ALGORITMA *DATA MINING* UNTUK OPTIMASI SUHU DAN WAKTU ROASTING NIBS BIJI KAKAO DI *COCOA TEACHING INDUSTRY* (CTI) UGM

SKRIPSI

Disusun oleh:

RIZKY ALIF RAMADHAN

19/446785/TK/49890

PROGRAM STUDI TEKNOLOGI INFORMASI

DEPARTEMEN TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI

FAKULTAS TEKNIK UNIVERSITAS GADJAH MADA

YOGYAKARTA

2023

HALAMAN PENGESAHAN

ALGORITMA DATA MINING UNTUK OPTIMASI SUHU DAN WAKTU ROASTING NIBS BIJI KAKAO DI COCOA TEACHING INDUSTRY (CTI) UGM

SKRIPSI

Diajukan Sebagai Salah Satu Syarat untuk Memperoleh
Gelar Sarjana Teknik
pada Departemen Pilih Fakultas Teknik
Universitas Gadjah Mada

Disusun oleh:
RIZKY ALIF RAMADHAN
19/446785/TK/49890

Telah disetujui dan disahkan
pada tanggal 15 Mei 2023

Dosen Pembimbing I	Dosen Pembimbing II
Enas Dhuhri Kusuma, S.T., M.Eng.	**Noor Akhmad Setiawan, Ir., S.T., M.T., Ph.D., IPM.**
NIP 198211082010121000	**NIP 197506071999031002**

PERNYATAAN BEBAS PLAGIASI

Saya yang bertanda tangan di bawah ini:

Nama	: Rizky Alif Ramadhan
NIM	: 19/446785/TK/49890
Tahun terdaftar	: 2019
Program Studi	: Teknologi Informasi
Fakultas	: Teknik Universitas Gadjah Mada

Menyatakan bahwa dalam dokumen ilmiah Skripsi ini tidak terdapat bagian dari karya ilmiah lain yang telah diajukan untuk memperoleh gelar akademik di suatu lembaga Pendidikan Tinggi, dan juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang/lembaga lain, kecuali yang secara tertulis disitasi dalam dokumen ini dan disebutkan sumbernya secara lengkap dalam daftar pustaka.

Dengan demikian saya menyatakan bahwa dokumen ilmiah ini bebas dari unsur-unsur plagiasi dan apabila dokumen ilmiah Skripsi ini di kemudian hari terbukti merupakan plagiasi dari hasil karya penulis lain dan/atau dengan sengaja mengajukan karya atau pendapat yang merupakan hasil karya penulis lain, maka penulis bersedia menerima sanksi akademik dan/atau sanksi hukum yang berlaku.

Yogyakarta, 05 Mei 2023

Materai Rp 10.000
(Tanda tangan)

Rizky Alif Ramadhan
NIM 19/446785/TK/49890

HALAMAN PERSEMBAHAN

Tugas akhir ini kupersembahkan kepada kedua orang tuaku. Kupersembahkan pula kepada keluarga dan teman-teman semua, Serta untuk bangsa, negara, dan agamaku.

KATA PENGANTAR

Puji syukur kehadirat Allah سبحانه وتعالى atas limpahan rahmat, karunia, serta petunjuk-Nya sehingga tugas akhir berupa penyusunan skripsi berjudul **"**Algoritma *Data Mining* untuk Optimasi Suhu dan Waktu Roasting Nibs Biji Kakao di *Cocoa Teaching Industry* (CTI) UGM" ini telah terselesaikan dengan baik. Sholawat serta salam tak lupa pula dihanturkan kepada Nabi Muhammad Shollallahu ﷺ, suri tauladan kita bersama. Dalam penyusunan tugas akhir ini penulis telah banyak mendapatkan arahan, bantuan, serta dukungan dari berbagai pihak. Oleh karena itu pada kesempatan ini penulis mengucapan terima kasih kepada:

1. Ir. Hanung Adi Nugroho, S.T., M.E., Ph.D., IPM.
2. Ir. Lesnanto Multa Putranto, S.T., M.Eng., Ph.D., IPM.
3. Bapak Enas Dhuhri Kusuma, S.T., M.Eng. dan Bapak Noor Akhmad Setiawan, Ir., S.T., M.T., Ph.D., IPM. Yang sudah memfasilitasi jalannya skripsi dan senantiasa sabar dalam membimbing dalam penyelesaian skripsi ini.
4. Pihak UGM *Cocoa Teaching Industry* (CTI) yang telah memberikan wadah untuk berkreasi.
5. Kedua Orang Tua, kakak dan adik yang selalu memberikan arahan selama belajar dan menyelesaikan tugas akhir ini.
6. Fifi Cintia Mahajasa Faizi yang sudah menjadi teman bercerita serta senantiasa mendukung dan men-support dalam lancarnya pengerjaan skripsi ini.
7. Teman-teman DTETI angaktan 2019 khususnya bagi teman-teman yang pernah satu tim dalam mengikuti lomba. Sungguh, pengalaman dan pengetahuan lomba khususnya dibidang data menjadi sebab lancarnya pengerjaan skripsi ini.

Akhir kata penulis berharap semoga skripsi ini dapat memberikan manfaat bagi kita semua, Aamiin

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR SINGKATAN

ELM	*= Extreme Learning Machine*
MAPE	*= Mean Absolute Percentage Error*
MLR	*= Multiple Linear Regression*
PSO	*= Particle Swarm Optimization*
RMSE	*= Root Mean Squared Error*
SVR	*= Support Vector Regression*

INTISARI

Algoritma *data mining* mulai digunakan pada berbagai bidang termasuk industri makanan. Dengan demikian, algoritma *data mining* ini tentunya juga dapat diterapkan pada industri kakao, terutama dalam proses pengolahan biji kakao. Salah satu proses dalam pengolahan biji kakao yang paling menentukan kualitas dari pengolahan biji kakao tersebut adalah proses *roasting.* Proses *roasting* adalah proses mengeluarkan kandungan air pada biji kakao, mengeringkannya, serta mengembangkan biji tersebut agar mendapatkan aroma dan warna yang khas dan sesuai dengan standar. Agar hasil *roasting* bagus, maka suhu dan durasi pemanasan harus optimal. Suhu dan durasi merupakan variabel yang memiliki peran penting dalam proses *roasting*. Tujuan dari penelitian ini membuat algoritma *data mining* yang dapat dijadikan rekomendasi suhu pada durasi *roasting* tertentu berdasarkan data-data pada proses *roasting* seperti kapasitas *roasting*, kadar air, pH, jenis biji dan lain sebagainya. Suhu akan dijadikan target variabel pada penelitian ini. Penulis membandingkan tiga algoritma *data mining*, yaitu *Support Vector Regression (SVR)*, *Multiple Linear Regression*, *Extreme Learning Machine (ELM)*, dan *Particle Swarm Optimization-Extreme Learning Machine (PSO-ELM)* pada penelitian ini. Algoritma *data mining* tersebut dievaluasi menggunakan ukuran standar regresi seperti MAPE dan RMSE untuk nantinya dibandingkan performanya. Setelah melakukan validasi silang, didapatkan algoritma *data mining* terbaik untuk memodelkan suhu adalah PSO-ELM dengan MAPE dan RMSE masing-masing 4.13% dan 7.36, kemudian SVR dengan MAPE 4.76% dan RMSE 9.17, selanjutnya ada ELM denngan MAPE 4.79% dan RMSE 8.62, dan yang terakhir ada MLR dengan MAPE 4.80% dan RMSE 8.63. Diharapkan dengan penelitian ini, operator dapat menentukan suhu dan durasi *roasting* yang optimal agar hasil *roasting* yang dihasilkan baik, sehingga kualitas produksi dari pengolahan biji kakao meningkat.

Kata kunci: *Data Mining*, Biji Kakao, *Roasting*, Suhu, Durasi, SVR, Regresi Linear, ELM, PSO

ABSTRACT

Data mining algorithms are starting to be used in various fields including the food industry. Thus, this data mining algorithm can of course also be applied to the cocoa industry, especially in the processing of cocoa beans. One of the processes in the processing of cocoa beans that most determines the quality of the processing of these cocoa beans is the roasting process. The roasting process is the process of removing the water content in the cocoa beans, drying them, and developing the beans to obtain a distinctive aroma and color according to standards. For good roasting results, the heating temperature and duration must be optimal. Temperature and duration are variables that have an important role in the roasting process. The purpose of this study is to make models for temperature recommendations on a certain roasting duration based on data on the roasting process such as roasting capacity, moisture content, pH, type of beans and so on. Temperature will be the target variable in this study. The author compares three data mining algorithms, namely Support Vector Regression (SVR), Multiple Linear Regression, Extreme Learning Machine (ELM), and Particle Swarm Optimization-Extreme Learning Machine (PSO-ELM) in this study. The data mining algorithm is evaluated using standard regression measures such as MAPE and RMSE to later compare its performance. After cross-validation, the best data mining algorithm for modeling temperature is PSO-ELM with MAPE and RMSE 4.13% and 7.36 respectively, then SVR with MAPE 4.76% and RMSE 9.17, then there is ELM with MAPE 4.79% and RMSE 8.62. and finally, there is MLR with MAPE 4.80% and RMSE 8.63. It is hoped that with this research, operators can determine the optimal roasting temperature and duration so that the resulting roasting results are good, so that the production quality of cocoa bean processing increases.

Keywords: *Data Mining, Cocoa Beans, Roasting, Temperature, Duration, SVR, Linear Regression, ELM, PSO*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan salah satu negara penghasil kakao terbesar di dunia. Tercatat pada tahun 2020, Indonesia mampu menghasilkan 659,7 ribu ton kakao [1] dengan nilai ekspor mencapai US$1,21 miliar [2]. Hal ini tentunya menjadikan kakao sebagai salah satu potensi komoditas perkebunan unggulan di Indonesia. Menurut data yang diambil dari situs web Kementerian Pertanian Republik Indonesia, pada tahun 2021, luas perkebunan kakao mencapai 1.497.467 Ha [3]. Selain menjadi komoditas perkebunan unggulan di Indonesia, berdasarkan hal tersebut kakao juga dapat menjadi sumber lapangan kerja bagi masyarakat sekitar perkebunan kakao.

Universitas Gadjah Mada melalui *UGM Cocoa Teaching and Learning Industry* ikut andil dalam menghasilkan produk kakao di Indonesia. *UGM Cocoa Teaching and Learning Industry* bertujuan untuk mendorong dan mempercepat program hilirisasi industri pengolahan kakao dan sebagai wahana produktif berbasis riset dan inovasi untuk mendukung proses pembelajaran yang bersinergi dengan industri. Industri coklat yang terletak di Kabupaten Batang tersebut merupakan hasil kerjasama dari berbagai pihak yaitu dari Universitas Gadjah Mada, Kementerian Perindustrian, Dikti, dan Pemerintah Kabupaten Batang. Industri ini sangat unik karena berlokasi di tengah-tengah perkebunan kakao warga [4].

Menurut wawancara dan kunjungan pabrik yang dilakukan oleh penulis, terdapat dua produk akhir yang dihasilkan pada *UGM Cocoa Teaching and Learning Industry* yaitu *butter* dan coklat bubuk. Sebelum menjadi *butter* dan coklat bubuk, tentunya biji kakao akan melalui beberapa proses produksi. Salah satunya adalah proses *roasting* atau penyangraian. Proses *roasting* adalah proses mengeluarkan kandungan air pada biji kakao, mengeringkannya, serta mengembangkan biji tersebut agar mendapatkan aroma dan warna yang khas. Variabel yang berpengaruh dalam proses *roasting* adalah waktu dan suhu yang diatur dalam proses *roasting* [5]. Agar hasil *roasting* bagus, maka suhu dan waktu *roasting* harus optimal. Pada penelitian ini, dengan menggunakan algoritma *data mining*, penulis akan merekomendasikan suhu dan durasi *roasting* berdasarkan data-data pada proses *roasting* seperti kapasitas roasting, kadar air, pH, jenis biji dan variabel lain. Penulis membandingkan tiga algoritma *data mining*, yaitu *Support Vector Regression (SVR)*, *Multiple Linear Regression*, *Extreme Learning Machine (ELM)*, dan *Particle Swarm Optimization-Extreme Learning Machine (PSO-ELM)* pada penelitian ini. Algoritma *data*

mining tersebut dievaluasi menggunakan ukuran standar regresi seperti MAPE dan RMSE untuk nantinya akan dibandingkan performanya. Dengan demikian, diharapkan operator dapat menentukan suhu dan durasi *roasting* yang optimal agar hasil *roasting* yang dihasilkan baik, sehingga kualitas produksi dari pengolahan biji kakao memiliki kualitas yang tinggi.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Apa saja variabel yang mempunyai tingkat signifikansi yang tinggi terhadap suhu *roasting*?
2. Bagaimana kinerja algoritma *data mining* dalam merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao?
3. Apa algoritma *data mining* yang mempunyai performa terbaik dalam merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao?

1.3 Tujuan Penelitian

Berikut adalah tujuan dari penelitian yang akan dilakukan :

1. Merancang algoritma *data mining* untuk merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao.
2. Membandingkan algoritma *data mining* untuk merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao.
3. Menentukan algoritma *data mining* terbaik untuk merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao.

1.4 Batasan Penelitian

Batasan penelitian pada penelitian ini adalah sebagai berikut :

1. Data yang digunakan adalah data produksi dan sensor suhu pada *roaster UGM Cocoa Teaching and Learning Industry* mulai bulan April 2021 sampai bulan Juli 2022 sebanyak 43 baris.
2. Komponen kimia dan fisis yang terlibat pada penelitian ini hanyalah pH dan kadar air.

1.5 Manfaat Penelitian

Setelah penelitian ini selesai, diharapkan penelitian ini bermanfaat bagi :

1. Dunia akademik, algoritma *data mining* yang berhasil dirancang pada penelitian ini diharapkan mampu menjadi pionir untuk penelitian-penelitian lanjutan dalam hal penerapan *data mining* pada industri coklat misalnya dalam pengembangan hardware seperti *smart-roaster*.
2. Operator Pabrik, sebelum mengoperasikan alat *roasting*, operator pabrik dapat terlebih dahulu melakukan simulasi menggunakan algoritma *data mining* yang sudah dikembangkan, sehingga dapat menentukan suhu dan durasi yang tepat dalam mengoperasikan alat *roaster*.
3. Pabrik Coklat, diharapkan kualitas produksi dari pabrik coklat yang menggunakan algoritma *data mining* ini meningkat

1.6 Sistematika Penulisan

1. BAB I Pendahuluan : Mengurai dan menjelaskan tentang latar belakang, rumusan masalah yang akan dijawab pada penelitian ini, batasan masalah yang membatasi pelaksanaan dari penelitian ini, tujuan yang akan dicapai pada penelitian ini, serta manfaat penelitian bagi pihak-pihak terkait.
2. BAB II Tinjauan Pustaka dan Dasar Teori : Didalamnya terdapat tinjauan pustaka yang merangkum penelitian-penelitian terdahulu kemudian mencari hal untuk dikembangkan dan dasar teori yang membahas tentang teori-teori yang mendasari penelitian yang dilakukan.
3. BAB III Metode Tugas Akhir : Menjelaskan tentang alat dan bahan yang digunakan dalam penelitian seperti komputer, dataset, baterai, dan lain sebagainya. Pada bab ini juga menjelaskan tentang alur penelitian dan hal-hal apa saja yang harus dilakukan untuk mencapai tujuan penelitian yang sedang dilakukan.
4. BAB IV Hasil dan Pembahasan : Membahas tentang hasil yang telah dicapai pada penelitian ini serta menjawab rumusan masalah. Pada bab ini juga dibahas perbandingan penelitian yang dilakukan oleh penulis dengan penelitian lainnya.
5. BAB V Kesimpulan dan Saran : Berisi tentang kesimpulan yaitu jawaban daripada tujuan penelitian ini, serta temuan-temuan setelah melakukan pengamatan dan analisis. Selain itu, pada bab ini juga terdapat saran yang diberikan untuk pengembangan penelitian yang telah dilakukan.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Pada tahun 2005 Misnawi dkk melakukan penelitian untuk menentukan suhu dan durasi *roasting* yang optimal menggunakan metode *Response Surface Methodology*. Parameter yang diamati pada penelitian tersebut adalah profil penyangraian, bilangan peroksida lemak, warna dan sifat organoleptik bubuk kakao yang dihasilkan serta uji mikrobiologi. Suhu yang digunakan berkisar antara 110-140 derajat celcius sementara untuk durasi *roasting* berkisar antara 20-60 menit dengan kapasitas roasting 15 kg. Penelitian itu menyatakan bahwa suhu dan durasi *roasting* berpengaruh terhadap bilangan peroksida lemak, warna, serta sifat organoleptik dari bubuk kakao yang dihasilkan. Suhu dan durasi *roasting* yang optimal menurut penelitian tersebut adalah 140 derajat celcius dan 20 menit [6].

Dengan metode yang sama yaitu *Response Surface Methodology*, pada tahun 2012 Misnawi dkk juga berusaha melakukan optimasi pada suhu dan durasi *roasting*. Namun, parameter yang diamati pada penelitian ini adalah konsentrasi dari pyrazine dan acrylamide yaitu komponen kimia di dalam biji kakao yang juga dapat memengaruhi rasa dari kakao itu sendiri. Suhu yang digunakan pada penelitian tersebut adalah 110-160 derajat celcius dengan durasi 15-40 menit. Hasil dari penelitian tersebut adalah bahwa pyrazine dipengaruhi oleh kondisi *roasting* sementara untuk acrylamide tidak dipengaruhi oleh kondisi *roasting*. Pada penelitian tersebut didapat suhu yang optimal yaitu 116 derajat celcius dengan durasi *roasting* 23 menit [7].

Ismara dkk pada tahun 2017 melakukan penelitian untuk melihat efek dari suhu dan durasi *roasting* terhadap karakteristik sensori seperti penampilan, aroma, rasa, dan tekstur. Metode yang digunakan sama seperti penelitian-penelitian sebelumnya yaitu *Response Surface Methodology*, ditambah *Principal Component Analysis* untuk melihat pengaruh suhu terhadap penampilan hasil *roasting*. Penelitian tersebut menyatakan bahwa suhu berpengaruh terhadap karakteristik sensori. Namun, untuk durasi *roasting* tidak berpengaruh terhadap karakteristik sensori. Suhu optimal yang didapat adalah 90-110 derajat celcius [8].

RSM dan PCA pada penelitian-penelitian di atas merupakan metode pendekatan dari ilmu statistika. Selain dari pendekatan ilmu statistika, *data mining* juga pernah digunakan untuk melakukan optimasi durasi *roasting*. Yang dkk melakukan penelitian berjudul "*Rapid determination of the roasting degree of cocoa beans by extreme learning machine (ELM)-based imaging analysis*". Tujuan dari penelitian ini adalah melakukan analisis pada gambar dan melihat

warna dari biji kakao yang sudah di-*roasting* berdasarkan level *roasting*. Apabila warna dan gambar dari biji kakao sudah dirasa sesuai dengan levelnya proses *roasting* dapat dihentikan. Hal tersebut membuat durasi *roasting* optimal [9].

Sedikit berbeda dari penelitian-penelitian sebelumnya, penelitian yang penulis lakukan adalah menentukan suhu yang optimal berdasarkan durasi *roasting* tertentu dan data-data produksi seperti kapasitas biji kakao, jenis biji, tipe produk, pH, serta kadar air. Penentuan suhu yang optimal dilakukan menggunakan algoritma *data mining*. Output dari penelitian ini adalah algoritma yang dapat menentukan suhu yang optimal berdasarkan durasi *roasting* tertentu dan inputan data produksi, bukan sebuah nilai skalar seperti pada penelitian sebelumnya. **Dengan demikian, penelitian ini diharapkan akan mempunyai skalabilitas dan fleksibilitas yang tinggi** karena rekomendasi suhu yang optimal akan berbeda-beda sesuai dengan inputan pada algoritma *data mining*.

2.2 Dasar Teori

2.2.1 Cocoa Nibs

Cocoa nibs adalah potongan kecil biji kakao yang dihancurkan yang memiliki rasa coklat yang pahit. Cocoa nibs diproduksi dari biji yang berasal dari pohon kakao *Theobroma,* juga dikenal sebagai pohon kakao. Biji kakao dikeringkan setelah dipanen, kemudian difermentasi dan dipecah untuk menghasilkan potongan-potongan kecil berwarna gelap [10].

Gambar 2.1 Cocoa Nibs

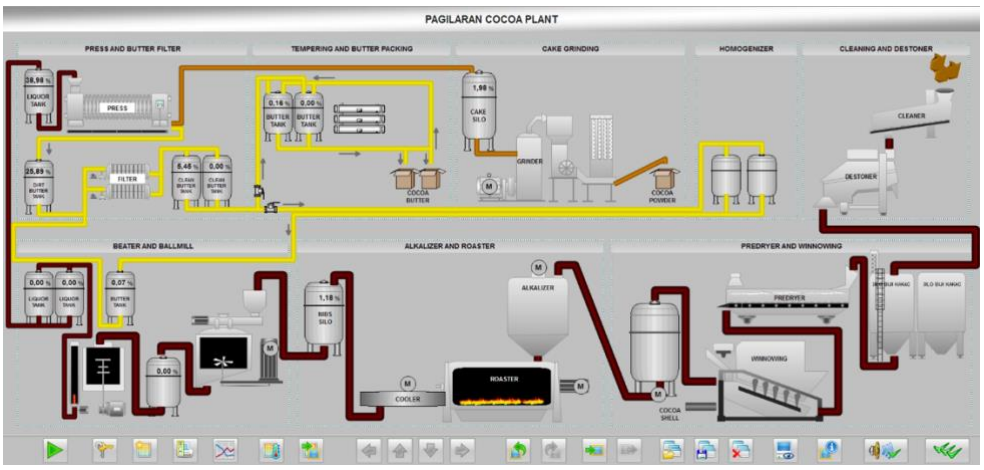
2.2.2 Cocoa Nibs Roasting

Langkah-langkah pra pemrosesan kakao (panen, pemecahan, fermentasi dan pengeringan) penting untuk memastikan kualitas biji yang tinggi, tetapi rasa kakao dipengaruhi selama proses fermentasi dan *roasting*. Biji kakao yang tidak di-*roasting* memiliki rasa pahit, asam, astringent, dan seperti kacang; dengan melakukan penyangraian biji kakao, keasaman akan berkurang dengan mengurangi konsentrasi asam volatil seperti asam asetat. *Roasting* atau

penyangraian adalah operasi teknologi terpenting dalam pengolahan biji kakao, dan tingkat perubahan kimianya tergantung pada suhu yang diterapkan selama proses. Sifat-sifat biji sangrai, seperti pembentukan warna coklat yang khas, tekstur biji sangrai, konsentrasi senyawa perasa yang mudah menguap, keasaman total dan kandungan lemak, bergantung pada kondisi penyangraian terutama suhu dan waktu pemrosesan [8].

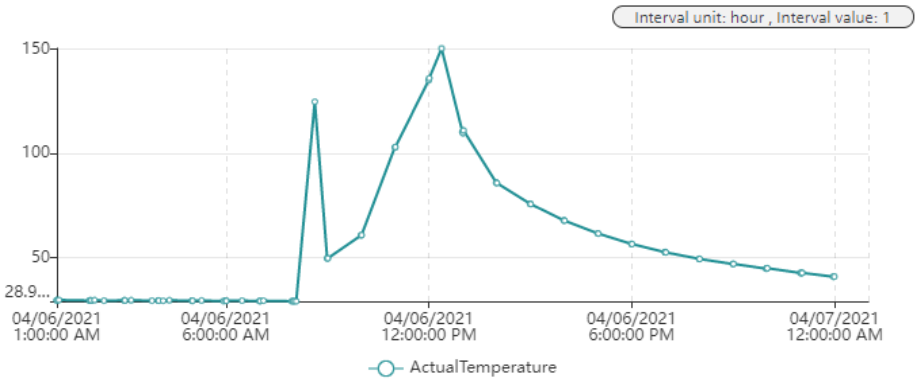
2.2.3 UGM Cocoa Teaching and Learning Industry (UGM CTLI)

UGM CTLI adalah pabrik pengolahan biji kakao sekaligus tempat pembelajaran dan pelatihan yang dikembangkan oleh UGM dalam menghasilkan produk kakao yang unggul di Indonesia. UGM CTLI bertujuan untuk mendorong dan mempercepat program hilirisasi industri pengolahan kakao dan sebagai wahana produktif berbasis riset dan inovasi untuk mendukung proses pembelajaran yang bersinergi dengan industri. Industri coklat yang terletak di Kabupaten Batang tersebut merupakan hasil kerjasama dari berbagai pihak yaitu dari Universitas Gadjah Mada, Kementerian Perindustrian, Dikti, dan Pemerintah Kabupaten Batang. Industri ini sangat unik karena berlokasi di tengah-tengah perkebunan kakao warga [4]. Menurut wawancara dan kunjungan pabrik yang dilakukan oleh penulis, terdapat dua produk akhir yang dihasilkan pada *UGM Cocoa Teaching and Learning Industry* yaitu *butter* dan coklat bubuk. Salah satu proses pada tahap produksi *butter* dan coklat bubuk adalah proses penyangraian. Semua proses yang ada pada UGM CTLI tersebut dapat diamati melalui sistem *monitoring*. Berikut adalah skema proses dari tahapan produksi pada UGM CTLI :



Gambar 2.2 UGM CTLI *Pagilaran Cocoa Plant*

Selain dapat melakukan monitoring, beberapa sensor dipasang pada alat yang digunakan untuk memproses biji kakao dan dapat diamati melalui aplikasi berbasis web bernama MindSphere. Berikut adalah contoh visualisasi suhu terhadap waktu pada alat *roasting*.



Gambar 2.3 Suhu *Roasting* terhadap Waktu

Selain data-data dari MindSphere, terdapat juga data-data dari tim laboratorium yang mencatat kuantitas fisik dan kimiawi seperti kadar air dan pH. Pekerja laboratorium mengonfirmasi bahwa suhu pada puncak kedua di gambar 2.3 merupakan suhu yang diatur oleh operator saat melakukan *roasting*.

2.2.4 Data Mining

Data mining atau penambangan data adalah proses ekstraksi informasi dan pola-pola yang berguna dari data yang tersedia. Tujuan dari penambangan data ini adalah untuk menemukan pola yang sebelumnya tidak diketahui. Setelah pola-pola ini ditemukan, selanjutnya dapat digunakan untuk membuat keputusan tertentu untuk pengembangan bisnis. Beberapa algoritma yang biasanya digunakan pada penambangan data adalah algoritma *clustering*, klasifikasi, regresi, *neural network* dan lain sebagainya. Penambangan data banyak diterapkan pada berbagai organisasi untuk menemukan pola dan koneksi yang sulit ditemukan. Misalnya untuk menentukan strategi marketing dari kebiasaan pelanggan [11].

2.2.5 Seleksi Fitur

Seleksi fitur adalah salah satu tahapan dalam proses data mining untuk memilih variabel yang relevan dengan variabel target dan membuang variabel yang tidak diperlukan atau redundan. Tujuan dari seleksi fitur adalah untuk meningkatkan performa, mempercepat waktu, dan mengurangi beban pada proses prediksi. Terdapat 3 metode populer yang digunakan dalam seleksi fitur yaitu *embedded, filter,* dan *wrapper* [12]. Seleksi fitur yang digunakan pada penelitian ini adalah seleksi fitur berbasis *filter*. Teknik seleksi fitur berbasis filter menggunakan metode statistik seperti kemiripan, ketergantungan, informasi, jarak untuk menunjukkan ketergantungan atau korelasi penting antara variabel input dan target [13]. Apabila variabel target

numerik dan variabel input numerik, korelasi pearson dapat digunakan. Jika variabel target numerik dan variabel input kategorik maka dapat digunakan ANOVA F-test [14].

2.2.6 Korelasi Pearson

Analisis korelasi adalah cara atau metode yang dapat digunakan untuk mengetahui hubungan antara variabel bebas dan variabel terikat. Derajat hubungan tersebut disebut koefisien korelasi [15]. Koefisien korelasi adalah koefisien yang merepresentasikan kedekatan hubungan antara dua variabel atau lebih. Besarnya koefisien korelasi tidak merepresentasikan hubungan sebab akibat antara dua variabel atau lebih, tetapi hanya merepresentasikan hubungan linier antar variabel [16]. Koefisien korelasi dapat ditulis dengan :

$$r = \frac{n\sum XY - (\sum X)(\sum Y)}{\sqrt{(n\sum X^2 - (\sum X)^2)(n\sum Y^2 - (\sum Y)^2)}} \quad (2.1)$$

Dengan X adalah variabel bebas, Y adalah variabel terikat, dan n adalah banyaknya data.
Pedoman untuk memberikan interpretasi koefisien korelasi adalah sebagai berikut [17]:

- 0.00 – 0.199 : Sangat rendah
- 0.20 – 0.399 : Rendah
- 0.40 – 0.599 : Sedang
- 0.60 – 0.799 : Kuat
- 0.80 – 1.000 : Sangat kuat

2.2.7 Analysis of Variance (ANOVA)

Analysis of Variance (ANOVA) adalah kumpulan model statistik parametrik dan prosedur estimasinya yang menentukan apakah rata-rata dari dua atau lebih sampel data berasal dari distribusi yang sama. Uji statistik pada ANOVA disebut juga *F-test. F-test* adalah serangkaian uji statistik untuk menghitung rasio nilai varians seperti varians dari dua sampel terpisah. Uji statistik ini adalah uji statistik univariat dimana setiap fitur atau variabel dibandingkan dengan fitur atau variabel target, untuk melihat apakah ada hubungan yang signifikan secara statistik di antara mereka. Skor F dapat didefinisikan sebagai :

$$F_{score} = \frac{\sum_{i=1}^{j} j_i(\overline{K_i} - \overline{K})^2/(S-1)}{\sum_{i=1}^{S}\sum_{p=1}^{j_i}\left(K_{ip} - \overline{K_i}\right)^2/(N-S)} \quad (2.2)$$

di mana N adalah ukuran sampel keseluruhan, S adalah jumlah kelompok, j_i adalah jumlah pengamatan dalam kelompok ke-j, $\overline{K_i}$ adalah rata-rata dari setiap grup, $\overline{K}$ adalah rata-rata dari keseluruhan data, dan K_{ip} adalah adalah pengamatan ke-p pada urutan ke-i dari kelompok S [18].

2.2.8 One-hot Encoding

One-Hot Encoding adalah skema pengkodean yang paling banyak digunakan. *One-Hot Encoding* membandingkan setiap tingkat variabel kategori dengan tingkat referensi tetap. Satu hot encoding mengubah satu variabel dengan n observasi dan d nilai berbeda, menjadi d variabel biner dengan masing-masing n observasi. Setiap pengamatan menunjukkan ada (1) atau tidak adanya (0) dari variabel biner dikotomis [19].

2.2.9 MinMax Normalization

MinMax *Normalization* adalah metode normalisasi menggunakan transformasi linear pada data asli sehingga menghasilkan data baru yang variasinya seimbang antara satu kolom dengan kolom lainnya [20]. MinMax *Normalization* dapat ditulis dengan perumusan sebagaimana berikut :

$$X_{new} = \frac{X_{old} - Min(X_{old})}{Max(X_{old}) - Min(X_{old})} \tag{2.3}$$

2.2.10 Support Vector Regression (SVR)

SVR adalah penerapan algoritma *Support Vector Machine* untuk kasus regresi. Dalam kasus regresi, output adalah bilangan real atau kontinu. SVR adalah metode yang dapat mengatasi over-fitting. Sehingga akan menghasilkan kinerja yang baik. Misalnya N adalah data latih (X, y) dengan menggunakan SVR, pengguna dapat menentukan fungsi $f(X)$. Fungsi tersebut memiliki deviasi terbesar E dari target sebenarnya untuk semua data pelatihan. Jika nilai E sama dengan 0 maka regresi dianggap sempurna. SVR disini bertujuan untuk menemukan fungsi regresi $f(X)$ yang dapat mendekati output ke target aktual, dengan toleransi kesalahan E, dan kompleksitas minimal. Fungsi regresi $f(X)$dapat dituliskan dengan :

$$f(X) = w^T \varphi(X) + B \tag{2.4}$$

$\varphi(X)$ menunjukkan suatu titik dalam ruang fitur berdimensi lebih tinggi dan hasil pemetaan input vektor X dalam ruang fitur berdimensi lebih rendah. Koefisien w dan b akan diestimasi menggunakan persamaan [21] :

$$min\frac{1}{2}\|w\|^2 + C\frac{1}{N}\sum_{i=1}^{N} L_E(y_i, f(X_i)) \quad (2.5)$$

$$y_i - w\varphi(X_i) - b \leq E \quad (2.6)$$

$$w\varphi(X_i) - y_i + b \leq E, \qquad i = 1,2,3,\ldots,N$$

Dimana,

$$L_E(y_i, f(X_i)) = |y_i - f(X_i)| - E|y_i - f(X_i)| \quad (2.7)$$

Terdapat 3 kernel yang digunakan pada SVR atau SVM pada umunya. Berikut adalah 3 kernel yang digunakan pada SVR [21] :

- Linear Kernel

$$k(x, y) = x^T \mathrm{y} + C \quad (2.8)$$

- Polynomial Kernel

$$k(x, y) = (ax^T \mathrm{y} + C)^d \quad (2.9)$$

- Radial Basis Function (RBF) Kernel

$$k(x, y) = \exp(-\gamma\|x - y\|^2)^d \quad (2.10)$$

2.2.11 Multiple Linear Regression (MLR)

Analisis regresi digunakan untuk menentukan korelasi antara dua atau lebih variabel yang mempunyai hubungan sebab-akibat. Selain itu analisis regresi juga dapat digunakan sebagai formula untuk memprediksi pada topik terkait menggunakan hubungan antar variabel. Regresi yang menggunakan satu variabel independen disebut regresi univariat, sementara regresi yang menggunakan lebih dari satu variabel independen dusebut regresi multivariat atau *multiple*. Dalam analisis regresi multivariat, dilakukan upaya untuk memperhitungkan variasi variabel independen dalam variabel dependen secara sinkron. Formula dari analisis regresi multivariat dapat dituliskan sebagai :

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \varepsilon \quad (2.11)$$

Dengan y adalah variabel dependen, X adalah variabel independen, β adalah parameter, dan ε adalah error [22].

2.2.12 Extreme Learning Machine (ELM)

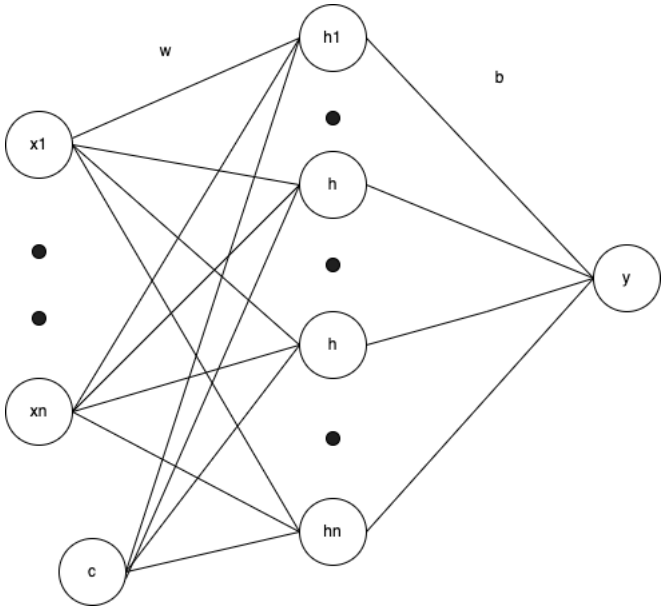
ELM adalah *Artificial Neural Network* berbasis *least-square* dan termasuk kedalam *single-layer feed-forward* yang dapat digunakan pada kasus klasifikasi maupun kasus regresi. Jumlah *neuron* atau *nod*e pada *hidden layer* berjumlah besar. ELM dapat diekspresikan sebagai :

$$e_j = \sum_{i=1}^{H} b_i f(w_i, c_i, x_j), \quad j = 1,2,\dots,N \tag{2.12}$$

H adalah jumlah *node* pada *hidden layer*, b adalah bobot pada *output layer*, w adalah bobot pada *input layer*, dan c adalah bias, serta x adalah *data input*, kemudian N adalah banyaknya *data input*. $f(w_i, c_i, x_j)$ merupakan fungsi aktivasi. Pada algoritma ELM biasa, nilai w dibuat *random* berdasarkan distribusi probabilitas. Nilai b yang merupakan bobot pada *output layer* dapat dicari menggunakan persamaan :

$$b = A^{+}Y \tag{2.13}$$

Dengan Y adalah vektor nilai aktual dan A^{+} adalah *pseudo-invers* dari matriks A. Dimana matriks A adalah matriks hasil dari *output layer* yang komponennya berisi nilai $f(w_i, c_i, x_j)$ [23]. Setelah kita mendapatkan w dan b maka w dan b tersebut bisa diujikan pada data uji. Arsitektur ELM dari penelitian ini adalah sebagai berikut :



Gambar 2.4 Arsitektur ELM

x adalah *input layer*, w adalah bobot input, c adalah bias, h adalah *hidden layer* yang komponennya adalah hasil dari fungsi aktivasi ReLU $f(w_i, c_i, x_j)$ dan y adalah *output layer*.

2.2.13 Grid Search

Grid search menguji semua kombinasi dari *hyperparameter* yang diberikan pada konfigurasi model *machine learning* [24]. *Grid search* membagi rentang parameter yang akan dioptimalkan ke dalam grid dan melintasi semua titik untuk mendapatkan parameter yang optimal. *Grid search* mengoptimalkan parameter SVM (dalam kasus ini SVR) menggunakan teknik validasi silang sebagai metrik kinerja [25]. *Grid search* dapat diimplementasikan menggunakan *library* sklearn dengan menggunakan perintah GridSearchCV. Beberapa keuntungan menggunakan grid search sebagai metode optimasi adalah sebagi berikut : penerapannya mudah, dapat menemukan λ yang jauh lebih baik daripada pengoptimalan sekuensial manual, dan keandalan dan dimensinya yang rendah [25].

2.2.14 Particle Swarm Optimization (PSO)

PSO termasuk di antara optimasi skohastik. Algoritma PSO mempekerjakan segerombolan partikel yang melintasi ruang pencarian multidimensi untuk mencari nilai optimal. Setiap partikel adalah solusi potensial dan dipengaruhi oleh pengalaman tetangganya serta dirinya sendiri [26]. Berikut adalah langkah-langkah dari PSO :

Populasi solusi potensial acak dirancang sebagai ruang pencarian. Misalkan D dan N masing-masing adalah dimensi ruang pencarian dan jumlah partikel. Setiap solusi potensial dinyatakan dalam posisi x_k^i dan kecepatannya v_k^i untuk N partikel dan k iterasi. Partikel-partikel ini kemudian "diterbangkan" melalui ruang pencarian solusi potensial. Menggunakan persamaan :

$$v_k^i(t+i) = wv_k^i(t) + c_1 rand() \left(p_k^i(t) - x_k^i(t)\right) + c_2 rand() \left(g_k^i(t) - x_k^i(t)\right) \quad (2.14)$$

$$x_k^i(t+i) = x_k^i(t) + v_k^i(t+i) \quad (2.15)$$

w adalah bobot iterasi, c adalah inersia p_k^iadalah solusi terbaik pada 1 partikel tertentu dan g_k^i adalah solusi terbaik global. Nilai w yang merupakan solusi potensial dievaluasi pada fungsi objektifnya Setelah itu untuk setiap iterasi dan berdasarkan iterasi sebelumnya ditentukan lah nilai p_k^i dan g_k^i. Algoritma tersebut akan terus beriterasi mencari g_k^i sampai konvergensi tercapai berdasarkan kriteria yang diinginkan [27].

PSO pada penelitian ini berguna sebagai algoritma optimasi untuk mencari nilai bobot input dan bias pada ELM. Fungsi objektif yang digunakan adalah RMSE. PSO-ELM akan terus beriterasi agar menemukan nilai RMSE yang minimal. Pada kondisi tersebutlah ELM berhasil dioptimasi bobot dan biasnya sehingga diharapkan akan memiliki performa yang lebih baik dibanding dengan menginisiasi bobot dan bias dengan nilai random.

2.2.15 Regression Evaluation Metrics

Terdapat 2 ukuran evaluasi yang akan digunakan pada penelitian kali ini :

1. *Mean Absolute Percentage Error* (MAPE)

   MAPE adalah salah satu ukuran standar yang digunakan dalam peramalan. MAPE adalah rata-rata absolut dari persen error saat nilai prediksi dan nilai aktual dibandingkan. MAPE dapat ditulis sebagai :

   $$MAPE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{A_t - F_t}{A_t}\right| \quad (2.16)$$

   N adalah jumlah data yang diprediksi, A_t adalah nilai aktual dan F_t adalah nilai prediksi. Persamaan (2.16) harus dikalikan 100 untuk menjadi persentase. MAPE memiliki skala independen dan mudah ditafsirkan, yang membuatnya populer di kalangan praktisi industri [28].

2. *Root Mean Squared Error* (RMSE)

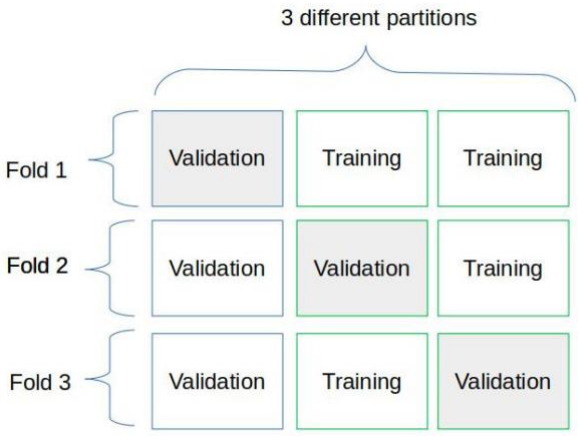
   RMSE adalah akar kuadrat dari rata-rata kuadrat dari semua kesalahan. Penggunaan RMSE sangat umum, dan dianggap sebagai metrik kesalahan tujuan umum yang sangat baik untuk prediksi numerik. Secara matematis, RMSE dapat ditulis dengan :

   $$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(A_i - F_i)^2} \quad (2.17)$$

   Dengan N adalah banyaknya data, A_i adalah nilai aktual dan F_i adalah nilai prediksi. RMSE adalah ukuran akurasi yang baik, tetapi hanya untuk membandingkan kesalahan peramalan dari model yang berbeda atau konfigurasi model untuk variabel tertentu [29].

2.2.16 Cross-validation

Cross-Validation atau validasi silang adalah metode statistik untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua segmen: satu digunakan untuk mempelajari atau melatih model dan yang lainnya digunakan untuk memvalidasi model. Dalam validasi silang, set pelatihan dan validasi harus saling silang dalam putaran berturut-turut sehingga setiap titik data memiliki peluang untuk divalidasi. Bentuk dasar dari cross-validation adalah *k-fold cross-validation* [30]. Skema dari validasi silang ini dapat dilihat pada gambar di bawah ini :



Gambar 2.5 *Skema 3-Fold Cross-Validation*

2.3 Analisis Perbandingan Metode

Sebagaimana yang telah dibahas pada bab 2.2, telah diketahui bahwa diharapkan penelitian ini, dengan menggunakan algoritma *data mining* mampu merekomendasikan suhu dan durasi *roasting* yang optimal berdasarkan data inputan seperti kapasitas, jenis biji, tipe produk, kadar air, dan pH. Dibandingkan dengan penelitian sebelumnya menggunakan metode *Response Surface Methodology* (RSM) yang menghasilkan suhu dan durasi *roasting* berupa sebuah nilai, maka penelitian ini akan merekomendasikan suhu dan durasi *roasting* yang berbeda-beda sesuai dengan inputan atau kondisi *roasting*. Dengan demikian, algoritma yang dihasilkan pada penelitian ini diharapkan akan mempunyai skalabilitas dan fleksibilitas yang tinggi.

Salah satu tantangan yang ada pada penelitian ini berdasarkan batasan masalah adalah jumlah datanya yang sedikit yaitu hanya 43 baris data. Karena hal tersebutlah penulis mencoba beberapa algoritma *data mining* dan membandingkan performanya untuk melihat mana algoritma *data mining* yang mampu diterapkan pada dataset yang jumlahnya sedikit. Sebelumnya banyak penelitian-penelitian yang sudah membandingkan algoritma-algoritma tersebut namun pada dataset yang tidak sedikit dan pada kasus klasifikasi, bukan regresi.

Penelitian yang dilakukan oleh Rath dkk dengan judul "*A Comparative Analysis of SVM and ELM Classification on Software Reliability Prediction Model*" mendapatkan bahwa Model klasifikasi ELM adalah yang paling akurat yaitu dengan 84.61 dibanding SVM yang mempunyai akurasi sebesar 78.68. Penelitian tersebut juga menunjukan bahwa ELM juga membutuhkan waktu lebih sedikit daripada model klasifikasi SVM [31].

Penelitian selanjutnya yang diaksanakan oleh Ying Xu dkk dengan judul "*A Binaural Sound Localization System using Deep Convolutional Neural Networks*" menunjukan bahwa ELM

mempunyai performa yang lebih baik dibanding regresi linear dengan perbandingan skor RMSE sebesar 27.32 untuk regresi linear dan 19.11 untuk ELM [32].

Berdasarkan kedua penelitian di atas, performa ELM pada kasus klasifikasi dan regresi mampu mengungguli SVM dan regresi linear. Penambahan algoritma optimasi PSO pada ELM juga diharapkan dapat meningkatkan performa dari model ELM itu sendiri, karena menurut jurnal yang ditulis oleh Mosbeh R. Kaloop dkk dengan judul "*Particle Swarm Optimization Algorithm-Extreme Learning Machine (PSO-ELM) Model for Predicting Resilient Modulus of Stabilized Aggregate Bases*" memperlihatkan bahwa penambahan PSO sebagai algoritma optimasi untuk inisiasi bobot awal dan bias mampu meningkatkan performa pada model secara signifikan, dari skor RMSE sebesar 1075 pada data tes, menjadi RMSE sebesar 369 [27].

BAB III
METODE TUGAS AKHIR

3.1 Alat dan Bahan Tugas akhir

3.1.1 Alat Tugas akhir

Pada Tugas Akhir ini, digunakan beberapa perangkat keras dan perangkat lunak sebagai berikut:

1. Laptop Acer Aspire 3, Windows 10, AMD Ryzen 3 2200U CPU, 12GB DDR4 RAM, Radeon Vega 3 GPU.
2. Google Chrome versi 106.0.5249.119 (Official Build) (64-bit) sebagai *browser* pengakses Google Collaboratory, Google Spreadsheet, Google Drive, dan MindSphere
3. Google Drive dengan 15 GB disk space, sebagai penyimpanan data *roasting*
4. Google Spreadsheet, sebagai tempat kerja untuk merapihkan data
5. Google Collaboratory Notebook, dengan VM Intel(R) Xeon(R) CPU @ 2.20GHz dan 12 GB RAM sebagai tempat pengolahan data
6. MindSphere, sebagai *software* pemantauan temperatur *roaster*

3.1.2 Bahan Tugas akhir

Bahan yang digunakan adalah dataset pemantauan produksi dari tim laboratorium *UGM Cocoa Teaching and Learning Industry* serta data suhu *roaster* dari MindSphere. Data yang digunakan mempunyai beberapa variabel sebagai berikut :

Tabel 3.1 Kamus Data Penelitian

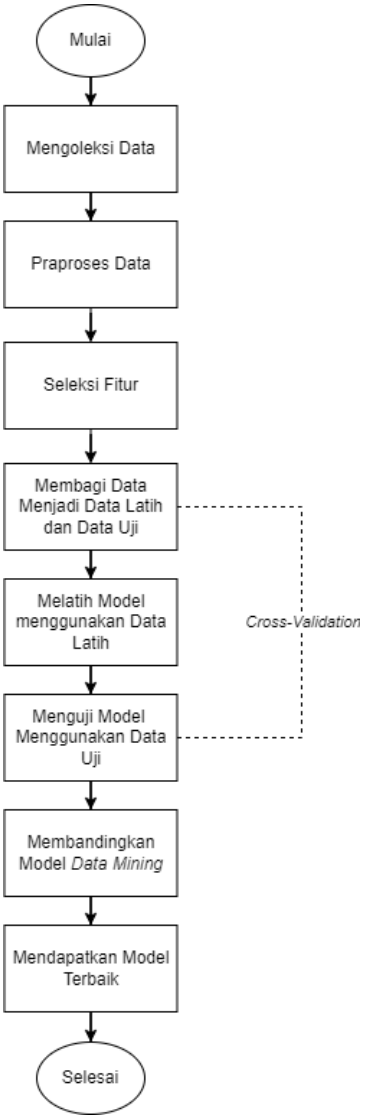
Kolom	Keterangan
nibs_capacity	kapasitas nibs (kg)
solution_load	jumlah air yang dibutuhkan (L)
beans_source	wilayah kebun dari biji tersebut
is_alkalized	apakah proses tersebut mengalami alkilasi?
product_type	tipe produk
durasi_roasting	durasi *roasting*
suhu	suhu pengaturan
pH_0	pH awal biji kakao

pH_N	pH akhir biji kakao
moist_0	kadar air (%) awal biji kakao
moist_N	kadar air (%) akhir biji kakao

Pada penelitian ini, model yang dikembangkan memiliki variabel dependen yaitu suhu dan variable independennya adalah variabel lainnya selain variabel suhu.

3.2 Alur Tugas Akhir

Diagram di bawah ini menunjukan alur daripada Tugas Akhir atau penelitian yang dikerjakan :



Gambar 3.1 Alur Tugas Akhir

1. Mengoleksi Data

   Tahap yang pertama adalah mengoleksi data. Data yang diperoleh berasal dari data laboratorium *UGM Cocoa Teaching and Learning Industry* untuk hal-hal yang berkaitan dengan biji kakao. Kemudian untuk temperatur atau suhu, data diperoleh dari sensor yang dipasang pada *roaster* melalui aplikasi MindSphere.

2. Pra Proses Data

   Pada tahap ini penulis melakukan beberapa hal yaitu menghapus *outlier*, data *encoding* untuk kolom kategorik, yaitu beans_source dan product_type. Penulis melakukan *one-hot encoding* untuk menangani kolom kategorik ini. Langkah terakhir pada tahap ini adalah melakukan normalisasi menggunakan MinMax Normalization pada data agar variasinya seimbang.

3. Seleksi Fitur

   Seleksi fitur dilakukan untuk mengeliminasi fitur atau variabel bebas. Seleksi fitur pada penelitian ini menggunakan metode statistik korelasi pearson untuk data numerik dan ANOVA untuk data kategorik. Penulis melakukan uji statistik dan melihat p-*value* dengan tingkat kepercayaan 99% untuk menentukan apakah sebuah variabel layak dimasukan ke dalam model atau tidak.

4. Membagi Data Latih dan Data Uji

   Pembagian data latih dan data uji bertujuan untuk melatih algoritma menggunakan data latih, kemudian mengujinya pada data uji. Pembagian data latih dan data uji menggunakan skema *3-fold Cross Validation*.

5. Melatih Algoritma dengan Data Latih

   Algoritma *data mining* atau model dilatih menggunakan dengan data latih yang mempunyai proporsi 2:1 dari data total. Model dilatih pada setiap *fold* menggunkan skema *3-fold Cross Validation*. Algoritma yang digunakan yaitu *Support Vector Regression*, *Multiple Linear Regression*, *Extreme Learning Machine*, dan *Particle Swarm Optimization-Extreme Learning Machine*. Untuk SVR terlebih dahulu dilakukan *hyperparameter tuning* yang berguna untuk mencari *hyperparameter* yang optimal menggunakan *grid search*, sementara untuk PSO-ELM, sebelum melatih algoritma, dicari bobot awal dan bias yang optimal terlebih dahulu menggunakan PSO yang kemudian bobot awal dan bias tersebut akan digunakan pada model ELM.

6. Menguji Algoritma dengan Data Uji

   Algoritma *data mining* atau model yang sudah dilatih kemudian diuji menggunakan data uji dengan ukuran pengujian yaitu MAPE dan RMSE untuk melihat performa dari

algoritma yang telah dibuat. Hasil pengujian pada kelima *fold* tersebut akan dirata-ratakan untuk melihat performa akhir dari setiap model.

7. Membandingkan Algoritma *Data Mining*

   SVR, *Multiple Linear Regression* (MLR*)*, ELM, PSO-ELM akan dibandingkan skor performanya menggunakan ukuran evaluasi MAPE dan RMSE.

8. Mendapatkan Algoritma *Data Mining* Terbaik

   Model yang memiliki performa terbaik yaitu yang memiliki nilai MAPE dan RMSE terkecil adalah model yang memiliki performa terbaik untuk menentukan suhu yang optimal.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Seleksi Fitur

Seleksi fitur diharapkan mampu memilih variabel independen yang mempunyai hubungan yang kuat dengan variabel dependen sehingga performa dari model yang dibuat lebih baik. Untuk variabel numerik, seleksi fitur dilakukan menggunakan metode statistik pearson sementara untuk variabel kategorik, seleksi fitur dilakukan menggunakan metode statistika *ANOVA F-score*, pada kedua metode tersebut dilakukan uji hipotesis dengan melihat p-value. Penelitian ini menggunakan tingkat kepercayaan 0.01 sehingga apabila p-value < 0.01 maka hipotesis 0 ditolak dan hipotesis 1 diterima. Berikut adalah pernyataan dari hipotesis 0 dan hipotesis 1 :

- H0 : variabel bebas x tidak memiliki hubungan dengan variabel target y (suhu)
- H1 : variabel bebas x memiliki hubungan dengan variabel target y (suhu)

Tabel 4.1 Hasil Uji Statistik Variabel Numerik

Variabel	r_suhu	pvalue_suhu
nibs_capacity	-0.5089	0.0009
solution_load	-0.2962	0.0671
pH_0	0.0392	0.8125
pH_N	0.1251	0.4480
delta_pH	-0.1771	0.2807
moist_0	-0.0248	0.8809
moist_N	-0.5860	0.0001
delta_moist	0.2155	0.1876

Berdasarkan uji statistik variabel numerik tersebut, untuk variabel dependen y yaitu suhu, terdapat 2 variabel independen yang secara signifikan berhubungan dengan variabel dependen suhu yaitu nibs_capacity dan moist_N. Dengan demikian, **variabel numerik yang akan dipilih** pada pemodelan menggunakan algoritma *data mining* adalah **nibs_capacity dan moist_N serta durasi_roasting**, durasi_roasting juga dipilih dalam pemodelan untuk menghubungkan antara durasi_roasting dengan suhu.

Tabel 4.2 Hasil Uji Statistik Variabel Kategorik

Variabel	fscore_suhu	pvalue_suhu
beans_source	29.522289	0.000004
product_type	0.512261	0.478655
is_alkalized	6.490727	0.015131

Berdasarkan uji statistik pada variabel kategorik, didapati bahwa terdapat 1 variabel independen yang secara signifikan mempunyai hubungan dengan variabel dependen suhu yaitu beans_source. Dengan demikian, **variabel kategorik yang akan digunakan** pada tahap pemodelan adalah variabel **beans_source** saja**.** Dari sekian banyak variabel yang ada pada data yang tersedia, variabel independen yang mempunyai tingkat signifikansi yang tinggi dengan variabel suhu adalah sebagai berikut:

- nibs_capacity
- moist_N
- durasi_roasting (untuk mengoptimalisasi durasi *roasting* dan suhu *roasting*)
- beans_source

Variabel tersebut akan dijadikan variabel indpenden pada model, dan yang menjadi variabel dependen atau variabel targetnya adalah suhu.

4.2 Pemodelan menggunakan SVR

Algoritma atau model pertama yang digunakan adalah *Support Vector Regressor* atau SVR. Tahap pertama yang dilakukan adalah mencari *hyperparameter* optimal menggunakan *grid search*. *Hyperparameter* yang dicari adalah C, gamma, epsilon, dan kernel. Berikut adalah *range* atau kumpulan nilai *hyperparameter* yang akan dicari nilai optimalnya :

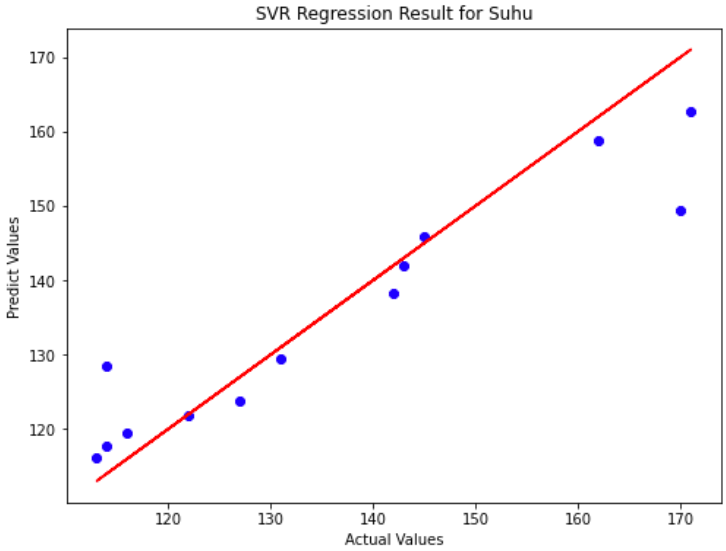
```
parameter = {'C': [0.1, 0.001, 1, 10, 12, 14, 16, 18, 20, 22],
             'gamma': [0.001, 0.01, 0.1, 1, 2, 5],
             'epsilon': [0.001, 0.01, 0.1, 1, 2, 4],
             'kernel': ("rbf", "poly", "linear")}
```

Setelah dilakukan pencarian *hyperparameter* yang optimal menggunakan grid search dengan 3-*fold* validasi silang, didapat *hyperparameter* yang optimal sebagai berikut :

```
{'C': 22, 'epsilon': 1, 'gamma': 1, 'kernel': 'rbf'}
```

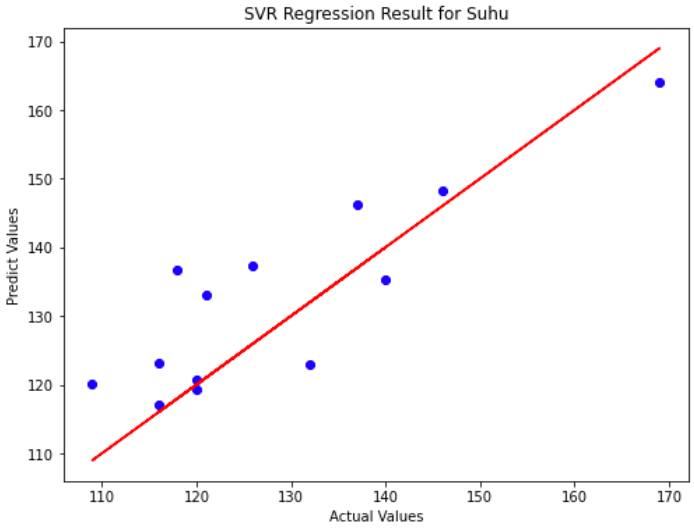
Skor rata-rata RMSE terbaik yang didapat dengan parameter tersebut sebesar 9.17 dan untuk skor rata-rata MAPE sebesar 4.76%. Berikut adalah hasil pengujian dari model tersebut untuk setiap *fold*-nya :

- *Fold 1*



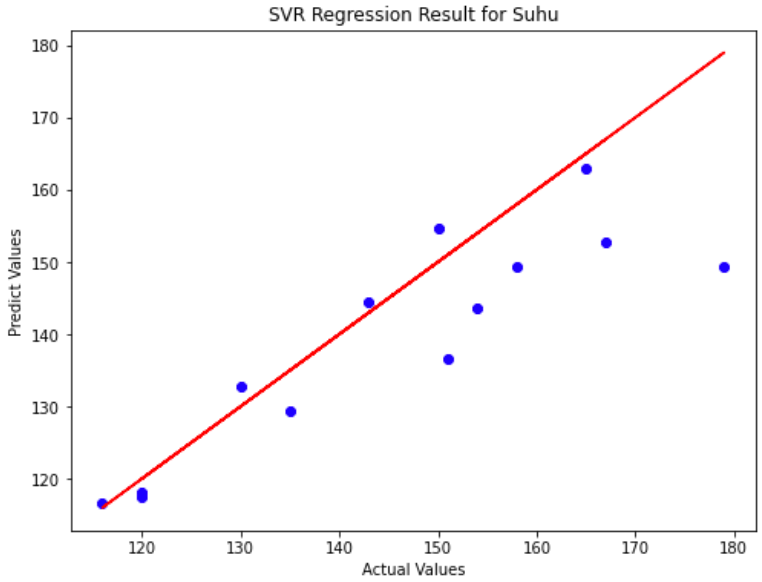
Gambar 4.1 Hasil Pengujian SVR pada Fold 1

- *Fold 2*



Gambar 4.2 Hasil Pengujian SVR pada Fold 2

- *Fold 3*



Gambar 4.3 Hasil Pengujian SVR pada Fold 3

Gambar di atas menunjukan perbandingan nilai prediksi dan aktual pada data tes untuk setiap *fold* pada pemodelan suhu menggunakan SVR. Untuk *fold* pertama, didapat nilai MAPE sebesar 3.71% dan nilai RMSE sebesar 7.72, pada *fold* kedua didapat nilai MAPE sebesar 5.73% dan nilai RMSE sebesar 8.85, dan pada fold ketiga didapat nilai MAPE sebesar 4.84% dan nilai RMSE sebesar 10.94. Apabila dirata-ratakan, pada pemodelan menggunakan SVR di dapat rata-rata MAPE sebesar 4.76 (1.01) dan RMSE sebesar 9.17 (1.63). Berikut adalah tabel lengkap dari hasil pengujiannya :

Tabel 4.3 Hasil Pengujian Model SVR

SVR		
Ukuran	**MAPE**	**RMSE**
Fold 1	3.71	7.72
Fold 2	5.73	8.85
Fold 3	4.84	10.94
Rata-rata	**4.76**	**9.17**
Std Dev	**1.01**	**1.63**

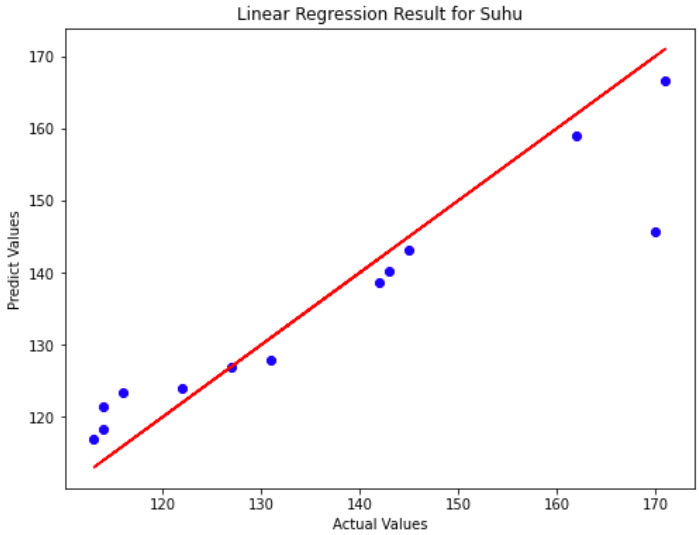
4.3 Pemodelan menggunakan MLR

Pada penelitian ini, MLR diimplementasikan menggunakan metode *least-square* pada pustaka scikit-learn menggunakan fungsi LinearRegression(). Fungsi yang dibentuk oleh model tersebut adalah sebagai berikut :

$$suhu = 153.84 - 6.36 \cdot nibs\_capacity - 15.99 \cdot moist\_N - 16.97 \cdot durasi\_roasting + categoric(source) \quad (2.18)$$

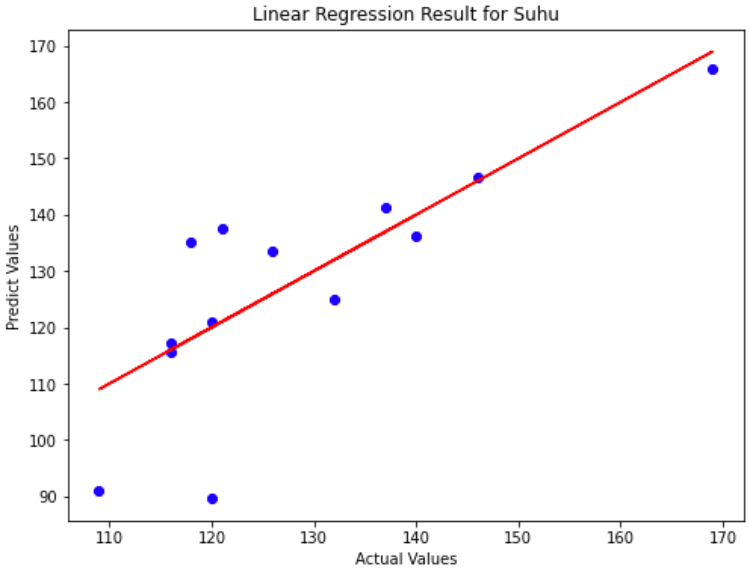
Dengan *categoric* bernilai -18.26 untuk beans *source* ADACHI, 0.10 untuk bean source BONDO, -19.46 untuk beans source GOBEL, -17.18 untuk beans source HMGN, -13.08 untuk beans source papua, 2.34 untuk beans source PGL, 21.89 untuk beans source SGU, 36.42 untuk beans source siklon, dan 7.23 untuk beans source UNID KOREA. Berikut adalah hasil pengujian dari model tersebut untuk setiap *fold*-nya :

- *Fold 1*



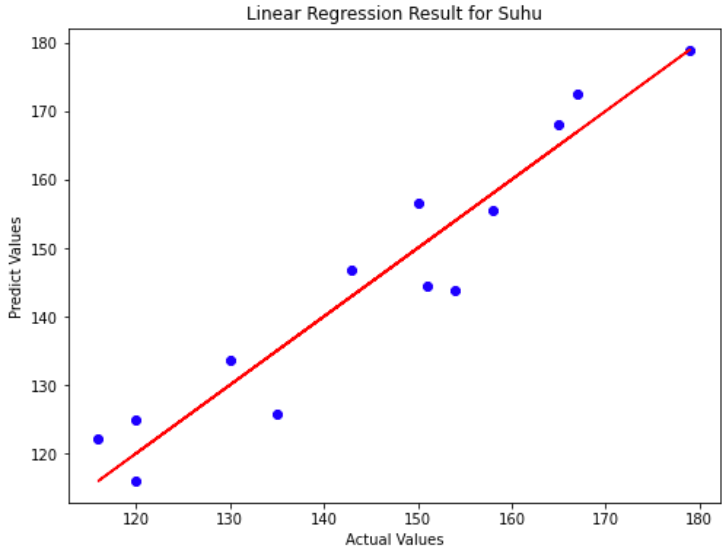
Gambar 4.4 Hasil Pengujian MLR pada Fold 1

- *Fold 2*



Gambar 4.5 Hasil Pengujian MLR pada Fold 2

- *Fold 3*



Gambar 4.6 Hasil Pengujian MLR pada Fold 3

Gambar 4.4 sampai 4.6 memperlihatkan perbandingan antara nilai aktual dan nilai prediksi dari pemodelan suhu menggunakan MLR. Untuk *fold* pertama, didapat nilai MAPE sebesar 3.72% dan nilai RMSE sebesar 7.83, pada *fold* kedua didapat nilai MAPE sebesar 7.06% dan nilai RMSE sebesar 12.32, dan pada fold ketiga didapat nilai MAPE sebesar 3.62% dan nilai RMSE sebesar 5.73. Apabila dirata-ratakan, pada pemodelan menggunakan SVR di dapat rata-rata MAPE sebesar 4.80% (1.96) dan RMSE sebesar 8.63 (3.37). Berikut adalah tabel lengkap dari hasil pengujiannya :

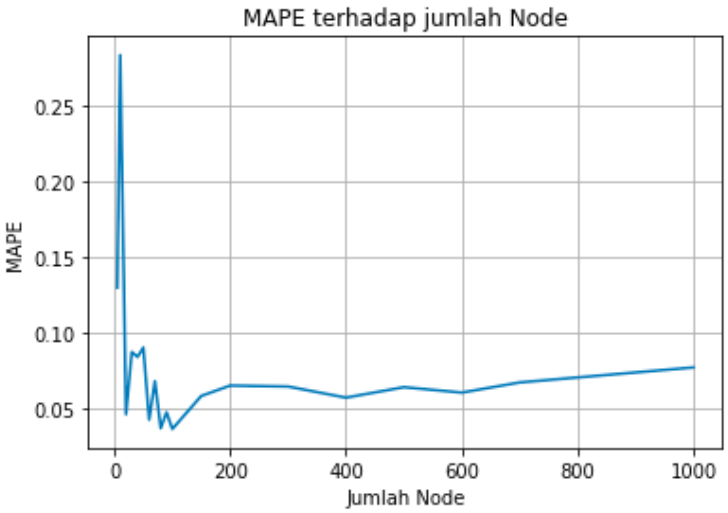
Tabel 4.4 Hasil Pengujian Model MLR

MLR		
Ukuran	**MAPE**	**RMSE**
Fold 1	3.72	7.83
Fold 2	7.06	12.32
Fold 3	3.62	5.73
Rata-rata	**4.8**	**8.63**
Std Dev	**1.96**	**3.37**

Apabila dibandingkan dengan model SVR nilai rata-rata MAPE pada pemodelan suhu menggunakan MLR lebih besar, namun tidak terlalu signifikan perbedaannya. Kemudian untuk nilai rata-rata RMSE, model MLR memiliki nilai yang lebih kecil dibandingkan dengan model SVR. Walaupun model MLR memiliki nilai error rata-rata yang lebih kecil dibandingkan denghan model SVR, namun model MLR memiliki standar deviasi yang lebih besar. Hal ini menunjukan bahwa model SVR lebih stabil untuk memodelkan suhu pada setiap foldnya dibandingkan dengan model MLR. Terlihat pada fold 2, model MLR kesulitan untuk memodelkan suhu karena memiliki error yang jauh lebih besar dibanding model SVR.

4.4 Pemodelan menggunakan ELM

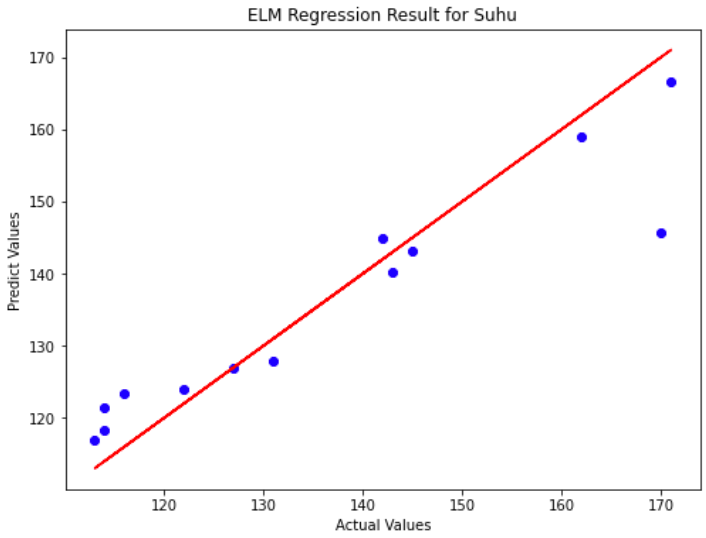
Algoritma ketiga yang diuji adalah ELM. Sama seperti algoritma sebelumnya, ELM diuji menggunakan skema *3-fold*. Pada penelitian ini ELM yang digunakan memiliki 1 *hidden layer* dengan 100 *nodes*. 100 *nodes* dipilih karena berdasarkan hasil percobaan pada berbagai jumlah *nodes*, jumlah *nodes* 100 memiliki MAPE terkecil dengan 3.69%.



Gambar 4.7 Nilai MAPE pada Percobaan Jumlah Node

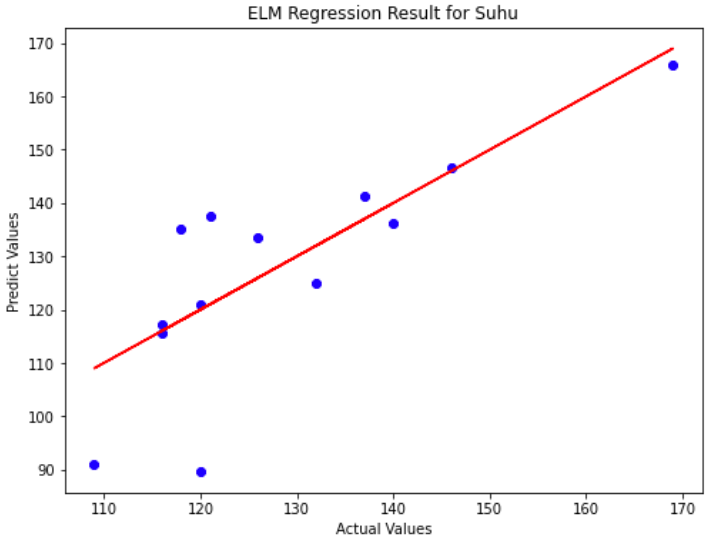
Setelah mendapatkan jumlah *node* dengan MAPE terkecil, selanjutnya akan dilakukan pengujian pada data tes menggunakan skema *3-fold cross-validation.* Berikut adalah hasil dari pengujian algoritma ELM dengan skema *3-fold* untuk setiap *fold*-nya :

- *Fold 1*



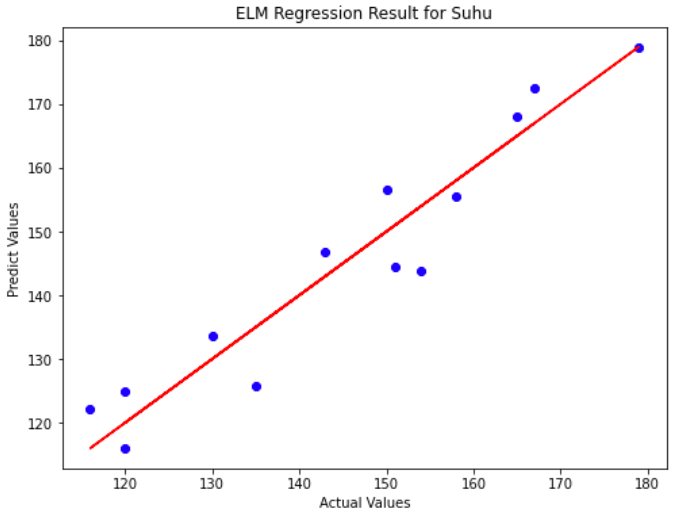
Gambar 4.8 Hasil Pengujian ELM pada Fold 1

- *Fold 2*



Gambar 4.9 Hasil Pengujian ELM pada Fold 2

- *Fold 3*



Gambar 4.10 Hasil Pengujian ELM pada Fold 3

Pada gambar 4.8 sampai 4.10 dapat dilihat perbandingan antara nilai aktual dan nilai prediksi dari pemodelan suhu menggunakan ELM. Untuk *fold* pertama, didapat nilai MAPE sebesar 3.69% dan nilai RMSE sebesar 7.81, pada f*old* kedua didapat nilai MAPE sebesar 7.06% dan nilai RMSE sebesar 12.32, dan pada *fold* ketiga didapat nilai MAPE sebesar 3.62% dan nilai RMSE sebesar 5.73. Apabila dirata-ratakan, pada pemodelan menggunakan SVR di dapat rata-rata MAPE sebesar 4.79% (1.97) dan RMSE sebesar 8.62 (3.37). Berikut adalah tabel lengkap dari hasil pengujiannya :

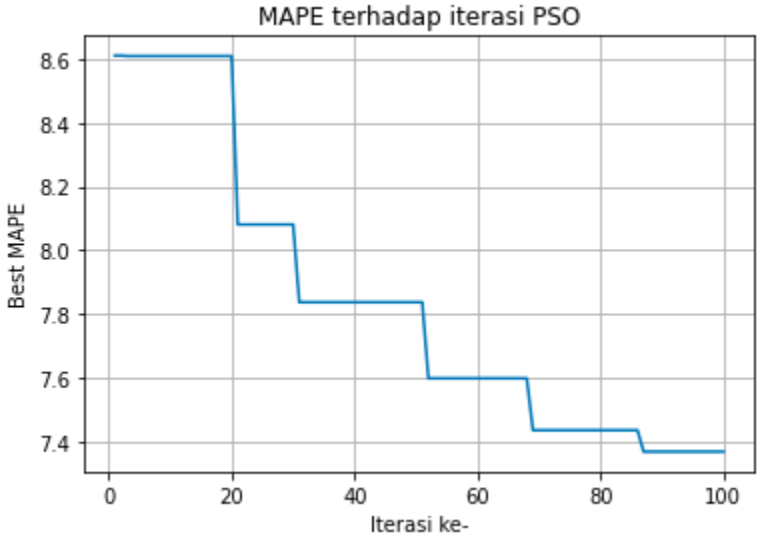
Tabel 4.5 Hasil Pengujian Model ELM

ELM		
Ukuran	**MAPE**	**RMSE**
Fold 1	3.69	7.81
Fold 2	7.06	12.32
Fold 3	3.62	5.73
Rata-rata	**4.79**	**8.62**
Std Dev	**1.97**	**3.37**

Hasil pemodelan menggunakan ELM memiliki performa yang mirip dengan pemodelan menggunakan MLR. Model ELM sedikit lebih baik memprediksi suhu pada *fold* 1 dibanding dengan model MLR. Sama seperti model MLR, model ELM kesulitan untuk memodelkan suhu pada fold 2.

4.5 Pemodelan menggunakan PSO-ELM

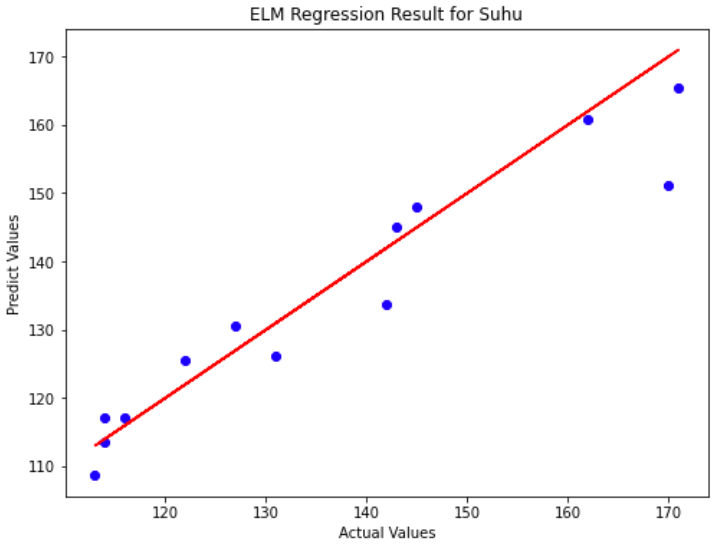
Algoritma terakhir yang diujikan pada penelitian ini adalah PSO-ELM. PSO bertindak sebagai algortima optimasi dari bobot awal dan bias yang nantinya akan diikutkan pada perhitungan dalam model ELM. Fungsi objektif yang digunakan pada PSO adalah nilai RMSE, sehingga PSO dikatakan berhasil atau optimal apabila memiliki nilai RMSE yang paling kecil. PSO akan melakukan iterasi sebanyak 100 kali dengan 10 kandidat bobot awal dan bias. Berikut adalah nilai MAPE untuk setiap iterasi yang dilakukan oleh algoritma PSO :



Gambar 4.11 Nilai RMSE terhadap iterasi PSO

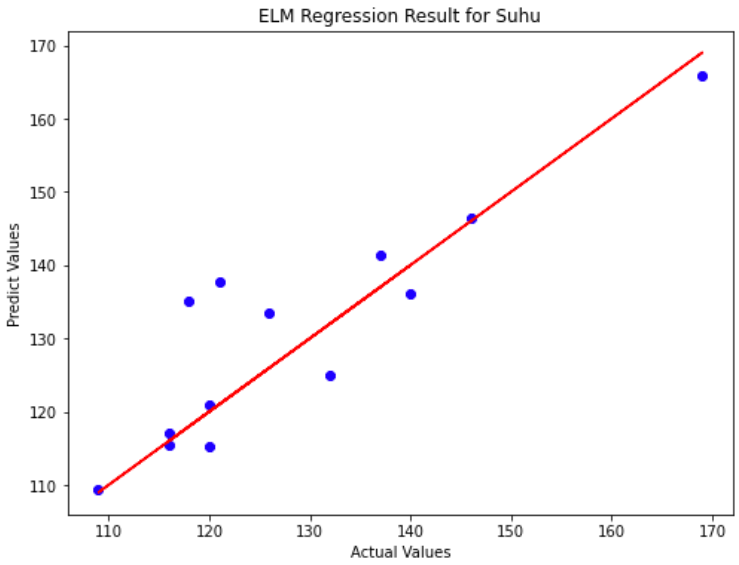
Didapat bahwa sampai iterasi ke-100 terdapat improvisasi nilai RMSE rata-rata menjadi 7.37. Setelah mendapat bobot awal dan bias yang optimal, algoritma tersebut diujikan menggunakan skema *3-fold*. Berikut adalah hasil pengujian dari algoritma PSO-ELM :

- *Fold 1*



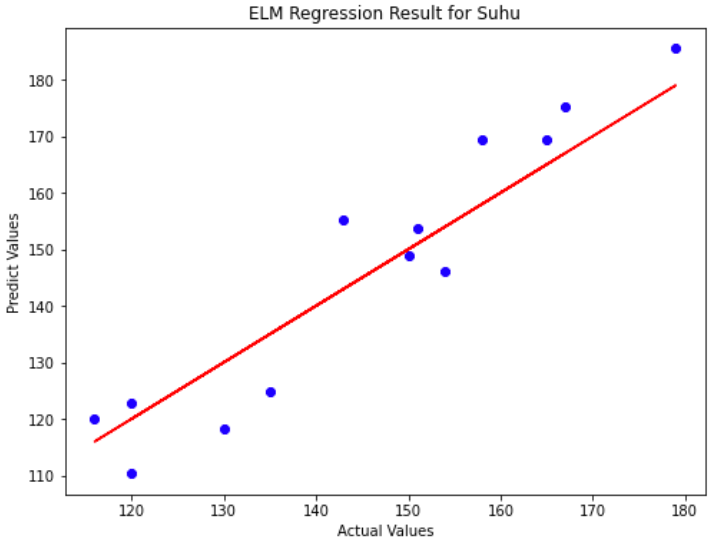
Gambar 4.12 Hasil Pengujian PSO-ELM pada Fold 1

- *Fold 2*



Gambar 4.13 Hasil Pengujian PSO-ELM pada Fold 2

- *Fold 3*



Gambar 4.14 Hasil Pengujian PSO-ELM pada Fold 3

Gambar 4.12 sampai 4.14 memperlihatkan perbandingan nilai aktual dan prediksi pada pemodelan suhu menggunakan PSO-ELM dengan skema *3-fold*. Untuk *fold* pertama, didapat nilai MAPE sebesar 3.21% dan nilai RMSE sebesar 6.48, pada *fold* kedua didapat nilai MAPE sebesar 4.91% dan nilai RMSE sebesar 7.59, dan pada *fold* ketiga didapat nilai MAPE sebesar 5.01% dan nilai RMSE sebesar 8.03. Apabila dirata-ratakan, pada pemodelan menggunakan SVR di dapat rata-rata MAPE sebesar 4.14% (0.90) dan RMSE sebesar 7.37 (0.80). Berikut adalah tabel lengkap dari hasil pengujiannya :

Tabel 4.6 Hasil Pengujian Model PSO-ELM

PSO-ELM		
Ukuran	**MAPE**	**RMSE**
Fold 1	3.21	6.48
Fold 2	4.19	7.59
Fold 3	5.01	8.03
Rata-rata	**4.14**	**7.37**
Std Dev	**0.90**	**0.80**

Hasil pemodelan menggunakan PSO-ELM lebih baik apabila dibandingkan dengan model sebelumnya (SVR, MLR, dan ELM). Untuk setiap *fold*-nya, nilai error dari model PSO-ELM lebih rendah dari model sebelumnya kecuali pada *fold* 3. Nilai rata-rata dan standar deviasi MAPE

maupun RMSE pada model PSO-ELM lebih kecil dibandingkan dengan model sebelumnya, hal ini menunjukan bahwa model PSO-ELM memiliki performa yang baik sekaligus memiliki stabilitas yang lebih baik dibandingkan dengan model-model sebelumnya.

4.6 Perbandingan Algoritma *Data Mining*

Tabel 4.7 Performa Setiap Algoritma *Data Mining*

Model	MAPE	RMSE
SVR	4.76	9.17
MLR	4.81	8.63
ELM	4.80	8.62
PSO-ELM	4.14	7.37

Menurut tabel 4.7 dapat dilihat bahwa ELM secara general memiliki performa yang lebih baik dibandingkan SVR dan MLR apabila ditinjau dari nilai RMSE-nya. Selain itu, penambahan PSO untuk mencari bobot awal dan bias yang optimal pada ELM juga dapat meningkatkan performa pada model ELM itu sendiri. Dibuktikan dengan improvisasi MAPE 4.80% menjadi 4.14% dan improvisasi RMSE 8.62 menjadi 7.37. Algoritma *data mining* terbaik pada penelitian ini adalah PSO-ELM yang memiliki nilai rata-rata MAPE 4.14% dan nilai rata-rata RMSE 7.37.

4.7 Tinjauan Hasil Tugas Akhir dibanding dengan Tugas Akhir Terdahulu

Penelitian terdahulu mengenai optimasi suhu dan durasi *roasting* biji kakao menggunakan RSM atau *Response Surface Methodology*. Penelitian tersebut hanya menghasilkan output berupa rekomendasi nilai suhu dan durasi *roasting* yang optimal secara eksak [6][7][8]. Pada penelitian ini, rekomendasi suhu dan durasi *roasting* ditentukan oleh input yang merupakan variabel-variabel yang merepresentasikan kondisi *roasting*. Rekomendasi tersebut didapat dari algoritma *data mining* yang telah diujikan menggunakan skema *3-fold*. Hasil pengujian tersebut cukup relevan dengan beberapa penelitian sebelumnya terutama yang berhubungan dengan metode. Dari sisi algoritma, terdapat 2 penelitian yang membandingkan algoritma ELM dengan algoritma lainnya yaitu SVM dan MLR. Kedua penelitian tersebut menunjukan bahwa ELM merupakan algoritma terbaik [31][32]. Hal ini sesuai dengan penelitian ini dimana ELM mampu memiliki performa yang paling baik di antara algoritma *data mining* lainnya dengan perbandingan yang cukup signifikan.

Selain itu, penambahan PSO untuk memilih bobot awal dan bias juga dapat meningkatan performa dari algoritma ELM, sebagaimana penelitian sebelumnya [27].

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

1. Algoritma data *mining* yang digunakan untuk merekomendasikan suhu dan durasi *roasting* pada proses *roasting* biji kakao dapat dirancang. Adapun algoritma *data mining* yang digunakan adalah SVM, MLR, ELM, dan PSO-ELM. Algoritma tersebut diujikan menggunakan skema *3-fold*.
2. SVM, MLR, ELM, dan PSO-ELM diujikan menggunakan skema *3-fold*. Berdasarkan MAPE dan RMSE, ELM dan PSO-ELM memiliki performa terbaik, diikuti SVM, kemudian MLR. Penambahan PSO untuk mencari bobot awal dan bias pada ELM juga dapat meningkatkan performa dari algoritma ELM.
3. Algoritma terbaik yang dapat digunakan untuk merekomendasikan suhu yang optimal berdasarkan durasi *roasting* tertentu pada proses *roasting* biji kakao adalah PSO-ELM yang memiliki nilai rata-rata MAPE 4.14% dan nilai rata-rata RMSE 7.37. Fitur atau variabel terpilih yang memiliki pengaruh terhadap suhu roasting dan digunakan pada algoritma *data mining* adalah nibs_capacity, moist_N, durasi_roasting, dan beans_source.

5.2 Saran

Pengumpulan data yang dilakukan oleh UGM CTLI antara data sensor dengan data produksi pada setiap proses *roasting* belum tersinkronisasi dengan baik, banyak data yang tidak tercatat dan hilang sehingga data yang hanya bisa digunakan pada penelitian ini sangat sedikit yaitu hanya 43 baris data. Berdasarkan hal tersebut, penulis menyarankan agar UGM CTLI mampu membuat infrastruktur data yang tersinkronisasi dengan baik, sehingga data-datanya dapat dimanfaatkan untuk penelitian lanjutan.

Penelitian lanjutan dapat dilakukan untuk melakukan uji algoritma dan beberapa perlakuan pada data seperti misalnya melakukan *oversampling* dan menambah metode optimalisasi *hyperparameter* atau bobot. Kemudian, penelitian lanjutan juga dapat dilakukan untuk melakukan eksplorasi terhadap metode seleksi fitur yang lain seperti *wrapped, embedded*, atau metode filter yang lainnya. Selain itu, penelitian lanjutan juga dapat dilakukan untuk melakukan eksplorasi terhadap algoritma yang belum diujikan pada penelitian ini terutama pengujian pada *tree-based algorithm*.

DAFTAR PUSTAKA

[1] V. A. Dihni. "5 Negara Penghasil Kakao Terbesar, Indonesia Urutan Berapa?," *Databoks*, 04 October 2021. [Online]. Available: https://databoks.katadata.co.id/datapublish/2021/10/04/5-negara-penghasil-kakao-terbesar-indonesia-urutan-berapa. [Accessed: 26 March 2023].

[2] A. Mahmudan. "Ekspor Kakao Indonesia Turun 2,92% pada 2021," *DataIndonesia.id*, 20 May 2022. [Online]. Available: https://dataindonesia.id/sektor-riil/detail/ekspor-kakao-indonesia-turun-292-pada-2021 . [Accessed: 26 March 2023].

[3] Direktorat Jendral Perkebunan, "Luas Areal Kakao Menurut Provinsi di Indonesia, 2017 - 2021," *Direktorat Jendral Perkebunan*, 2021. [Online]. Available: https://www.pertanian.go.id/home/index.php?show=repo&fileNum=224. [Accessed: 26 March 2023].

[4] Ditpui. "Proses Pengolahan Cokelat di Tingkat UGM Cocoa Teaching and Learning Industry," *Direktorat Pengembangan Bisnis dan Inkubasi UGM*, 31 October 2020. . [Online]. Available: https://ditpui.ugm.ac.id/proses-pengolahan-cokelat-di-tingkat-ugm-cocoa-teaching-and-learning-industry/ . [Accessed: 26 March 2023].

[5] S. Wijanarti, A. M. Rahmatika, and R. Hardiyanti, "Pengaruh lama penyangraian Manual Terhadap Karakteristik Kakao Bubuk," *Jurnal Nasional Teknologi Terapan (JNTT)*, vol. 2, no. 2, p. 212, 2019.

[6] Misnawi, S. Mulato, S. Widyotomo, A. Sewet, and Sugiyono, "Optimasi Suhu dan Lama Penyangraian Biji Kakao Menggunakan Penyangrai Skala Kecil Tipe Silinder," *Pelita Perkebunan 2005*, vol. 21, no. 3, p. 169, 2005.

[7] D. M. H. Farah, A. H. Zaibunnisa, J. Misnawi, and S. Zainal, "Optimization of cocoa beans roasting process using Response Surface Methodology based on concentration of pyrazine and acrylamide," *International Food Research Journal*, vol. 19, pp. 1355-1359, 2012.

[8] I. S. Rocha, L. R. Santana, S. E. Soares, And E. Da Bispo, "Effect of the roasting temperature and time of cocoa beans on the sensory characteristics and acceptability of chocolate," *Food Science and Technology*, vol. 37, no. 4, pp. 522–530, 2017.

[9] Y. Yang, A. G. Darwish, I. El-Sharkawy, Q. Zhu, S. Sun, and J. Tan, "Rapid determination of the roasting degree of cocoa beans by extreme learning machine (elm)-based imaging analysis," *Journal of Agriculture and Food Research*, vol. 10, p. 100437, 2022.

[10] J. Kubala, "What are cacao nibs? nutrition, benefits, and culinary uses," *Healthline*, 28 March 2019. [Online]. Available: https://www.healthline.com/nutrition/cacao-nibs. [Accessed: 26 March 2023].

[11] C. Stedman and A. Hughes, "What is data mining?," *Business Analytics*, 07 September 2021. [Online]. Available: https://www.techtarget.com/searchbusinessanalytics/definition/data-mining#:~:text=Data%20mining%20is%20the%20process,make%20more%2Dinformed%20business%20decisions. [Accessed: 27 March 2023].

[12] F. F. Firdaus, H. A. Nugroho, and I. Soesanti, "A review of feature selection and classification approaches for heart disease prediction," *IJITEE (International Journal of Information Technology and Electrical Engineering)*, vol. 4, no. 3, p. 75, 2021.

[13] N. Sánchez-Maroño, A. Alonso-Betanzos, and M. Tombilla-Sanromán, "Filter methods for feature selection – A comparative study," *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, pp. 178–187, 2007.

[14] J. Brownlee, "How to choose a feature selection method for machine learning," *MachineLearningMastery.com*, 20 August 2020. [Online]. Available: https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/. [Accessed: 27 March 2023].

[15] A. M. Telussa, E. R. Persulessy, and Z. A. Leleury, "Penerapan Analisis Korelasi parsial Untuk Menentukan Hubungan Pelaksanaan FUNGSI Manajemen Kepegawaian Dengan efektivitas Kerja Pegawai," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 7, no. 1, pp. 15–18, 2013.

[16] A. A. Mattjik and M. Sumertajaya, *Perancangan Percobaan Dengan aplikasi sas Dan Minitab*. Bogor: PT Penerbit IPB Press, 2013.

[17] Sugiyono, *Metode Penelitian Pendidikan: (Pendekatan Kuantitatif, Kualitatif Dan R R & D)*. Bandung: Alfabeta, 2008.

[18] M. S. Pathan, A. Nag, M. M. Pathan, and S. Dev, "Analyzing the impact of feature selection on the accuracy of heart disease prediction," *Healthcare Analytics*, vol. 2, p. 100060, 2022.

[19] K. Potdar, T. S., and C. D., "A comparative study of categorical variable encoding techniques for neural network classifiers," *International Journal of Computer Applications*, vol. 175, no. 4, pp. 7–9, 2017.

[20] O. A. Akanbi, I. S. Amiri, and E. Fazeldehkordi, *A machine learning approach to phishing detection and Defense*. Amsterdam: Elsevier, 2015.

[21] A. Shirzad, M. Tabesh, and R. Farmani, "Performance Comparison between Support Vector Regression and Artificial Neural Network for Prediction of Oil Palm Production," *Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)*, vol. 9, no. 1, pp. 1-8, 2016.

[22] G. K. Uyanık and N. Güler, "A study on multiple linear regression analysis," *Procedia - Social and Behavioral Sciences*, vol. 106, pp. 234–240, 2013.

[23] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[24] D. M. Belete and M. D. Huchaiah, "Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results," *International Journal of Computers and Applications*, vol. 44, no. 9, pp. 875–886, 2021.

[25] Sulistiana and M. A. Muslim, "Support Vector Machine (SVM) optimization using grid search and UNIGRAM to improve e-commerce review accuracy," *Journal of Soft Computing Exploration*, vol. 1, no. 1, 2020.

[26] P. Godbole and Dr. M. Pathak, "Particle Swarm Optimization (PSO) Model and Its Application in ANN Controller", I*nternational Journal for Modern Trends in Science and Technology*, vol. 8, no.1, pp. 153-157, 2022.

[27] M. R. Kaloop, D. Kumar, P. Samui, A. R. Gabr, J. W. Hu, X. Jin, and B. Roy, "Particle swarm optimization algorithm-extreme learning machine (PSO-ELM) model for predicting resilient modulus of stabilized aggregate bases," *Applied Sciences*, vol. 9, no. 16, p. 3221, 2019.

[28] S. Kim and H. Kim, "A new metric of absolute percentage error for intermittent demand forecasts," *International Journal of Forecasting*, vol. 32, no. 3, pp. 669–679, 2016.

[29] D. Christie and S. P. Neill, "Measuring and observing the Ocean Renewable Energy Resource," *Comprehensive Renewable Energy*, pp. 149–175, 2022.

[30] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation," *Encyclopedia of Database Systems*, pp. 532–538, 2009.

[31] S. K. Rath, M. Sahu, S. P. Das, S. K. Bisoy, and M. Sain, "A comparative analysis of SVM and ELM Classification on Software Reliability Prediction Model," *Electronics*, vol. 11, no. 17, p. 2707, 2022.

[32] Y. Xu, S. Afshar, R. K. Singh, R. Wang, A. van Schaik, and T. J. Hamilton, “A binaural sound localization system using deep convolutional neural networks,” *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019.

LAMPIRAN

L.1 Source Code

```
# -*- coding: utf-8 -*-
"""Skirpsi Rapih.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1fXTNe1r2NXuxe09dtN3Q7mMurm-Ri-kQ
"""

#Import Library
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

"""##1. Praproses Data"""

#Mengunduh Data
!wget --no-check-certificate
'https://docs.google.com/uc?export=download&id=1OIskHy2jSeCGTcuzpmznXO8qPcvdVPsH' -O DataProd.csv

#Membaca Data
prod = pd.read_csv("/content/DataProd.csv")

#NA to Natural
prod = prod.fillna("Natural")

#Mengubah Tipe Data Jam dari string menjadi Waktu
prod["Jam_Mulai"] = pd.to_datetime(prod["Jam_Mulai"].astype(str), format='%H.%M')
prod["Jam_Turun"] = pd.to_datetime(prod["Jam_Turun"].astype(str), format='%H.%M')
prod["Jam_Akhir"] = pd.to_datetime(prod["Jam_Akhir"].astype(str), format='%H.%M')

#Membuat Variabel Lama Roasting dan Lama Mesin Jalan
prod["durasi_roasting"] = ((prod["Jam_Akhir"]-prod["Jam_Turun"]).dt.total_seconds().astype(int))/60

prod.rename(columns = {'T_Puncak2':'suhu','Moist_0':'moist_0','Moist_N':'moist_N'}, inplace = True)
```

```
# Agar tidak ada yang outlier
prod=prod[prod["durasi_roasting"]<275]

df_num=
prod.drop(columns=['beans_source','product_type','is_alkalized','batch_id',"batch_id","date","Jam_Mulai","Jam_Turun"
,"Jam_Akhir","T_Puncak1"])
df_cat= prod[['beans_source','product_type','is_alkalized','suhu','durasi_roasting']]

# One-hot Encoding pada Variabel Kategorik
cat_features = ['beans_source', 'product_type']
for feature in cat_features:
    a = pd.get_dummies(prod[feature], prefix = feature).astype(int)
    frames = [prod, a]
    prod = pd.concat(frames, axis = 1)
prod.drop(cat_features, axis = 1, inplace=True)

#Menghapus Beberapa Kolom yang Tidak Berguna
deleted = ["batch_id","date","Jam_Mulai","Jam_Turun","Jam_Akhir","T_Puncak1"]
df = prod.drop(columns=deleted)

df.rename(columns = {'T_Puncak2':'suhu','Moist_0':'moist_0','Moist_N':'moist_N'}, inplace = True)
df_num.rename(columns = {'T_Puncak2':'suhu','Moist_0':'moist_0','Moist_N':'moist_N'}, inplace = True)

"""## 2. EDA"""

def plot_line(x_,y_) :
  xx=df[x_]
  yy=df[y_]

  m, b = np.polyfit(xx, yy, 1)
  df.plot.scatter(x=x_, y=y_)
  plt.plot(xx, m*xx + b, color ="red")
  plt.grid()
  plt.title(str(y_)+" terhadap " +str(x_))

x_num = ['nibs_capacity', 'solution_load', 'pH_0', 'pH_N', 'delta_pH', 'moist_0',
       'moist_N', 'delta_moist']

for x in x_num :
```

```
  plot_line(x,"suhu")

def mean_plot (df,x,y,r) :
  x1 = df.groupby(x).median().reset_index()
  plt.grid()
  plt.bar(range(len(x1)), x1[y])
  plt.xticks(range(len(x1)), x1[x],rotation=r)
  plt.xlabel(x)
  plt.ylabel('Median of '+y)
  plt.title("Median "+y+" terhadap "+x)

mean_plot(df,"is_alkalized","suhu",0)

mean_plot(df_cat,"beans_source","suhu",45)

mean_plot(df_cat,"product_type","suhu",45)

"""## 3. Uji korelasi dan seleksi fitur"""

mask = np.zeros_like(df_num.corr(), dtype=np.bool)
## in order to reverse the bar replace "RdBu" with "RdBu_r"
plt.subplots(figsize = (15,12))
sns.heatmap(df_num.corr(), annot=True,mask = False,cmap = 'OrRd', linewidths=.2, linecolor='black',fmt='.1g',center
= 0,square=True)

plt.title("Correlations", y = 1.03,fontsize = 20, fontweight = 'bold', pad = 40);

"""### 3.1 Uji Korelasi Variabel Numerik"""

from scipy import stats

X = df.drop(columns=['suhu','durasi_roasting'])
y = df[["suhu","durasi_roasting"]]

x_num = ['nibs_capacity', 'solution_load', 'pH_0', 'pH_N', 'delta_pH', 'moist_0',
       'moist_N', 'delta_moist']

corr_df = pd.DataFrame()
corr_df ["Variabel"] = x_num
```

```
r=[]
pvalue = []
for x in x_num :
  a,b = stats.pearsonr(X[x],y["suhu"])
  r.append(a)
  pvalue.append(b)

corr_df["r_suhu"] = r
corr_df["pvalue_suhu"] = pvalue

corr_df

corr_df.to_excel('/content/drive/MyDrive/Data Skripsi/table result/corr_test.xlsx',index=False)

"""### 3.2 Uji F Variabel Kategorik"""

df_cat = df_cat[["beans_source","product_type","is_alkalized"]]
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
for x in ["beans_source","product_type","is_alkalized"] :
  df_cat[x] = le.fit_transform(df_cat[x])

#ANOVA test
from sklearn.feature_selection import f_regression

f_score, p_value = f_regression(df_cat,y["suhu"])
f_test = pd.DataFrame()
f_test["Variabel"] = df_cat.columns
f_test["fscore_suhu"] = f_score
f_test["pvalue_suhu"] = p_value
#f_test = f_test.sort_values(by=["f_score"],ascending=False)
f_test

# # # Hasil nilai F-Score
# # sns.barplot(x ='f_score', y='Variabel',data = f_test)

f_test.to_excel('/content/drive/MyDrive/Data Skripsi/table result/f_test.xlsx',index=False)

"""## 4. Modelling"""
```

```
desel_num = np.array(corr_df[corr_df["pvalue_suhu"]>0.01]["Variabel"])
desel_cat =['product_type_RB',
      'product_type_Natural',
       "is_alkalized"]

desel = np.concatenate((desel_num,desel_cat))

df=df.drop(columns=desel)
df=df.sample(len(df),random_state=1234)

X = df.drop(columns=['suhu'])
y = df[["suhu"]]

X.columns

# Normaliasi
from sklearn.preprocessing import MinMaxScaler

X_norm = MinMaxScaler().fit_transform(X)
X[X.columns] = X_norm

from sklearn.model_selection import train_test_split

#Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3333, random_state=1234)

#Data Preprocessing and Algebra
import numpy as np
import pandas as pd

#Data Visualization
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

#Machine Learning and Evaluation Model
from sklearn.model_selection import train_test_split, cross_val_score, KFold, cross_validate, cross_val_predict
!pip install xgboost
from xgboost import XGBRegressor
from xgboost import XGBRFRegressor
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score, make_scorer,
mean_absolute_percentage_error

from sklearn.neural_network import MLPRegressor
from sklearn.svm import SVR

"""###4.1 SVR

####4.1.1 Tunning
"""

from sklearn.model_selection import TimeSeriesSplit

param_grid = {'C': [0.1, 0.001, 1, 10, 12, 14, 16, 18, 20, 22],
          'gamma': [0.001, 0.01, 0.1, 1, 2, 5],
          'epsilon': [0.001, 0.01, 0.1, 1, 2, 4],
          'kernel': ("rbf", "poly", "linear")}
search = GridSearchCV(SVR(), param_grid,
                      cv = 3, n_jobs = -1, verbose = 1000, scoring="neg_mean_squared_error")
search.fit(X, y)
print(search.best_params_)

"""#### 4.1.2 Cross-Validation"""

KFold_ = KFold(n_splits=3,shuffle=True, random_state=1234)
kfold = KFold_.split(X, y)
rmse_data = []
mape_data = []

for k, (train, test) in enumerate(kfold):
  X_train = X.iloc[train, :]
  y_train = y.iloc[train]

  X_test = X.iloc[test, :]
  y_test = y.iloc[test]

  param = {'C': 22, 'epsilon': 1, 'gamma': 1, 'kernel': 'rbf'}
  model = SVR(**param)
  model.fit(X_train,y_train)
```

```
 y_pred = model.predict(X_test)


 # Aktual vs Prediksi non FS
 fig, ax1 = plt.subplots(figsize=(8,6))
 plt.scatter(y_test,y_pred,color='blue')
 plt.plot(y_test,y_test,color='red')
 plt.title('SVR Regression Result for Suhu')
 plt.xlabel('Actual Values')
 plt.ylabel('Predict Values')
 plt.show()
 plt.close()


 #root_mean_squared_error (RMSE)
 rmse_suhu = np.sqrt(mean_squared_error(y_test, y_pred))
 rmse_data.append(rmse_suhu)
 #MAPE
 mape_suhu = mean_absolute_percentage_error(y_test, y_pred)
 mape_data.append(mape_suhu*100)

 print("MAPE Suhu : " + str(mape_suhu*100))
 print("RMSE Suhu : " + str(rmse_suhu))

print("")
print("-----")
print("3-Fold Cross Vaidation")
print("MAPE Suhu : "+str(np.array(mape_data).mean()))
print("RMSE Suhu : "+str(np.array(rmse_data).mean()))
# print("MAPE Durasi : "+str(np.array(mape_data_).mean()))
# print("RMSE Durasi : "+str(np.array(rmse_data_).mean()))

svm_mape = mape_data
svm_rmse = rmse_data

"""### 4.2 Linear Regression"""

from sklearn.linear_model import LinearRegression

model = LinearRegression()
```

```
"""#### 4.2.2 Cross-Validation"""

KFold_ = KFold(n_splits=3,shuffle=True, random_state=1234)
kfold = KFold_.split(X, y)
rmse_data = []
mape_data = []
params = []

for k, (train, test) in enumerate(kfold):
  X_train = X.iloc[train, :]
  y_train = y.iloc[train]

  X_test = X.iloc[test, :]
  y_test = y.iloc[test]

  #mlp = MLPRegressor(solver='lbfgs',hidden_layer_sizes=(50, 2),max_iter=10000)
  model = model.fit(X_train,y_train)
  y_pred = model.predict(X_test)


  # Aktual vs Prediksi non FS
  fig, ax1 = plt.subplots(figsize=(8,6))
  plt.scatter(y_test,y_pred,color='blue')
  plt.plot(y_test,y_test,color='red')
  plt.title('Linear Regression Result for Suhu')
  plt.xlabel('Actual Values')
  plt.ylabel('Predict Values')
  plt.show()
  plt.close()


  #root_mean_squared_error (RMSE)
  rmse_suhu = np.sqrt(mean_squared_error(y_test, y_pred))
  rmse_data.append(rmse_suhu)
  #MAPE
  mape_suhu = mean_absolute_percentage_error(y_test, y_pred)
  mape_data.append(mape_suhu*100)

  print("MAPE Suhu : " + str(mape_suhu*100))
```

```
  print("RMSE Suhu : " + str(rmse_suhu))

print("")
print("-----")
print("3-Fold Cross Vaidation")
print("MAPE Suhu : "+str(np.array(mape_data).mean()))
print("RMSE Suhu : "+str(np.array(rmse_data).mean()))
# print("Best Param : "+str(params[np.argmin(mape_data)]))
# mlp_params = params[np.argmin(mape_data)]
# print("MAPE Durasi : "+str(np.array(mape_data_).mean()))
# print("RMSE Durasi : "+str(np.array(rmse_data_).mean()))

lr_mape = mape_data
lr_rmse = rmse_data

"""### 4.3 ELM"""

import scipy

def elm_fit(X_train,y_train,n) :
  #1. Definisiin banyaknya variabel (m) dan jumlah hidden node (n)
  m = X_train.shape[1]
  W_train = abs(np.random.normal(size=[m,n])) #InputWeightLatih

  #2. Operasi dot X_train dengan W_train
  h_train = np.dot(X_train, W_train)

  #3. Masukin h_train ke relu
  H_train = np.maximum(h_train, 0, h_train)

  #4. Mencari Beta Latih
  H_train_inv = scipy.linalg.pinv(H_train)
  B_train = np.dot(H_train_inv,y_train)
  #np.save("B_train_"+str(N)+".npy", B_train)
  return W_train, B_train

def elm_predict(X_test,W_train,B_train) :
  #5. Uji
  h_test = np.dot(X_test,W_train) #hasil dot product dari X_test dan W_train
  H_test = np.maximum(h_test, 0, h_test) #hasil relu
```

```
 y_pred = np.dot(H_test, B_train) #hasil prediksi
 return y_pred

"""#### 4.3.1 Tunning"""

from sklearn.model_selection import train_test_split

#Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3333, random_state=1234)

node = [5,10,20,30,40,50,60,70,80,90,100,150,200,300,400,500,600,700,1000]
mape = []
W_train__ = []
for x in node :
 a,b = elm_fit(X_train,y_train,x)
 y_pred = elm_predict(X_test,a,b)
 W_train__.append(a)
 mape.append(mean_absolute_percentage_error(y_test,y_pred))

print("Best Node is "+str(node[np.argmin(mape)]))
print("With score mape = "+str(min(mape)*100))
best_w = W_train__[np.argmin(mape)]

plt.title("MAPE terhadap jumlah Node")
plt.xlabel("Jumlah Node")
plt.ylabel("MAPE")
plt.grid()

plt.plot(node,mape)

"""#### 4.3.2 Cross-Validation"""

KFold_ = KFold(n_splits=3,shuffle=True, random_state=1234)
kfold = KFold_.split(X, y)
rmse_data = []
mape_data = []
params = []

for k, (train, test) in enumerate(kfold):
 X_train = X.iloc[train, :]
```

```
y_train = y.iloc[train]

X_test = X.iloc[test, :]
y_test = y.iloc[test]

#1. Definisiin banyaknya variabel (m) dan jumlah hidden node (n)
m = X_train.shape[1]
W_train = best_w
#2. Operasi dot X_train dengan W_train
h_train = np.dot(X_train, W_train)

#3. Masukin h_train ke relu
H_train = np.maximum(h_train, 0, h_train)

#4. Mencari Beta Latih
H_train_inv = scipy.linalg.pinv(H_train)
B_train = np.dot(H_train_inv,y_train)

h_test = np.dot(X_test,W_train) #hasil dot product dari X_test dan W_train
H_test = np.maximum(h_test, 0, h_test) #hasil relu
y_pred = np.dot(H_test, B_train) #hasil prediksi

mape.append(mean_absolute_percentage_error(y_test,y_pred))


# Aktual vs Prediksi non FS
fig, ax1 = plt.subplots(figsize=(8,6))
plt.scatter(y_test,y_pred,color='blue')
plt.plot(y_test,y_test,color='red')
plt.title('ELM Regression Result for Suhu')
plt.xlabel('Actual Values')
plt.ylabel('Predict Values')
plt.show()
plt.close()


#root_mean_squared_error (RMSE)
rmse_suhu = np.sqrt(mean_squared_error(y_test, y_pred))
rmse_data.append(rmse_suhu)
#MAPE
```

```
  mape_suhu = mean_absolute_percentage_error(y_test, y_pred)
  mape_data.append(mape_suhu*100)

  print("MAPE Suhu : " + str(mape_suhu*100))
  print("RMSE Suhu : " + str(rmse_suhu))

print("")
print("-----")
print("3-Fold Cross Vaidation")
print("MAPE Suhu : "+str(np.array(mape_data).mean()))
print("RMSE Suhu : "+str(np.array(rmse_data).mean()))
# print("Best Param : "+str(params[np.argmin(mape_data)]))
# mlp_params = params[np.argmin(mape_data)]
# print("MAPE Durasi : "+str(np.array(mape_data_).mean()))
# print("RMSE Durasi : "+str(np.array(rmse_data_).mean()))

elm_mape = mape_data
elm_rmse = rmse_data

np.save("/content/drive/MyDrive/Data Skripsi/W_best_ELM.npy", best_w)

"""### 4.4 PSO-ELM"""

import scipy

def elm_fit(X_train,y_train,W_train, n) :
  #1. Definisiin banyaknya variabel (m) dan jumlah hidden node (n)
  m = X_train.shape[1]
  W_train = abs(np.random.normal(size=[m,n])) #InputWeightLatih

  #2. Operasi dot X_train dengan W_train
  h_train = np.dot(X_train, W_train)+bias

  #3. Masukin h_train ke relu
  H_train = np.maximum(h_train, 0, h_train)

  #4. Mencari Beta Latih
  H_train_inv = scipy.linalg.pinv(H_train)
  B_train = np.dot(H_train_inv,y_train)
  #np.save("B_train_"+str(N)+".npy", B_train)
```

```
  return W_train, B_train

def elm_predict(X_test,W_train,B_train) :
  #5. Uji
  h_test = np.dot(X_test,W_train)+bias #hasil dot product dari X_test dan W_train
  H_test = np.maximum(h_test, 0, h_test) #hasil relu
  y_pred = np.dot(H_test, B_train) #hasil prediksi
  return y_pred

"""#### 4.4.1 Tunning"""

# Membuat Fungsi Objektif
def obj(bias,W_train) :
  KFold_ = KFold(n_splits=3,shuffle=True, random_state=1234)
  kfold = KFold_.split(X, y)
  rmse_data = []
  mape_data = []

  for k, (train, test) in enumerate(kfold):
    X_train = X.iloc[train, :]
    y_train = y.iloc[train]

    X_test = X.iloc[test, :]
    y_test = y.iloc[test]

    #1. Definisiin banyaknya variabel (m) dan jumlah hidden node (n)
    m = X_train.shape[1]
    #W_train = abs(np.random.normal(size=[m,n])) #InputWeightLatih

    #2. Operasi dot X_train dengan W_train
    h_train = np.dot(X_train, W_train)+bias

    #3. Masukin h_train ke relu
    H_train = np.maximum(h_train, 0, h_train)

    #4. Mencari Beta Latih
    H_train_inv = scipy.linalg.pinv(H_train)
    B_train = np.dot(H_train_inv,y_train)
    #np.save("B_train_"+str(N)+".npy", B_train)
```

```
    h_test = np.dot(X_test,W_train)+bias #hasil dot product dari X_test dan W_train
    H_test = np.maximum(h_test, 0, h_test) #hasil relu
    y_pred = np.dot(H_test, B_train) #hasil prediksi

    #Scoring
    #root_mean_squared_error (RMSE)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    rmse_data.append(rmse)

    #MAPE
    mape = mean_absolute_percentage_error(y_test,y_pred)
    mape_data.append(mape*100)

  return np.array(rmse_data).mean()

import random

def update_velocity(particle, velocity, pbest, gbest, w_min=0.5, max=1.0, c=0.1):
  # Initialise new velocity array
  num_particle = len(particle)
  new_velocity = np.array([0.0 for i in range(num_particle)])
  # Randomly generate r1, r2 and inertia weight from normal distribution
  r1 = random.uniform(0,max)
  r2 = random.uniform(0,max)
  w = random.uniform(w_min,max)
  c1 = c
  c2 = c
  # Calculate new velocity
  for i in range(num_particle):
    new_velocity[i] = w*velocity[i] + c1*r1*(pbest[i]-particle[i])+c2*r2*(gbest[i]-particle[i])
  return new_velocity

def update_position(particle, velocity):
  # Move particles by adding velocity
  new_particle = particle + velocity
  return new_particle

def pso(population, dimension, position_min, position_max, generation, fitness_criterion):
  # Initialisation
  # Population
```

```
particles = [[random.uniform(position_min, position_max) for j in range(dimension)] for i in range(population)]
# Particle's best position
pbest_position = particles
# Fitness

# bias = x0[i][0]
# w = x0[i][1:m]
# W_train = np.resize(x0[i][1:m],(X.shape[1],N))

pbest_fitness = []
for i in range(0,population):
  bias = particles[i][0]
  w = particles[i][1:m]
  W_train = np.resize(w,(X.shape[1],N))
  y_ = obj(bias,W_train)
  pbest_fitness.append(y_)
# [fitness_function(p[0],np.resize(p[1:m],(X.shape[1],N))) for p in particles]
# print(pbest_fitness)
# Index of the best particle
gbest_index = np.argmin(pbest_fitness)
# Global best particle position
gbest_position = pbest_position[gbest_index]
# Velocity (starting from 0 speed)
velocity = [[0.0 for j in range(dimension)] for i in range(population)]

# Loop for the number of generation
for t in range(generation):
  # Stop if the average fitness value reached a predefined success criterion
  if np.average(pbest_fitness) <= fitness_criterion:
    break
  else:
    for n in range(population):
      # Update the velocity of each particle
      velocity[n] = update_velocity(particles[n], velocity[n], pbest_position[n], gbest_position)
      # Move the particles to new position
      particles[n] = update_position(particles[n], velocity[n])
  # Calculate the fitness value
  pbest_fitness = []
  for i in range(0,population):
    bias = particles[i][0]
```

```
      w = particles[i][1:m]
      W_train = np.resize(w,(X.shape[1],N))
      y_ = obj(bias,W_train)
      pbest_fitness.append(y_)
    print(min(pbest_fitness))
    # print(pbest_fitness)
    # Find the index of the best particle
    gbest_index = np.argmin(pbest_fitness)
    # Update the position of the best particle
    gbest_position = pbest_position[gbest_index]

  # Print the results
  print('Global Best Position: ', gbest_position)
  print('Best Fitness Value: ', min(pbest_fitness))
  print('Average Particle Best Fitness Value: ', np.average(pbest_fitness))
  print('Number of Generation: ', t)
  return(gbest_position)

#Step 1 Initialization

import random

N = 100 #Jumlah Node

m = X.shape[1]*N+1
n = 10
Wmax = 0.9
Wmin = 0.4
c1 = 2
c2 = 2
MaxIteration = 50

x0 = np.zeros((n, m))


LB_w= np.zeros((m-1)) #Nilai Awal
UB_w = np.ones((m-1)) #Nilai Akhir

LB=np.append(np.array([0]),LB_w)
UB=np.append(np.array([10]),UB_w)
```

```
for i in range (0,n):
  for j in range(0,m):
    x0[i][j] = LB[j]+random.random()*(UB[j]-LB[j])

v = 0.1*x0
x0 = x0 + v

#Step 2 fitting for t initialization
result_0 = []
for i in range(0,n):
  bias = x0[i][0]
  w = x0[i][1:m]
  W_train = np.resize(w,(X.shape[1],N))
  y_ = obj(bias,W_train)
  result_0.append(y_)
index_min_0 = np.argmin(result_0)

result_before = result_0
x_before = x0
gbest_before = x0[index_min_0]
xbest = np.zeros((n, m))
a=1
p=0

iter=100

result_gbest=[]
gbest_ = []
iterr = []
best_score = []
for p in range(0,iter) :
  for i in range (0,n):
    for j in range(0,m):
      if p == 0 :
        v[i][j] = Wmax*v[i][j]+c1*random.random()*(x0[i][j]-x0[i][j]) + c2*random.random()*(x0[index_min_0][j]-x0[i][j])
      else :
        v[i][j] = Wmax*v[i][j]+c1*random.random()*(xbest[i][j]-x_[i][j]) + c2*random.random()*(gbest[j]-x_[i][j])

  x_=x_before+v
```

```
result = []
for i in range(0,n):
  bias = x_[i][0]
  w = x_[i][1:m]
  W_train = np.resize(w,(X.shape[1],N))
  y_ = obj(bias,W_train)
  result.append(y_)

# print("Before " + str(min(result_before)))
# print("After " + str(min(result)))
# #print(np.argmin(result))

#xbest
for i in range(0,n):
  if result[i]<result_before[i]:
    xbest[i] = x_[i]
    #print(str(i)+" update")
  else:
    xbest[i] = x_before[i]
    #print(str(i)+" not-update")

# print(min(result))
# print(np.argmin(result))

# result_gbest = []
# for i in range(0,n):
#   bias = gbest_before[i][0]
#   w = xbest[i][1:m]
#   W_train = np.resize(w,(X_train.shape[1],N))
#   y_ = obj(bias,W_train)
#   result_gbest.append(y_)

bias = gbest_before[0]
w = gbest_before[1:m]
W_train = np.resize(w,(X.shape[1],N))
result_gbest = obj(bias,W_train)

#gbest
for i in range(0,n):
```

```
    if result[i]<result_gbest :
      gbest = x_[np.argmin(result)]
    else :
      gbest = gbest_before

  # if min(result_xbest)<min(result_before) :
  #   gbest = xbest[np.argmin(result_xbest)]
  #   # print("Update")
  #   # print(gbest[0])
  # else:
  #   gbest = gbest_before
  #   # print("Non-Update")
  #   # print(gbest[0])

  #gbest_.append(gbest)

  bias = gbest[0]
  w = gbest[1:m]
  W_train = np.resize(w,(X.shape[1],N))
  error = obj(bias,W_train)
  #result_gbest.append(error)

  # print(abs(error))
  # if abs(error) <= abs(0.2):
  #   break

  result_before = result
  x_before = x_
  gbest_before = gbest
  p=p+1
  print("Error iterasi ke-"+str(p)+" "+str(error))
  iterr.append(p)
  best_score.append(error)

bias_ = gbest[0]
w = gbest[1:m]
W_train_ = np.resize(w,(X.shape[1],N))

np.save("/content/drive/MyDrive/Data Skripsi/GBest_PSO-ELM.npy", gbest)
```

```
plt.title("MAPE terhadap iterasi PSO")
plt.xlabel("Iterasi ke-")
plt.ylabel("Best MAPE")
plt.grid()

plt.plot(iterr,best_score)

"""#### 4.4.2 Cross-Validation"""

KFold_ = KFold(n_splits=3,shuffle=True, random_state=1234)
kfold = KFold_.split(X, y)
rmse_data = []
mape_data = []
params = []

for k, (train, test) in enumerate(kfold):
  X_train = X.iloc[train, :]
  y_train = y.iloc[train]

  X_test = X.iloc[test, :]
  y_test = y.iloc[test]

  #1. Definisiin banyaknya variabel (m) dan jumlah hidden node (n)
  m = X_train.shape[1]
  n = 50
  W_train = W_train_

  #2. Operasi dot X_train dengan W_train
  h_train = np.dot(X_train, W_train)+bias_

  #3. Masukin h_train ke relu
  H_train = np.maximum(h_train, 0, h_train)

  #4. Mencari Beta Latih
  H_train_inv = scipy.linalg.pinv(H_train)
  B_train = np.dot(H_train_inv,y_train)
  #np.save("B_train_"+str(N)+".npy", B_train)

  h_test = np.dot(X_test,W_train)+bias_ #hasil dot product dari X_test dan W_train
  H_test = np.maximum(h_test, 0, h_test) #hasil relu
```

```
 y_pred = np.dot(H_test, B_train) #hasil prediksi


 # Aktual vs Prediksi non FS
 fig, ax1 = plt.subplots(figsize=(8,6))
 plt.scatter(y_test,y_pred,color='blue')
 plt.plot(y_test,y_test,color='red')
 plt.title('ELM Regression Result for Suhu')
 plt.xlabel('Actual Values')
 plt.ylabel('Predict Values')
 plt.show()
 plt.close()


 #root_mean_squared_error (RMSE)
 rmse_suhu = np.sqrt(mean_squared_error(y_test, y_pred))
 rmse_data.append(rmse_suhu)
 #MAPE
 mape_suhu = mean_absolute_percentage_error(y_test, y_pred)
 mape_data.append(mape_suhu*100)

 print("MAPE Suhu : " + str(mape_suhu*100))
 print("RMSE Suhu : " + str(rmse_suhu))

print("")
print("-----")
print("3-Fold Cross Vaidation")
print("MAPE Suhu : "+str(np.array(mape_data).mean()))
print("RMSE Suhu : "+str(np.array(rmse_data).mean()))
# print("Best Param : "+str(params[np.argmin(mape_data)]))
# mlp_params = params[np.argmin(mape_data)]
# print("MAPE Durasi : "+str(np.array(mape_data_).mean()))
# print("RMSE Durasi : "+str(np.array(rmse_data_).mean()))

pso_elm_mape = mape_data
pso_elm_rmse = rmse_data

"""## 5. Summary"""

def mean(lst):
```

```
    return sum(lst) / len(lst)

summary = pd.DataFrame()

summary["Model"] = ["SVR","Linear Regression","ELM","PSO-ELM"]
summary["MAPE"] = [mean(svm_mape),mean(lr_mape),mean(elm_mape),mean(pso_elm_mape)]
summary["RMSE"] = [mean(svm_rmse),mean(lr_rmse),mean(elm_rmse),mean(pso_elm_rmse)]

summary

summary.to_excel('/content/drive/MyDrive/Data Skripsi/table result/summary.xlsx',index=False)

score_mape = pd.DataFrame()
score_mape["Fold"] = np.arange(0,3)+1
score_mape["SVR"] = svm_mape
score_mape["Linear Regression"] = lr_mape
score_mape["ELM"] = elm_mape
score_mape["PSO-ELM"] = pso_elm_mape

plt.plot(score_mape["Fold"],score_mape["SVR"])
plt.plot(score_mape["Fold"],score_mape["Linear Regression"])
plt.plot(score_mape["Fold"],score_mape["ELM"])
plt.plot(score_mape["Fold"],score_mape["PSO-ELM"])
plt.legend(["SVR","LR","ELM","PSO-ELM"])
plt.grid()
plt.xlabel("Fold")
plt.xticks([1,2,3])
plt.ylabel("MAPE")
plt.title("MAPE untuk Setiap Model per Fold")

score_mape = pd.DataFrame()
score_mape["Fold"] = np.arange(0,3)+1
score_mape["SVR"] = svm_rmse
score_mape["Linear Regression"] = lr_rmse
score_mape["ELM"] = elm_rmse
score_mape["PSO-ELM"] = pso_elm_rmse

plt.plot(score_mape["Fold"],score_mape["SVR"])
plt.plot(score_mape["Fold"],score_mape["Linear Regression"])
plt.plot(score_mape["Fold"],score_mape["ELM"])
```

```
plt.plot(score_mape["Fold"],score_mape["PSO-ELM"])
plt.legend(["SVR","LR","ELM","PSO-ELM"])
plt.grid()
plt.xlabel("Fold")
plt.xticks([1,2,3])
plt.ylabel("RMSE")
plt.title("RMSE untuk Setiap Model per Fold")
```

L.2 Tautan

1. Dataset: https://drive.google.com/drive/u/0/folders/1NVQooOz2W1zGE-bmzww9dXK-HJsRtVsf
2. Source Code: https://colab.research.google.com/drive/1Oljm5rN3-Q2A4VvbyHUh9drSfp025zzj?usp=sharing
3. Array W Best ELM: https://drive.google.com/file/d/15ElfM1nGAQapfIvWtlLnycC78abqg3EB/view?usp=share_link
4. Array G Best PSO-ELM: https://drive.google.com/file/d/1-1MLnW92zFEJV7i7lOOSEdBcU3xuEA7x/view?usp=share_link