi sensor pada kontaminan tertentu. Hal ini disebabkan dataset yang diperoleh baik dari PDAM, Kementerian Lingkungan Hidup, maupun beberapa sumber di internet merupakan data hasil monitoring air sungai/danau yang kondisi airnya cenderung

stabil, kondisi yang sangat jarang terjadi adanya kontaminan tertentu dalam level berbahaya, hingga tidak tampaknya perubahan signifikan pada hasil karena adanya kontaminan tertentu.

Ke depannya, penelitian ini dapat dilanjutkan dengan menggali lebih dalam sumber-sumber dataset yang dapat menyelesaikan persoalan di atas atau melakukan kolaborasi dengan pakar di bidang teknik/kimia lingkungan yang dapat membantu melakukan pembuatan dataset secara langsung/manual dengan melibatkan beberapa kontaminan berbahaya yang dilarutkan ke dalam air dengan memperhatikan reaksi dari sensor fisika yang digunakan hingga dapat ditentukan pola pembacaan sensor fisika akan keberadaan kontaminan kimia.

BAB 5

PENUTUP

5.1 KESIMPULAN

Pada penelitian ini bertujuan untuk membangun sistem klasifikasi *realtime* air sungai dengan menggunakan teknologi IoT dan *Big Data*. Dimana dalam pengeimplementasiannya dikembangkan sebuah platform IoT yang terdiri dari 7 lapisan IoT dan diintegrasikan dengan teknologi *Big Data* pada beberapa lapisan tersebut. Penggunaan teknologi *Big Data* utamanya berada pada sisi penyimpanan, analisa dengan klasifikasi, dan manajemen data dalam server.

Hasil klasifikasi dengan menggunakan algoritma Linear SVM dan Decision Tree menunjukkan performa yang baik, dimana akurasi berada pada level di atas 90%. Pada dataset uji laboratorium algoritma Linear SVM menunjukkan akurasi sebesar 0.935897 sedangkan Decision Tree 0.994872. Sementara untuk dataset sensor live algoritma Linear SVM menunjukkan akurasi sebesar 0.993634 sedangkan Decision Tree 0.999251. Hal ini dapat disimpulkan bahwa algoritma Decision Tree memiliki akurasi yang lebih baik dibandingkan dengan algoritma Linear SVM. Dimana algoritma Decision Tree memiliki rata-rata tingkat akurasi sebesar 0.9970615 dan algoritma Linear SVM sebesar 0.9647655.

Pengujian hasil validasi yang telah dilakukan pada dataset uji laboratorium maupun dataset sensor live berdasarkan grafik ROC dengan nilai Area Under ROC menunjukkan di atas angka 0.9. Dengan demikian dapat dikatakan bahwa unjuk kerja nilai Area Under ROC menunjukkan kinerja 'Excellent'.

Sistem yang dibangun hanya membutuhkan rata-rata 508 miliseconds dalam memproses data oleh server yang diterima dari node sensor. Hal ini menunjukkan sistem platform IoT-Big Data yang dibangun memiliki kinerja yang sangat baik.

5.2 SARAN

Berdasarkan hasil penelitian tesis desain dan implementasi IoT-Big Data analytic pada Smart Environment Monitoring system maka dapat disarankan:

- a. Pengembangan sistem monitoring lingkungan yang tidak hanya dapat melakukan analisa teks, tapi juga dapat melakukan analisa terhadap gambar secara *realtime*.
- b. Pengembangan *online classification* pada sistem SEMAR.
- c. Pengembangan penelitian yang ditujukan untuk peruntukan monitoring pada lingkungan lainnya, seperti udara.
- d. Pengembangan sistem analisa untuk kebutuhan *clustering* ataupun *regression* pada data lingkungan.

DAFTAR PUSTAKA

1. Pratama, Fajar Gilang. **Potensi Pengembangan Perikanan Darat dan Laut Indonesia**. Institut Teknologi Bandung, 2008.
2. Hapsari, Ratih Indri, and Mohammad Zenurianto. **View of flood disaster management in Indonesia and the key solutions**. *American Journal of Engineering Research* 5, no. 3 (2016): 140-151.
3. Zulkarnain, Muhammad Rizki, Suwito, Tasripan. **Sistem Monitoring Kualitas Air Sungai yang Dilengkapi dengan Data Logger dan Komunikasi Wireless sebagai Media Pengawasan Pencemaran Limbah Cair**. Institut Teknologi Sepuluh Nopember Surabaya, 2014.
4. Republik Indonesia, **Undang-Undang Republik Indonesia nomor 7 tahun 2004 tentang Sumber Daya Air**, 2004.
5. Republik Indonesia, **Peraturan Pemerintah nomor 82 tahun 2001 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air**, 2001.
6. Provinsi Jawa Timur, **Peraturan Daerah Provinsi Jawa Timur nomor 2 tahun 2008 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air Di Provinsi Jawa Timur**, 2008.
7. Kota Surabaya, **Peraturan Daerah Kota Surabaya nomor 2 tahun 2004 tentang Pengelolaan Kualitas Air dan Pengendalian Pencemaran Air**, 2004.
8. Menteri Kesehatan, **Peraturan Menteri Kesehatan nomor 492/MENKES/PER/IV/2010 tentang Persyaratan Kualitas Air Minum**, 2010.
9. Nemerow, Nelson Leonard. **Scientific Stream Pollution Analysis**. McGraw-Hill, 1974.
10. Menteri Negara Lingkungan Hidup, **Keputusan Menteri Negara Lingkungan Hidup Nomer 115 tahun 2003 tentang Pedoman Penentuan Status Mutu Air**, 2003.
11. Recommendation ITU-T Y.2060, **Overview of Internet of Thing**, 2012.

12. Cisco, **The Internet of Things Reference Model**. White Paper, 2014.
13. Nave, C. R, **pH as a Measure of Acid and Base Properties**. <http://hyperphysics.phy-astr.gsu.edu/hbase/Chemical/ph.html>, diakses tanggal 8 maret 2016.
14. Miller, R. L., Bradford, W. L., & Peters, N. E. **Specific Conductance: Theoretical Considerations and Application to Analytical Quality Control**. <http://pubs.usgs.gov/wsp/2311/report.pdf>, diakses tanggal 6 mei 2016.
15. Kemker, Christine, **Dissolved Oxygen**, Fundamental of Environmental Measurements, <http://www.fondriest.com/environmental-measurements/parameters/water-quality/dissolved-oxygen/>, diakses pada tanggal, 28 mei 2016.
16. Jskinner, **Oxygen (Dissolved Oxygen, DO) In Water Chemistry**, http://academic.keystone.edu/jskinner/Limnology/Water_Chemistry_Lecture_Notes.html, diakses pada tanggal 28 desember 2015.
17. Wetzel, R. G. (2001). *Limnology: Lake and River Ecosystems* (3rd ed.). San Diego, CA: Academic Press.
18. Kemker, Christine. **Water Temperature**, Fundamentals of Environmental Measurements. <http://www.fondriest.com/environmental-measurements/parameters/water-quality/water-temperature/>, diakses pada tanggal 28 mei 2016.
19. Kemker, Christine. **Conductivity, Salinity and Total Dissolved Solid**, Fundamentals of Environmental Measurements. <http://www.fondriest.com/environmental-measurements/parameters/water-quality/conductivity-salinity-tds/>, diakses pada tanggal 28 januari 2016
20. Raspberry, **Raspberry – Teach Learn and Make with Rapberry Pi**, <http://www.raspberrypi.org>, diakses 8 Maret 2017.
21. Alcnct, **USB Modem 4G DT-100**, <http://alnect.net/product/7633/Page-Modem-GSM-4GLTE-Advance-Jetz-DT100-Plus-Soft-AP>, diakses 8 Maret 2017.
22. MQTT, **Message Queue Telemetry Transport**, <http://mqtt.org>, diakses tanggal 8 maret 2017.

23. Light, R., **Mosquitto - An Open Source MQTT v3.1/v3.1.1 Broker**, <http://mosquitto.org>, diakses tanggal 8 maret 2017.
24. MQTT, **The MQTT Protocol Concept**, <http://mosquitto.org/man/mqtt-7.html>, diakses tanggal 30 Nopember 2016.
25. Chen, Min, Shiwen Mao, and Yunhao Liu. **Big Data: a survey**, *Mobile Networks and Applications* 19, no. 2, 2014: 171- 209.
26. Hadoop, **Apache Hadoop**, <http://hadoop.apache.org>, diakses tanggal 6 mei 2016.
27. Hive, **Apache Hive**, <http://hive.apache.org>, diakses tanggal 6 mei 2016.
28. Kafka, **Apache Kafka: A Distributed Streaming Platform**, <http://kafka.apache.org>, diakses tanggal 6 mei 2016.
29. Bishop, C. M. **Pattern Recognition and Machine Learning**, 2001.
30. Prasetyo, Eko, **Data Mining-Mengolah Data Menjadi Informasi Menggunakan Matlab**, Andi Offset, Ed.1, pp. 28-30, 2014.
31. Anyanwu, Matthew N., and Sajjan G. Shiva. **Comparative analysis of serial decision tree classification algorithms**. *International Journal of Computer Science and Security* 3, no. 3 (2009): 230-240.
32. Breiman, Leo, Jerome Friedman, Charles J. Stone, and Richard A. Olshen, **Classification and Regression Trees**. CRC press, 1984.
33. Bramer, Max. **Principles of Data Mining**, Vol. 180, Springer, 2007.
34. I. Steinwart and A. Christmann, **Support Vector Machine**, Information Science and Statistics, Springer, 2008.
35. Spark, **Apache Spark**, <http://spark.apache.org>, diakses tanggal 8 maret 2016.
36. Zeppelin, **Apache Zeppelin**, <http://zeppelin.apache.org>, diakses tanggal 28 mei 2016.
37. Sukaridhoto, Sri trusta, Dadet Pramadihanto, Muhammad Alif, Andrie Yuwono, and Nobuo Funabiki. **A design of radio-controlled submarine modification for river water quality monitoring**, *In Intelligent Technology and Its Applications (ISITIA), 2015 International Seminar on*, pp. 75-80. IEEE, 2015.
38. Yuliandoko, Herman, Sri trusta Sukaridhoto, M. Udin Harun Al Rasyid, and Nobuo Funabiki. **Performance of Implementation IBR-DTN and Batman-**

- Adv Routing Protocol in Wireless Mesh Networks**, *EMITTER International Journal of Engineering Technology* 3, no. 1 (2016).
39. Sukaridhoto, Sritrusta, Rahardhita Widyatra Sudibyo, Widi Sarinastiti, Rizky Dharmawan, Atit Sasono, Ahmad Andika Saputra, and Shiori Sasaki. **Design and development of a portable low-cost COTS-based water quality monitoring system**, In *Intelligent Technology and Its Applications (ISITIA), 2016 International Seminar on*, pp. 635-640. IEEE, 2016.
 40. Abdillah, Abid, Muhammad Herwindra, Yohanes Panduman, Muhammad Akbar, Marlanisa Afifah, Sritrusta Sukaridhoto, Shiori Sasaki. **Design and Development Low Cost Coral Monitoring System for Shallow Water Based on Internet of Underwater Things**, In *Advanced Researh in Electronic Engineering and Information Technology International Conference (AVAREIT)*, 2016.
 41. Berlian, Muhammad Herwindra, Tegar Esa Rindang Sahputra, Buyung Jofi Wahana Ardi, Luhung Wahya Dzatmika, Adnan Rachmat Anom Besari, Rahardhita Widyatra Sudibyo, and Sritrusta Sukaridhoto. **Design and implementation of smart environment monitoring and analytics in real-time system framework based on internet of underwater things and big data.** In *Electronics Symposium (IES), 2016 International*, pp. 403-408. IEEE, 2016.
 42. Modaresi, Fereshteh, and Shahab Araghinejad. **A comparative assessment of support vector machines, probabilistic neural networks, and K-nearest neighbor algorithms for water quality classification**, *Water resources management* 28, no. 12 (2014): 4095-4111.
 43. Ladjal, Mohamed, Mohamed Bouamar, Mohamed Djerioui, and Youcef Brik. **Performance evaluation of ANN and SVM multiclass models for intelligent water quality classification using Dempster-Shafer Theory**, In *Electrical and Information Technologies (ICEIT), 2016 International Conference on*, pp. 191-196. IEEE, 2016.
 44. Jaloree, Shailesh, Anil Rajput, and Sanjeev Gour. **Decision tree approach to build a model for water quality**, *Binary Journal of Data Mining & Networking* 4, no. 1 (2014): 25-28

45. Sagheblian, S. Mehdi, M. Taghi Sattari, Rasoul Mirabbasi, and Mahesh Pal. **Ground water quality classification by decision tree method in Ardebil region, Iran**, *Arabian Journal of Geosciences* 7, no. 11 (2014): 4767-4777.
46. Fazio, Maria, Antonio Celesti, Antonio Puliafito, and Massimo Villari. **Big data storage in the cloud for smart environment monitoring**, *Procedia Computer Science* 52 (2015): 500-506.
47. Meng, Xiangrui, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman et al., **Mllib: Machine learning in apache spark**, *Journal of Machine Learning Research* 17, no. 34 (2016): 1-7.
48. Richter, Aaron N., Taghi M. Khoshgoftaar, Sara Landset, and Tawfiq Hasanin., **A multi-dimensional comparison of toolkits for machine learning with big data**, In *Information Reuse and Integration (IRI), 2015 IEEE International Conference on*, pp. 1-8. IEEE, 2015.
49. Kalaria, Dhruv, **MQTTKafkaBridge: Bridge MQTT Messages from Mosquitto to Kafka Broker on the same topic**, <http://github.com/DhruvKalaria/MQTTKafkaBridge>, diakses 28 April 2017
50. Huizenga, Jan Marten, Michael Silberbauer, Rainier Dennis, and Ingrid Dennis. **An inorganic water chemistry dataset (1972–2011) of rivers, dams and lakes in South Africa**. *Water SA* 39, no. 2 (2013): 335-340.

LAMPIRAN

Surat rekomendasi penelitian dari Bakesbangpol Linmas Surabaya kepada
BLH Kota Surabaya dan PDAM Kota Surabaya

PEMERINTAH KOTA SURABAYA
BADAN KESATUAN BANGSA, POLITIK DAN PERLINDUNGAN MASYARAKAT
Jl. Tambaksari No. 11 Telp. (031) 99443016 - 99443066
SURABAYA (60136)

Surabaya, 31 Agustus 2016

Kepada

Nomor : 070 / 8667 / 436.7.3 / 2016
Lampiran : -
Hal : Penelitian

Yth. 1. Kepala Badan Lingkungan Hidup Kota Surabaya
2. Direktur PDAM Kota Surabaya

di - SURABAYA

REKOMENDASI PENELITIAN

Dasar : 1. Peraturan Menteri Dalam Negeri Nomor 64 Tahun 2011 tentang Pedoman Penerbitan Rekomendasi Penelitian, sebagaimana telah diubah dengan Peraturan Menteri Dalam Negeri Nomor 7 Tahun 2014 tentang Perubahan Atas Peraturan Menteri Dalam Negeri Nomor 64 Tahun 2011 ;
2. Peraturan Walikota Surabaya Nomor 37 Tahun 2011 Tentang Rincian Tugas dan Fungsi Lembaga Teknis Daerah Kota Surabaya, Bagian Kedua Badan Kesatuan Bangsa, Politik dan Perlindungan Masyarakat.

Memperhatikan : Surat Ketua Program Pascasarjana Politeknik Elektronika Negeri Surabaya tanggal 11 Agustus 2016 Nomor : 2752/PL14/AK/2016 hal Surat Pengantar Izin Penelitian

Kepala Badan Kesatuan Bangsa, Politik dan Perlindungan Masyarakat Kota Surabaya memberikan rekomendasi kepada :

a. Nama : Riyadh Arridha
b. Alamat : Lapasu RT 007 RW 001 Ds. Balusu Barru Sulawesi Selatan
c. Pekerjaan/Jabatan : Pegawai Negeri Sipil
d. Instansi/Organisasi : Politeknik Elektronika Negeri Surabaya
e. Kewarganegaraan : Indonesia

Untuk melakukan penelitian/survey/kegiatan dengan :

a. Judul / Thema : Desain dan Implementasi Big Data Analytic Pada Smart Environment Monitoring System
b. Tujuan : Penelitian
c. Bidang Penelitian : Teknik Informatika & Komputer
d. Penanggung Jawab: Sritrusta Sukaridhoto, S.T., Ph.D.
e. Anggota Peserta : -
f. Waktu : 3 (Tiga) Bulan, TMT Surat dikeluarkan
g. Lokasi : Badan Lingkungan Hidup, PDAM

Dengan persyaratan : 1. Penelitian/survey/kegiatan yang dilakukan harus sesuai dengan surat permohonan dan wajib mentaati persyaratan / peraturan yang berlaku di Lokasi / Tempat dilakukan Penelitian/survey/kegiatan ;
2. Saudara yang bersangkutan agar setelah melakukan Penelitian/survey/kegiatan wajib melaporkan pelaksanaan dan hasilnya kepada Kepala Bakesbang, Politik dan Linmas Kota Surabaya ;
3. Penelitian/survey/kegiatan yang dilaksanakan tidak boleh menimbulkan keresahan dimasyarakat, disintegrasi bangsa atau mengganggu keutuhan NKRI ;
4. Rekomendasi ini akan dicabut / tidak berlaku apabila yang bersangkutan tidak memenuhi persyaratan seperti tersebut diatas.

Demikian atas bantuannya disampaikan terima kasih.

a.n. KEPALA BADAN
Kesatuan Bangsa, Politik dan Perlindungan Masyarakat,
Drs. Deddy Rosalia, M.Si
Surabaya
NIP. 19621212 198903 1 029

Tembusan :
Yth. 1. Ketua Program Pascasarjana Politeknik Elektronika Negeri Surabaya
2. Saudara yang bersangkutan

PERUSAHAAN DAERAH AIR MINUM SURYA SEMBADA KOTA SURABAYA	
Nomor : 072/ 1100 /436.9.1/2016 Sifat : BIASA Lampiran : - Hal : <u>Penelitian</u>		Surabaya, 29 SEP 2016 Kepada : Yth Kepala BAKESBANGPOL Jl. Tambak Sari No 11 di - <u>SURABAYA</u>	
<p>Sehubungan dengan Surat Saudara Nomor : 070 / 8667 / 436.7.3 / 2016 tanggal 31 Agustus 2016 perihal sebagaimana tersebut dalam pokok surat, dengan ini diberitahukan bahwa Perusahaan Daerah Air Minum Surya Sembada Kota Surabaya dapat menyetujui permohonan Saudara dan pelaksanaannya dijadwalkan pada :</p>			
Tanggal	: 3 Oktober - 4 November 2016		
Waktu	: 07.30 WIB - selesai		
Jumlah Peserta	: 1 (Satu) orang		
Tempat	: Pengendalian Proses;		
Catatan	: Hasil Penelitian agar diserahkan kepada Sekretariat & Humas setelah selesai penelitian dan tidak mempublikasikan tanpa seijin / sepengetahuan PDAM Surya Sembada Kota Surabaya.		
Demikian atas perhatiannya disampaikan terima kasih.			
a.n. Direksi Perusahaan Daerah Air Minum Surya Sembada Kota Surabaya Sekretaris Perusahaan, SAYID M. IQBAL, S.Kom NIP. 1.05.01321			
<u>Tembusan:</u> Yth. 1. Pjs. Direktur Utama (sebagai laporan); 2. Manajer Pengendalian Proses; 3. Manajer Sekretariat dan Humas; <u>PDAM Surya Sembada Kota Surabaya.</u> 4. Riyadh Arridha; <u>Politeknik Elektronika Negeri Surabaya</u>			
Kantor : Jl. Mayjen Prof. Dr. Moestopo No. 2, Telp. 031-5039373, 5039676, Fax 031-5030100, Surabaya 60131 Website : www.pdam-sby.go.id Call Center : 292-6666 Layanan SMS : 081.2331-6666			

BIODATA PENULIS

Riyadh Arridha, menempuh pendidikan S1 pada jurusan Teknik Informatika, Universitas Islam Negeri (UIN) Alauddin Makassar. Pendidikan S1 diselesaikan pada tahun 2011 dengan gelar S.Kom. Saat ini melanjutkan pendidikan Magister di Politeknik Elektronika Negeri Surabaya (PENS), Program Pascasarjana Teknologi Rekayasa, Program Studi Magister Terapan Teknik Informatika dan Komputer. Sejak tahun 2013 penulis bekerja sebagai dosen tetap di Politeknik Negeri Fakfak, Propinsi Papua Barat. Penulis menempuh pendidikan Magister pada tahun 2015 sebagai penerima Beasiswa BPP-DN dari KEMENRISTEKDIKTI yang pada tahun 2016 dialihkan pengelolaannya ke LPDP KEMENKEU. Bidang penelitian penulis antara lain *Internet of Things*, *Big Data*, *Embedded System* dan *Machine Learning*.

Riwayat Pendidikan:

1. SD Negeri Lapasu, Barru, Sulawesi Selatan
2. MTs. DDI Ambo Dalle Mangkoso, Barru, Sulawesi Selatan
3. SMA Negeri 1 Soppeng Riaja, Barru, Sulawesi Selatan
4. S1 Teknik Informatika Universitas Islam Negeri (UIN) Alauddin Makassar, Sulawesi Selatan

Publikasi:

Classification Extension based on IoT-Big Data Analytic for Smart Environment Monitoring and Analytic in Real-time System, International Journal of Space-Based and Situated Computing, 2017