KLASIFIKASI KONDISI TANAH BERDASARKAN REKOMENDASI TANAMAN PERTANIAN DAN PERKEBUNAN MELALUI PENGGUNAAN JARINGAN SYARAF TIRUAN KLASIFIKASI MULTINOMIAL

SKRIPSI

Oleh:

Ivan Budianto

NIM 192410103035

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS JEMBER

2022

KLASIFIKASI KONDISI TANAH BERDASARKAN REKOMENDASI TANAMAN PERTANIAN DAN PERKEBUNAN MELALUI PENGGUNAAN JARINGAN SYARAF TIRUAN KLASIFIKASI MULTINOMIAL

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk
menyelesaikan Program Studi Informatika (S1)
dan mencapai gelar Sarjana Komputer

Oleh:

Ivan Budianto

NIM 192410103035

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS JEMBER

2022

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Papa dan Mama yang selalu mendukung saya untuk menyelesaikan skripsi ini, dan mendukung segala keputusan yang saya buat,
2. Kedua kakak saya yang selalu membantu dan mengganggu saya dalam mengerjakan skripsi,
3. Jessica Maya Berlianasari yang selalu mendukung dan memotivasi saya dalam penyusunan ide hingga revisi dari naskah skripsi ini,
4. Keluarga Laboratorium Pemrograman Universitas Jember dengan pertanyaan andalannya "Kapan sidang?",
5. Almamater Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Jember,
6. Seluruh pihak yang mendukung saya dalam pengerjaan skripsi ini yang tidak dapat saya sebutkan satu-persatu.

MOTTO

"How can you move forward if you keep regretting the past?"

-Edward Elric-

"The moment you think of giving up, think of the reason why you held on so long"

-Natsu Dragneel-

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Ivan Budianto

NIM : 192410103035

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul ”Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman Pertanian dan Perkebunan Melalui Penggunaan Jaringan Syaraf Tiruan Klasifikasi Multinomial” adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak mana pun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 9 Januari 2022

Yang menyatakan,

Ivan Budianto

NIM 192410103035

SKRIPSI

KLASIFIKASI KONDISI TANAH BERDASARKAN REKOMENDASI TANAMAN PERTANIAN DAN PERKEBUNAN MELALUI PENGGUNAAN JARINGAN SYARAF TIRUAN KLASIFIKASI MULTINOMIAL

Oleh:

Ivan Budianto

NIM 192410103035

Pembimbing:

Dosen Pembimbing Utama : Prof. Dr. Saiful Bukhori, ST., M.Kom
Dosen Pembimbing Pendamping : Nova El Maidah, S.Si., M.Cs

PENGESAHAN PEMBIMBING

Skripsi berjudul "Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman Pertanian dan Perkebunan Melalui Penggunaan Jaringan Syaraf Tiruan Klasifikasi Multinomial" karya Ivan Budianto telah diuji dan disahkan pada:

hari, tanggal : 9 Januari 2023

tempat : Program Studi Informatika Fakultas Ilmu Komputer
Universitas Jember

Disetujui oleh:

Pembimbing I,	Pembimbing II,
Prof. Dr. Saiful Bukhori, ST., M.Kom	Nova El Maidah, S.Si., M.Cs
NIP. 196811131994121001	NIP. 198411012015042001

PENGESAHAN PENGUJI

Skripsi berjudul "Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman Pertanian dan Perkebunan Melalui Penggunaan Jaringan Syaraf Tiruan Klasifikasi Multinomial" karya Ivan Budianto telah diuji dan disahkan pada:

hari, tanggal : 9 Januari 2023

tempat : Program Studi Informatika Fakultas Ilmu Komputer
Universitas Jember

Disetujui oleh:

Penguji I,	Penguji II,
Prof. Drs. Slamin, M.Comp.Sc., Ph.D	Gayatri Dwi Santika, S.SI., M.Kom
NIP. 196704201992011001	NRP. 0016019201

Mengesahkan
Dekan Fakultas Ilmu Komputer

Drs. Antonius Cahya Prihandoko, M.App.Sc, Ph.D
NIP. 196909281993021001

RINGKASAN

Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman Pertanian dan Perkebunan Melalui Penggunaan Jaringan Syaraf Tiruan Klasifikasi Multinomial; Ivan Budianto, 192410103035; 2022; 110 Halaman; Program Studi Informatika Fakultas Ilmu Komputer Universitas Jember

Luas lahan pertanian untuk Indonesia mencapai 7,46 juta hektar. Coronavirus-19 mulai menjadi pandemi dan menyebabkan peningkatan mencapai 8 juta petani di Indonesia. Pertambahan jumlah petani tidak didampingi dengan literasi yang baik terkait pemilihan tanaman pertanian dan perkebunan yang paling sesuai untuk untuk kondisi tanah. Kondisi tanah dibagi menjadi dua, yaitu kondisi tanah secara kimiawi dan fisik. Kondisi tanah secara kimiawi meliputi kandungan Natrium, Fosfor, Hidrogen, Kalium, dan Kalsium. Sedangkan, secara fisik meliputi suhu harian, kelembapan, pH, dan curah hujan. Mengembangkan sebuah algoritma yang mengakomodasi atribut dalam jumlah besar membutuhkan waktu yang banyak untuk developer, serta dapat meningkatkan waktu eksekusi, serta biaya untuk perawatan dan pengembangan perangkat lunak. Machine Learning adalah sebuah paradigma yang berbeda. Machine learning dapat mengubah data masukan dan data keluaran untuk menghasilkan sebuah model sebagai keluaran. Melalui analisa pola data, dapat ditemukan sebuah model yang cocok untuk mengidentifikasi pola dari data masukan dan keluaran. Salah satu bidang machine learning adalah jaringan syaraf tiruan. Dalam penelitian ini, terdapat 9 skenario pengembangan model yang bervariasi pada rasio pembagian data dan jumlah layer yang digunakan. Berdasarkan seluruh percobaan yang dilakukan, skenario kedelapan adalah skenario terbaik dengan pembagian data 90% data latih, 5% data validasi, dan 5% data uji dengan 4 layer. Model ini mencapai 99.30% akurasi latih, 99.24% akurasi validasi, dan 98.93% akurasi uji. Model ini juga mendapatkan metrik 99.24% Precision, 99.49% Recall, 99.32% F1 Score.

PRAKATA

Puji syukur dihaturkan kepada Tuhan Yang Maha Esa. Atas berkat, rahmat, dan karunia-Nya penulis dapat menyelesaikan skripsi dengan judul "Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman Pertanian dan Perkebunan Melalui Penggunaan Jaringan Syaraf Tiruan Klasifikasi Multinomial". Penyusunan skripsi ini dilakukan untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan strata satu (S1) Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Jember. Penyusunan skripsi ini tidak lepas dari bantuan banyak pihak, maka penulis menyampaikan banyak terima kasih kepada:

1. Ayah saya Sugianto dan Ibu saya Lilly Mulianingsih yang terus mendukung baik dan mendoakan saya agar dilancarkan dalam menempuh pendidikan,
2. Kakak saya yang selalu memotivasi dalam penyelesaian skripsi ini,
3. Jessica Maya Berlianasari yang selalu mendukung dan memotivasi saya dalam penyusunan ide hingga revisi dari naskah skripsi ini,
4. Teman-teman Laboratorium Pemrograman yang selalu menyemangati saya dalam pengerjaan skripsi ini,
5. Prof. Drs. Slamin, M.Comp.Sc., Ph.D, selaku Plt. Dekan Fakultas Ilmu Komputer,
6. Dosen Pembimbing Utama Prof. Dr. Saiful Bukhori, ST., M.Kom dan Dosen Pembimbing Pendamping Ibu Nova El Maidah, S.Si., M.Cs yang dengan sabar telah membimbing saya hingga bisa menyelesaikan tugas akhir saya,
7. Dosen Penguji skripsi Prof. Drs. Slamin, M.Comp.Sc., Ph.D. dan Ibu Gayatri Dwi Santika, S.SI., M.Kom,
8. Semua pihak yang terlibat dalam skripsi ini yang tidak dapat saya sebutkan satu-persatu.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1. PENDAHULUAN

Bagian pendahuluan memberikan penjelasan terhadap latar belakang dari penelitian yang dilakukan hingga penjabaran permasalahan dan hasil yang diinginkan. Bagian ini mencakup latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, dan batasan masalah.

1.1 Latar Belakang

Luas lahan pertanian untuk Indonesia mencapai 7,46 juta hektar (Bahfein, 2021). Berdasarkan data dari Kominfo (2021), 27,33% dari penduduk Indonesia memilih untuk bekerja di sektor pertanian. Menteri Pertanian Indonesia, Syahrul Yasin Limpo, menyatakan bahwa jumlah petani di Indonesia terus meningkat sejak pandemi Coronavirus-19, dan telah mencapai angka pertambahan sebanyak 8 juta orang. Namun, pertambahan jumlah petani tidak didampingi dengan literasi yang baik terkait pemilihan tanaman pertanian dan perkebunan yang paling sesuai untuk untuk kondisi tanah, meskipun kecocokan antara kondisi tanah dan tanaman berdampak besar dalam produktivitas tanaman. Contohnya adalah kesalahan dalam memilih lokasi lahan tanam yang tidak sesuai membuat menurunnya hasil produksi kentang di Kabupaten Wonosobo (Khomsatun, 2020).

Penelitian yang dilakukan oleh Darol (2021) membuktikan bahwa terdapat relasi antara kondisi tanah dengan hasil tanam yang didapatkan. Dalam penelitiannya, beberapa jenis tanah yaitu tanah alluvial, regosol, mediteran, dan tanah kontrol digunakan sebagai media penanaman terong asam. Dari hasil yang didapatkan, dapat diketahui bahwa selisih rata-rata pertambahan tinggi batang tanaman terong asam berbeda-beda antar perlakuan. Hal ini membuktikan bahwa setiap jenis tanah memiliki kondisi tanah yang berbeda-beda, dan memiliki tingkat kecocokan yang berbeda dengan tanaman terong asam.

Kondisi tanah sendiri dibagi menjadi dua jenis, yaitu kondisi tanah secara kimiawi dan fisik. Secara kimiawi, unsur-unsur tersebut meliputi persentase kandungan unsur-unsur dan ion seperti Natrium, Fosfat, Hidrogen, Potassium, dan Kalsium (Russel, 2005). Selain kondisi tanah secara kimiawi, Karamina (2017)

juga menambahkan bahwa kondisi lingkungan secara fisik juga patut diperhatikan. Faktor kondisi fisik tanah meliputi seperti temperatur harian dari lahan, kelembapan, pH, dan curah hujan dari daerah tersebut.

Solusi dari permasalahan dalam penentuan tanaman yang cocok untuk kondisi tanah tertentu adalah pengembangan sebuah model jaringan syaraf tiruan untuk melakukan klasifikasi kondisi tanah berdasarkan rekomendasi tanaman untuk kondisi tanah tersebut. Pemilihan paradigma *machine learning* dalam penyelesaian ini didasarkan pada penelitian oleh Alassadi (2019). Penelitian ini membuktikan bahwa mengembangkan sebuah algoritma yang mengakomodasi atribut dalam jumlah besar membutuhkan waktu yang banyak untuk *developer*, serta dapat meningkatkan waktu eksekusi, serta biaya untuk perawatan dan pengembangan perangkat lunak (Alassadi, 2019). *Machine learning* mengenalkan sebuah perubahan paradigma dalam menghadapi permasalahan ini. Dalam machine learning, diminta data masukan dan data keluaran untuk menghasilkan sebuah model sebagai keluaran. Penelitian oleh Alassadi juga membuktikan algoritma *machine learning* menghasilkan akurasi yang lebih tinggi dibandingkan dengan penggunaan algoritma tradisional.

Terdapat penelitian terdahulu yang membahas mengenai klasifikasi kondisi tanah berdasarkan rekomendasi tanaman dengan perbedaan pada metode dan parameter yang diteliti. Salah satunya adalah penelitian Rohit Kumar Rajak et.al. (2017). Penelitian ini membahas mengenai pengembangan sebuah sistem rekomendasi dengan menggunakan algoritma *Support Vector Machine* (SVM), *Naïve Bayes, Multi-layer perceptron,* dan *Random Forest*. Data yang digunakan adalah dataset dengan atribut tanah berupa *Depth, Texture,* Ph, *Soil Color, Permeability, Drainage, Water holding* dan *Erosion*.

Metode yang dapat digunakan dalam penyelesaian masalah ini adalah penggunaan jaringan syaraf tiruan. Melalui penggunaan jaringan syaraf tiruan, informasi dapat diberikan kepada komputer sehingga komputer memiliki kecerdasan dan kemampuan deduksi selayaknya manusia (Korner, 2022). Jaringan syaraf tiruan digunakan untuk mengenali pola data kondisi tanah dengan parameter yang telah ditentukan sebelumnya. Parameter yang diteliti dalam penelitian ini

adalah kondisi tanah secara kimiawi yang meliputi kadar Natrium, Fosfat, Kalium, serta kondisi tanah secara fisik yang meliputi temperatur, humiditas, pH, dan curah hujan.

Solusi yang menjadi luaran dari penelitian ini adalah pengembangan model berbasis jaringan syaraf tiruan untuk menyelesaikan permasalahan berupa mengetahui tanaman pertanian dan perkebunan terbaik sesuai kondisi tanah secara spesifik melalui parameter berupa Natrium, Fosfat, Kalium, temperatur, humiditas, pH tanah, dan curah hujan.

1.2 Rumusan Masalah

Berdasarkan paparan latar belakang, berikut adalah rumusan masalah yang dirancang dalam penelitian yang dilakukan.

1. Bagaimana merancang dan mengembangkan jaringan syaraf tiruan untuk melakukan klasifikasi kondisi tanah berdasarkan rekomendasi tanaman?
2. Bagaimana tingkat akurasi dan hasil evaluasi pada klasifikasi kondisi tanah berdasarkan rekomendasi tanaman menggunakan metrik akurasi?

1.3 Tujuan Penelitian

Berdasarkan kedua poin rumusan masalah yang telah disampaikan, tujuan dari penelitian yang dilakukan adalah sebagai berikut.

1. Merancang dan mengembangkan jaringan syaraf tiruan untuk melakukan klasifikasi kondisi tanah berdasarkan rekomendasi tanaman pertanian dan perkebunan yang paling sesuai.
2. Mengukur dan mengevaluasi hasil perancangan dan pengembangan melalui metrik akurasi pada model jaringan syaraf tiruan yang telah dikembangkan dalam melakukan klasifikasi kondisi tanah berdasarkan rekomendasi tanaman.

1.4 Manfaat Penelitian

Manfaat praktis dari penelitian ini adalah sebagai berikut.

1. Manfaat Bagi Penulis

   Sebagai kesempatan untuk mengaplikasikan ilmu yang telah dipelajari selama masa perkuliahan terkait pengembangan jaringan syaraf tiruan dan proses deployment.

2. Manfaat Akademis

   Memberikan studi literatur terhadap cara untuk mengembangkan jaringan syaraf tiruan untuk membantu mengembangkan sektor agroindustri melalui klasifikasi kondisi tanah berdasarkan nama tanaman yang paling sesuai.

1.5 Batasan Masalah

Untuk mendefinisikan lingkup permasalahan secara lebih spesifik, berikut adalah batasan dalam penelitian ini.

1. *Dataset* yang digunakan adalah *dataset* yang didapatkan dari Kaggle *"Crop Recommendation Dataset"* oleh Atharva Ingle yang terdiri dari 8 kolom dan 2200 *record* data.
2. Label hasil klasifikasi terdiri dari 22 nama tanaman yang merupakan tanaman pertanian dan perkebunan yang dapat tumbuh di tanah Indonesia.
3. Penelitian berfokus pada proses pengembangan model dengan menggunakan *framework deep learning* TensorFlow.
4. Penelitian ini tidak membahas mengenai cara ekstraksi data secara langsung dari tanah.

BAB 2. TINJAUAN PUSTAKA

Bagian tinjauan pustaka berisi referensi yang digunakan dalam penelitian ini, dimulai dari literatur yang mendukung penelitian ini, hingga penjelasan detail mengenai jaringan syaraf tiruan (*Artificial Neural Network*) yang akan dikembangkan.

2.1 Kondisi Tanah

Kondisi tanah berpengaruh secara signifikan terhadap produktivitas lahan. Produktivitas lahan dapat berkurang akibat perlakuan kepada tanah oleh petani. Hal ini diduga disebabkan oleh beberapa faktor, diantaranya cara pengelolaan lahan yang kurang baik yang berakibat terhadap menurunnya tingkat kesuburan fisik, kimia, dan biologi tanah. Dari ketiga parameter kesuburan lahan tersebut, sifat fisik tanah sangat berpengaruh terhadap kesuburan kimia dan biologi tanah (Sumarni, 2010). Berdasarkan pernyataan tersebut, dapat disimpulkan bahwa kesuburan tanaman sangatlah dipengaruhi oleh kondisi tanah tempat tumbuhnya.

Pernyataan ini diperkuat melalui studi kasus yang dilakukan oleh Darol (2021). Dalam penelitiannya, beberapa jenis tanah yaitu tanah alluvial, regosol, mediteran, dan tanah kontrol digunakan sebagai media penanaman terong asam. Dari hasil yang didapatkan, dapat diketahui bahwa selisih rata-rata pertambahan tinggi batang tanaman terong asam berbeda-beda antar perlakuan (Darol, 2021). Kedua penelitian ini membuktikan bahwa terdapat korelasi antara kondisi tanah tempat tumbuh dari tanaman dengan tanaman yang ditanam. Tanaman akan tumbuh lebih baik dengan kondisi tanah yang cocok.

2.1.1 Kondisi Kimiawi Tanah

Unsur-unsur yang mempengaruhi kesuburan tanah secara kimiawi meliputi persentase kandungan unsur-unsur dan ion seperti Natrium, Fosfat, Hidrogen, Potassium, dan Kalsium (Russel, 2005). Hal ini juga menjadi salah satu alasan mengapa pupuk atau *fertilizer* yang digunakan dalam meningkatkan

kesuburan tanah memiliki unsur NPK atau Natrium, Fosfat, dan Kalium untuk memperbaiki kesuburan tanah.

2.1.2 Kondisi Fisik Tanah

Cepat dan lambatnya suatu pertumbuhan pada berbagai jenis tanaman sangat ditentukan oleh pH tanah itu sendiri. Dalam ilmu pertanian pengaruh terhadap pH tanah sangat memiliki peranan yang sangat penting gunanya untuk menentukan mudah tidaknya ion-ion unsur hara diserap oleh tanaman. Pada umumnya, unsur hara akan mudah diserap tanaman pada pH 6-7, karena pada pH tersebut sebagian besar unsur hara akan mudah larut dalam air. Derajat pH dalam tanah juga menunjukkan keberadaan unsur-unsur yang bersifat racun bagi tanaman. Kelembaban dan temperatur tanah yang baik membuat tanah menjadi memiliki ruang pori yang cukup sehingga sirkulasi udara di dalam tanah dapat berjalan dengan baik. Dengan tanah yang sehat tanah mampu memiliki nilai pH netral sehingga sebagian besar jenis tanaman dapat tumbuh dengan baik (Karamina, 2017). Sehingga, dapat disimpulkan bahwa kelembaban, temperatur, dan pH dari tanah akan mempengaruhi tanaman yang dapat ditanam pada lahan tersebut.

2.2 Tanaman Pertanian

Tanaman pertanian adalah bagian dari dunia tumbuh-tumbuhan (*plant*) yang berupa sekelompok makhluk hidup yang bertambah besar dan berkembang serta memiliki batang, akar, daun, dan sebagainya yang memiliki klorofil. Tanaman adalah tumbuhan yang sengaja ditanam dan dipelihara oleh manusia untuk dimanfaatkan. Tanaman pertanian, dengan demikian, adalah tanaman yang bermanfaat secara ekonomi dan cocok dengan rencana kerja dan eksistensi manusia. Tanaman pertanian memiliki sifat yang sangat khas, yaitu pengelolaan. Artinya tanaman itu selama kehidupannya dikelola untuk dipanen hasilnya meskipun tingkat pengelolaannya tidak intensif. Misalnya, tanaman dalam sistem pertanian ladang berpindah ataupun tanaman di lahan pekarangan (Sagala, 2021). Subsektor pertanian tanaman pangan meliputi komoditas padi, jagung, kacang tanah, kedelai, ketela rambat, dan tanaman-tanaman lainnya (Sri, 2013).

2.3 Tanaman Perkebunan

Undang-Undang Nomor 39 Tahun 2014 tentang Perkebunan menyatakan bahwa Perkebunan adalah segala kegiatan pengelolaan sumber daya alam, sumber daya manusia, sarana produksi, alat dan mesin, budidaya, panen, pengolahan, dan pemasaran terkait Tanaman Perkebunan. Tanaman Perkebunan adalah tanaman semusim atau tanaman tahunan yang jenis dan tujuan pengelolaannya ditetapkan untuk usaha perkebunan (de Lima, 2019).

Tanaman perkebunan yang ada di Indonesia antara lain: kelapa (*Cocos mufifera*), pala (*Myristisca frogranas houtt)*, aren (*Arenga pinnata*), kakao (*Theobroma cacao*), cengkih (*Syzygiumaromatioum*). Tanaman buah-buahan antara lain: jambu biji (*Psidium guajava*), jambu mete (*Arnacidium occidentale*), rambutan (*Nephelium lappaceum*), mangga (*mangifera indica*), pisang (*Musa paradisiacal*) dan langsat (*Lansium domesticium*) (Darmawan, 2021).

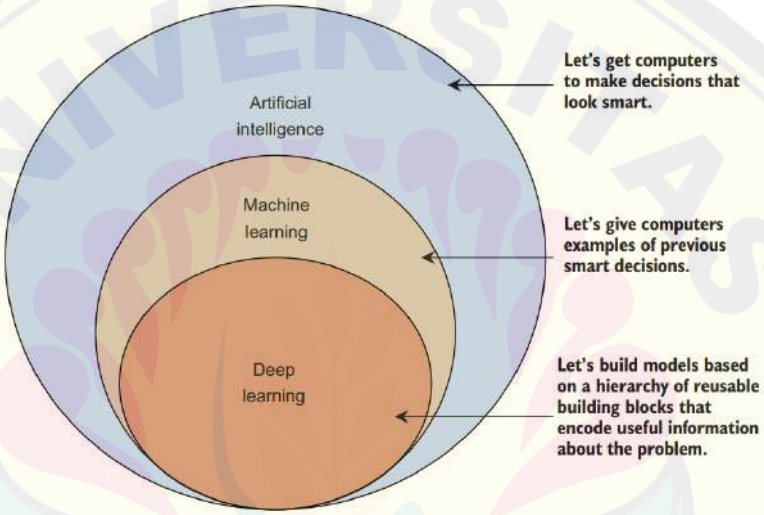
2.4 Penyuluhan Pertanian Lapangan

Penyuluhan pertanian mempunyai pengertian yaitu proses pembelajaran bagi pelaku utama serta pelaku usaha agar mereka mau dan mampu menolong dan mengorganisasikan dirinya dalam mengakses informasi pasar, teknologi, permodalan, dan sumber daya lainnya, sebagai upaya untuk meningkatkan produktivitas, efisiensi usaha, pendapatan, dan kesejahteraannya, serta meningkatkan kesadaran dalam pelestarian fungsi lingkungan hidup (Peraturan Menteri Pertanian Republik Indonesia No. 03 Tahun 2018 Tentang Pedoman Penyelenggaraan Penyuluhan Pertanian).

Jumlah tenaga penyuluh pertanian di Indonesia mencapai 44.000 orang. Tercatat dari 72.000 desa yang berpotensi di bidang pertanian, baru tersedia 44.000 tenaga penyuluh pertanian. Jumlah tenaga penyuluh yang berstatus pegawai negeri sipil saat ini mencapai 25.000 orang, sedangkan yang bersatus Tenaga Harian Lepas Tenaga Bantu Penyuluh Pertanian (THL-TBPP) berjumlah 19.000 orang. Dari 44.000 tenaga penyuluh, 32.000 diantaranya yang bersentuhan langsung dengan petani di lapangan (Vintarno, 2019).

Terdapat beberapa tingkatan penyuluh, yaitu tingkat kabupaten, kecamatan, dan desa. Penyuluh pertanian tingkat desa memiliki kewajiban untuk membimbing langsung petani. Merekrut atau mengadakan tenaga penyuluh pertanian sangat penting dalam melakukan penyuluhan kepada kelompok tani dan secara tidak langsung keberadaan penyuluh dapat mengubah perilaku petani untuk mencapai swasembada pangan di Indonesia (Pricylia et al, 2018).

2.5 Kecerdasan Buatan



Gambar 2.1 Perbedaan Artificial Intelligence, Machine Learning, dan Deep Learning (Wolf, 2022)

Kecerdasan Buatan, atau *Artificial Intelligence* menurut Russell dan Norvig (2009) adalah cabang ilmu pengetahuan alam yang membuat mesin mampu menampilkan kecerdasan seperti manusia untuk sebuah segmen spesifik. Alan Turing pada sekitar tahun 1950 mempresentasikan hasil penemuannya dalam komputasi mesin (Sowa, 1992 dikutip dari Russel dan Norvig (2009)). Beberapa sumber juga memberikan penjelasan bahwa "AI adalah bagian dari ilmu komputer yang dikarakteristikkan dengan kecerdasan dalam perilaku manusia". Kecerdasan Buatan berdasarkan Gambar 2.1 juga merupakan daerah yang mencakup seluruh konsep yang ada pada *machine learning* dan *deep learning*.

Kecerdasan buatan mencakup seluruh kemampuan mesin untuk dapat berpikir seperti manusia secara luas. Contohnya adalah saat membuat komputer untuk memainkan sebuah video gim yang dapat dipecahkan melalui penggunaan *approach* berbasis AI murni, dimana hanya peraturan gamenya saja yang di-*encode* tanpa perlu memperlihatkan bagaimana cara bermainnya (Raff, 2022).

2.6 Machine Learning

Gambar 2.1 menjelaskan bahwa *machine learning* merupakan subdomain dari kecerdasan buatan. *Machine learning* sebagai pengembangan dari kecerdasan buatan akan memberikan kemampuan pada komputer untuk mengembangkan AI untuk bermain catur dengan memberikan beberapa contoh permainan yang dimainkan oleh *Grandmaster* catur untuk dipelajari (setiap permainan ada pemenang dan yang kalah, dan hasil berupa keputusan yang diambil apakah langkah bagus atau kurang bagus) (Raff, 2022).

Kesimpulan yang bisa didapat adalah *machine learning* merupakan implementasi untuk mengajarkan bagaimana untuk melakukan sebuah tugas secara spesifik melalui data yang diberikan (Wolf, 2022). *Machine learning* sendiri dibagi menjadi dua bagian, yaitu *supervised* ML, dan *unsupervised* ML (Wolf, 2022).

Pengembangan machine learning telah mencapai sebuah titik dimana terdapat algoritma dan metode yang dapat digunakan dengan mudah dalam pengembangan model. Terdapat beberapa algoritma dan metode yang telah ditemukan dan dikembangkan dalam *machine learning*, seperti algoritma tradisional dengan contoh *Support Vector Machine* (SVM) atau *Decision Tree* (DT), dan metode baru seperti *Deep Learning* (DL) (Ghayoumi, 2022).

2.7 Supervised Learning

Implementasi machine learning dalam permasalahan dunia nyata berpusat pada penyelesaian masalah yang memiliki korelasi antara fitur dan labelnya. Dalam permasalah *real-world*, diberikan sebuah objek dengan detail yang telah kita ketahui, seperti hasil pengamatan, pengukuran, lalu memprediksi bagaimana hasil

dari objek lain yang masih tidak diketahui (Wolf, 2022). Berdasarkan penjelasan tersebut, *supervised learning* akan memprediksi variabel target yang bertipe data kategorikal, atau numerik. Bila dilakukan klasifikasi lebih lanjut, maka terdapat dua macam *supervised learning*, yaitu permasalahan klasifikasi dan regresi.

Permasalahan klasifikasi dapat didefinisikan sebagai berikut. "Diberikan sebuah set data latih dengan label latih yang terkait, tentukan sebuah label kelas untuk sebuah data yang tidak berlabel" (Alassadi, 2019). Inti dari *supervised learning* adalah prediksi dilakukan pada kategori yang masih belum diketahui, atau disebut *class label*, berdasarkan variabel pada set data yang telah diketahui dan/atau diobservasi (Wolf, 2022). *Supervised* ML digunakan untuk melakukan klasifikasi untuk memberikan sebuah label pada sebuah kategori yang diskrit. Dalam pengaplikasiannya, mayoritas dari solusi berbasis ML adalah solusi berbasis *supervised* ML (Ping, 2022).

2.8 Jaringan Syaraf Tiruan (*Artificial Neural Network*)

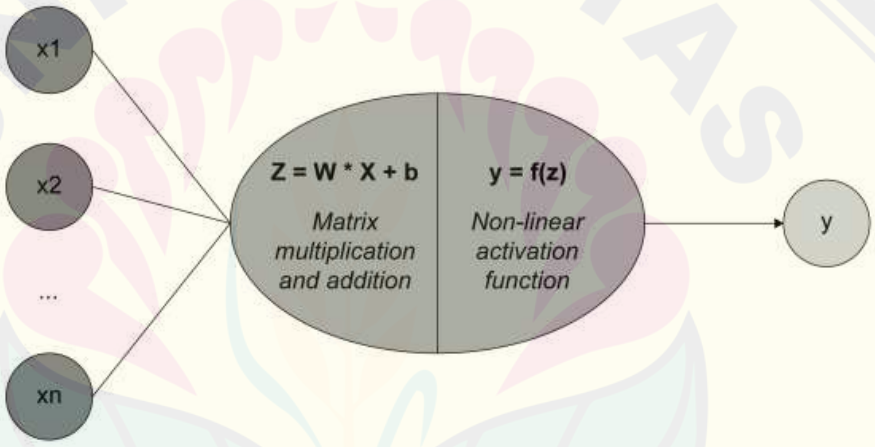
Jaringan syaraf tiruan, atau lebih dikenal dengan *Artificial Neural Network*, merupakan cara untuk mereplikasi bagaimana otak manusia bekerja, terutama pada bagian bagaimana cara kerja dari neuron (So, 2020). Neuron adalah sel otak yang berkomunikasi dengan sel lain melalui sinyal elektrik. Neuron dapat merespon melalui stimuli seperti suara, cahaya, dan sentuhan. Neuron juga dapat memicu kontraksi otot, dan lainnya. Dalam ANN, inputnya adalah dendroid, tempat terjadi proses kalkulasi adalah nukleus, dan outputnya adalah akson (So, 2020).

Sebuah neuron tiruan didesain untuk mereplikasi cara kerja dari nukleus. Neuron tersebut akan mengubah sinyal input menjadi sebuah perkalian matriks, dan dilanjutkan dengan sebuah fungsi aktivasi. Apabila fungsi ini menentukan bahwa sebuah neuron harus aktif, maka sinyal tersebut akan muncul pada *output*. Terdapat kemungkinan bahwa sinyal ini akan menjadi input dari neuron lain pada jaringan tersebut (So, 2020).

Jaringan syaraf tiruan adalah peta representasi, dan merupakan pengenalan pola dari fitur poin data sebagai input menuju label yang ingin diprediksi sebagai

outputnya (Jung, 2022). Dalam pengaplikasian Jaringan syaraf tiruan akan dibangun sebuah jaringan artifisial dari neuron manusia melalui perancangan layer sedemikian rupa dengan masing-masing jumlah neuron dan fungsi aktivasi yang berbeda untuk mencapai sebuah tujuan spesifik yang diinginkan. ANN memiliki beberapa kelebihan yang membuatnya cocok untuk masalah dan kondisi tertentu: (Hasnira et al., 2020) yang akan dideskripsikan dalam poin-poin sebagai berikut.

1. ANN memiliki respon kecepatan yang tinggi.
2. ANN memiliki ketahan operasi
3. ANN dapat menyebabkan sedikit komputasi
4. ANN dapat dijadikan solusi untuk masalah yang *multi-variable*



Gambar 2.2 Ilustrasi Contoh Jaringan Syaraf Tiruan (So, 2020)

Ilustrasi mengenai contoh jaringan syaraf tiruan digambarkan pada Gambar 1.2. Gambar tersebut menceritakan mengenai cara kerja jaringan syaraf tiruan. Berikut adalah gambaran mengenai komputasi yang dilakukan pada Gambar 2.2.

Masukan dari pengguna berupa parameter fitur yang diberikan (disimbolkan melalui x_1, x_2, x_3, hingga x_n) akan diberikan. Setiap fitur memiliki *weight* (bobot) yang berbeda-beda. Pada neuron yang memiliki fungsi aktivasi linear, maka rumusnya adalah hasil perkalian setiap fitur dengan bobot tersebut, ditambahkan dengan biasnya.

Bila diambil contoh terdapat 4 buah fitur, maka contoh dari persamaan adalah serangkaian penghitungan Z dan aktivasi fungsi yang digambarkan melalui Persamaan 2.1 dan Persamaan 2.2 dengan poin-poin permisalan sebagai berikut.

1. **X** adalah matriks *input*, yang terdiri dari x_1, x_2, x_3, dan x_4.
2. **W** adalah matriks bobot, yang terdiri dari w_1, w_2, w_3, dan w_4.
3. **b** adalah bias.
4. **f** adalah fungsi aktivasi yang digunakan.

Pertama perlu dihitung z melalui perkalian matriks dan penambahan dengan bias seperti Persamaan 2.1.

$$Z = \left(\sum_{n=1}^{n=4} x_n w_n\right) + b \qquad (2.1)$$

Kemudian, hasil berupa prediksi, y, akan dihitung dengan mengaplikasikan fungsi aktivasi f pada hasil Z yang telah didapatkan pada persamaan 2.1 sebelumnya seperti pada kalkulasi Persamaan 2.2.

$$y = f(Z) = f\left(\left(\sum_{n=1}^{n=4} x_n w_n\right) + b\right) \qquad (2.2)$$

Hasil berupa y merupakan keluaran yang diharapkan dari proses prediksi yang dilakukan oleh jaringan syaraf tiruan (So, 2020).

2.9 Bobot

Bobot, atau biasa lebih dikenal dengan matriks bobot, adalah parameter yang dipelajari oleh jaringan syaraf secara otomatis untuk memprediksi output yaitu y, secara akurat. Sebuah neuron tunggal adalah kombinasi dari total perkalian input dan bobot dengan fungsi aktivasi, dan bisa direferensikan sebagai sebuah *hidden layer*. Jaringan syaraf dengan sebuah *hidden layer* disebut dengan *regular neural network* (So, 2020).

Penggunaan jaringan syaraf tiruan adalah saat sebuah jaringan terkompilasi. Jaringan tersebut dapat dicocokkan, yang berarti mengadaptasikan bobot dari model untuk merespon terhadap *dataset* latih. Hasil dari adaptasi tersebut

akan digunakan sebagai parameter yang akan dikalikan dengan data yang akan diprediksi untuk mendapatkan hasil akhir berupa label prediksi, yaitu y (Brownlee, 2019).

2.10 Bias

Bias adalah sebuah konstan yang sering juga disebut *intercept*. Dengan bantuan sebuah faktor konstan yang dibangun dalam sebuah neuron, sebuah model jaringan syaraf menjadi lebih fleksibel dalam proses *fitting* pada *training dataset* yang spesifik (So, 2020). Bias dalam model jaringan syaraf juga membantu untuk mengurangi terjadinya kesalahan, terutama pada data ambang atau pada data baru yang didapat. Data-data tersebut dapat mempengaruhi akurasi model, dan menyebabkan model menjadi lebih rigid.

2.11 Fungsi Aktivasi

Sebuah fungsi aktivasi digunakan untuk melakukan konversi dari nilai masukan menjadi nilai yang spesifik, seperti 0 dan 1 sebagai keluaran dari klasifikasi biner (Palmas, 2020). Beberapa fungsi aktivasi yang berbeda dapat digunakan dalam jaringan syaraf. Tanpa fungsi aktivasi, sebuah jaringan syaraf hanya merupakan model linear yang dapat dideskripsikan dengan mudah melalui perkalian matriks. Penggunaan fungsi aktivasi menyediakan sifat non-linear, maka dari itu, dimungkinkan untuk memodelkan pola yang lebih kompleks (So, 2020).

Sebuah fungsi aktivasi berperan penting dalam pengembangan sebuah model jaringan syaraf tiruan. Dikutip dari Mire et.al. (2022) bahwa limitasi dari jaringan syaraf adalah performanya bergantung pada fungsi aktivasi yang digunakan, jumlah *hidden layer* dan neuron, dan algoritma yang digunakan dalam melatih jaringan syaraf tersebut. Terdapat beberapa macam fungsi aktivasi yang akan digunakan, dan dideskripsikan sebagai berikut.

2.11.1 Fungsi Aktivasi *ReLU*

ReLU merupakan singkatan dari *Rectified Linear Unit*. Fungsi aktivasi ini adalah fungsi yang paling sering digunakan untuk hidden layer. Persamaan 2.3 adalah persamaan untuk melihat cara kerja ReLU. (So, 2020).

$$f(x) = \begin{cases} 0 \text{ for } x \leq 0 \\ x \text{ for } x > 0 \end{cases} \quad (2.3)$$

ReLU memiliki konsep seperti indra manusia. Apabila nilai lebih kecil atau sama dengan 0, maka tidak ada respon dari neuron. Namun, bila nilai ini lebih dari 0, maka akan terdapat reaksi bernilai x itu sendiri.

2.11.2 Fungsi Aktivasi *Softmax*

Fungsi aktivasi *softmax* merupakan fungsi yang biasa digunakan pada *layer* terakhir pada jaringan syaraf untuk permasalahan *multi-class classification* dimana *softmax* dapat membuat probabilitas untuk setiap kelas nilai keluaran (So, 2020).

Fungsi aktivasi *softmax* dapat mentransformasikan nilai keluaran menjadi sebuah kumpulan angka yang bernilai antara 0 dan 1, sehingga jumlah elemen dalam kumpulan tersebut menjadi 1. Persamaan 2.4 merupakan persamaan dalam fungsi aktivasi *softmax* (So, 2020).

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \text{ for } i = 1, ..., J \quad (2.4)$$

2.12 Normalisasi (*MinMax Scaling*)

Normalisasi adalah melakukan *rescale* dari data asli sehingga nilai dari seluruh data asli bernilai antara range 0 dan 1. Normalisasi membutuhkan pengetahuan secara akurat mengenai estimasi dari nilai terkecil dan terbesar dari data yang dapat diobservasi. Nilai tersebut dapat diperoleh dari data yang *available*. Salah satu nilai *default* dari MinMaxScaler adalah melakukan *rescale* dari variabel menuju nilai antara 0 dan 1 (Brownlee, 2019).

Normalisasi data akan sangat membantu dalam komputasi yang lebih baik, dan biaya komputasi yang lebih rendah. Normalisasi juga dapat meningkatkan akurasi dari hasil (Ghayoumi, 2022). Persamaan 2.5 akan menjelaskan mengenai proses normalisasi *MinMax Scaling* yang dilakukan.

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}} \qquad (2.5)$$

Nilai x yang melalui proses normalisasi MinMax Scaling akan bernilai diantara 0 dan 1. *Range* 0 dan 1 diciptakan melalui penyebut yang merupakan hasil pengurangan dari nilai terbesar dan terkecil yang akan selalu bernilai 1, yang dikombinasikan dengan pengurangan pembilang yang merupakan nilai yang ingin dinormalisasi yang dikurangkan dengan nilai terkecil.

2.13 Metrik Evaluasi

Terdapat tahapan lanjutan setelah melakukan proses prediksi dari sebuah model, yaitu melakukan evaluasi model. setelah melakukan prediksi, perlu diketahui apakah prediksi yang dibuat merupakan prediksi yang baik atau tidak. Terdapat beberapa evaluasi standar yang dapat digunakan untuk mengetahui prediksi yang telah dilakukan. Dengan mengetahui hasil prediksi yang tepat, dapat diketahui estimasi bagaimana model yang telah dibuat dalam menyelesaikan permasalahan (Brownlee, 2016). Berikut adalah beberapa metrik evaluasi yang umum digunakan dalam proses evaluasi permasalahan klasifikasi multinomial.

2.13.1 Akurasi

Pemodelan klasifikasi prediktif juga melibatkan prediksi dari sebuah label dari kelas yang telah diberikan pada permasalahan. Metrik paling umum yang digunakan untuk mengevaluasi kemampuan dari sebuah klasifikasi prediktif adalah akurasi klasifikasi. Biasanya, akurasi dari sebuah model prediktif dianggap bagus bila lebih dari 90% akurasi. Akurasi klasifikasi melibatkan penggunaan dari sebuah model klasifikasi untuk membuat klasifikasi untuk setiap data yang ada pada data uji. Prediksi tersebut kemudian akan dibandingkan dengan label yang telah diketahui sebelumnya dari data tersebut pada data uji. Akurasi kemudian dihitung

sebagai proporsi dari data yang ada pada data uji yang dapat diprediksi dengan tepat, kemudian dibagi dengan seluruh prediksi yang telah dilakukan pada data uji (Brownlee, 2020). Rumus dari klasifikasi dapat dilihat pada Persamaan X.

Akurasi adalah metrik yang paling simpel dan paling banyak digunakan untuk mengukur kemampuan dari sebuah model klasifikasi. Namun, akurasi tidak selalu merupakan metrik yang terbaik, terutama bila data yang diteliti tidak seimbang. Permasalahan mendasarnya adalah apabila kelas negatif tersebut dominan, maka akurasi tinggi dapat dicapai hanya dengan memprediksi nilai negatif pada seluruh kasus.

Salah satu cara untuk mengukur akurasi adalah melalui penggunaan confusion matrix. Sebuah confusion matrix adalah rangkuman dari seluruh prediksi yang telah dilakukan oleh model klasifikasi yang ditata dalam sebuah tabel dengan kelas yang ada. Setiap baris pada tabel mengindikasikan kelas secara aktual dan setiap kolom merepresentasikan prediksi kelas. Sebuah sel pada tabel adalah penghitungan dari jumlah prediksi untuk sebuah kelas. Setiap sel secara diagonal merepresentasikan prediksi yang benar, dimana harusnya prediksi dan ekspektasi kelas sejajar (Brownlee, 2020).

Confusion matrix memberikan lebih banyak insight tidak hanya pada akurasi dari sebuah model prediktif, namun juga memberikan informasi mengenai kelas mana saja yang berhasil diprediksi dengan benar dan tidak, dan tipe kesalahan apa yang dibuat (Brownlee, 2020).

2.13.2 Precision

Precision adalah sebuah metrik yang menghitung jumlah dari prediksi dengan kelas positif yang diprediksi. Maka dari itu, precision menghitung akurasi untuk kelas minoritas. Precision digunakan untuk menghitung rasio dari prediksi kelas positif benar yang dibagi dengan jumlah dari seluruh kelas positif yang diprediksi. Precision tidak hanya terbatas pada permasalahan klasifikasi biner saja. Dalam permasalahan klasifikasi multinomial dengan data yang tidak seimbang, precision dihitung sebagai total seluruh True Positive dibagi dengan jumlah True

Positive dan False Positive dari seluruh kelas (Brownlee, 2020). Rumus dari precision dapat dilihat pada Persamaan 2.6.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (2.6)$$

2.13.3 Recall

Recall adalah sebuah metrik yang menghitung jumlah dari prediksi dengan kelas positif yang diprediksi. Tidak seperti precision yang hanya mengindikasikan prediksi positif yang benar dari seluruh prediksi positif, recall menyediakan sebuah indikasi dari prediksi positif yang terlewat. Melalui hal ini, recall menyediakan informasi terkait kelas positif dan negatif (Brownlee, 2020). Rumus dari recall dapat dilihat pada Persamaan 2.7.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \quad (2.7)$$

2.13.4 F-Measure

F-measure menyediakan sebuah jalan untuk mengkombinasikan baik precision dan recall dalam sebuah pengukuran yang mencakup kedua properti ini. Precision ataupun recall tidak dapat untuk menceritakan seluruh hal yang terjadi. Terdapat kemungkinan terdapat model yang memiliki precision yang sangat baik, namun recall yang buruk, ataupun sebaliknya. Penggunaan F-measure dapat mengakomodasi keduanya melalui sebuah nilai saja. Setelah menghitung precision dan recall dari model, kedua nilai tersebut dapat dikombinasikan dalam kalkulasi F-measure. Rumus dari F-measure yang umum digunakan adalah F1 Score. Rumus dari F1 Score dapat dilihat pada Persamaan 2.8 (Brownlee, 2020).

$$F1\ Score = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall} \quad (2.8)$$

2.14 Testing

Testing adalah bagian yang tidak dapat dihindari dalam proses pengembangan sistem. Tujuan dari testing perangkat lunak adalah untuk menemukan kekurangan pada perangkat lunak dan untuk memverifikasi bahwa produk yang telah dikembangkan telah memenuhi kebutuhan pengguna yang telah ditentukan pada awal dari proses pengembangan perangkat lunak. Testing perangkat lunak menjadi salah satu proses yang perlu diperhatikan, dimana membutuhkan perencanaan, eksekusi, dan kontrol yang lebih sistematis agar membuat testing menjadi lebih produktif (Jaaskelainen, 2012). Seluruh aktivitas testing memiliki tujuan yang sama, yaitu melakukan evaluasi pada kualitas produk untuk meningkatkan kepercayaan pada pengembangan perangkat lunak untuk berfungsi sesuai dengan fungsi pada awalnya (Lonetti, 2018).

Apabila diklasifikasikan berdasarkan informasi saat testing, terdapat 2 macam testing, yaitu black-box testing dan white-box testing. Berikut adalah penjelasan mengenai black-box testing dan white-box testing.

2.14.1 Black-box Testing

Black-box testing adalah sebuah strategi testing berdasarkan kebutuhan dan spesifikasi perangkat lunak tersebut. Black-box testing tidak membutuhkan pengetahuan mengenai bagian dalam perangkat lunak, struktur, ataupun implementasinya. Metodologi testing ini melihat pada masukan dari sebuah aplikasi, dan ekspektasi keluaran berdasarkan setiap masukan. Sebuah contoh adalah sebuah alat untuk melakukan test otomatis. Seorang tester akan menggunakan alat tersebut dengan skrip yang telah ditulis sebelumnya dan menjalankannya. Tetapi, tester tersebut tidak wajib mengerti mengenai permasalahan teknikalnya ataupun skrip yang sedang digunakan (Jaaskelainen, 2012).

2.14.2 White-box Testing

White-box testing adalah sebuah strategi testing berdasarkan bagian dalam, struktur kode, dan implementasi dari sistem yang sedang ditesting. White-box testing pada umumnya membutuhkan skill programming yang detail. Programmer tersebut juga harus memiliki kemampuan dalam implementasi detail

dari perangkat lunak terkait dan juga pengetahuan mengenai test scripting (Jaaskelainen, 2012).

2.15 Penelitian Terdahulu

Penelitian terdahulu yang ditemukan dan berkaitan dengan penggunaan jaringan syaraf tiruan, atau *Artificial Neural Network* (ANN) adalah jurnal oleh Rohit Kumar Rajak et.al. (2017). Penelitian ini membahas mengenai pengembangan sebuah sistem rekomendasi untuk memaksimalkan hasil tanaman melalui penggunaan teknik machine learning. Teknik yang digunakan dalam penelitian ini meliputi *Support Vector Machine (*SVM*), Naïve Bayes, Multi-layer perceptron,* dan *Random Forest*. Data yang digunakan adalah *dataset* dengan atribut tanah berupa *Depth, Texture,* Ph*, Soil Color, Permeability, Drainage, Water holding* dan *Erosion*. Melalui penelitian tersebut, didapatkan rules untuk menentukan kondisi tanah. Penelitian terdahulu ini bermanfaat dalam mengetahui parameter yang dibutuhkan dan penguatan dasar dari latar belakang mengenai parameter yang dibutuhkan dalam melakukan prediksi.

Penelitian lainnya adalah penelitian yang dilkaukan oleh Nidhi H. et.al. pada tahun 2018. Penelitian ini menggunakan metode ensemble dari 3 metode, yaitu *Random Forest, Naïve Bayes*, dan SVM. Akan dibuat model dari setiap metode secara parsial terlebih dahulu. Setiap model telah memprediksi masing-masing data dengan akurasi yang dapat diterima. Lalu, label kelas dari masing-masing model akan dikumpulkan menjadi sebuah ensemble, dimana akan diambil hasil vote terbanyak dari 3 model tersebut untuk dijadikan label akhir klasifikasi (Nidhi, 2018). Melalui metode ensemble, didapatkan hasil akhir berupa model yang dapat memprediksi tanaman terbaik dengan akurasi mencapai 99,91%. Penelitian ini memberikan gambaran mengenai model dan kumpulan algoritma yang dapat digunakan untuk menyelesaikan masalah yang serumpun melalui metode ensemble, serta metrik yang digunakan untuk mengukur akurasi dari model. Penelitian ini juga menjadi salah satu dasar dalam pemecahan masalah dengan latar belakang serupa, yaitu klasifikasi kondisi tanah berdasarkan tanaman terbaik untuk ditanam pada kondisi tanah tersebut.

Penelitian terdahulu berikutnya membahas mengenai software model untuk pertanian presisi oleh Satish Babu (2013). Dalam penelitian tersebut, digunakan sebuah model analitik yang mensimulasikan kalender tanam melalui temperatur dan curah hujan melalui beberapa sensor yang dapat digunakan untuk mendapatkan parameter-parameter yang dibutuhkan (Babu, 2013). Penelitian ini memberikan gambaran garis besar mengenai dibutuhkannya beberapa parameter fisik tanah untuk membantu proses analisis yang dilakukan, yaitu temperatur dan curah hujan dari tanah tersebut.

Penelitian berikutnya adalah penelitian oleh Wibowo (2021) mengenai perbandingan metode klasifiasi data mining untuk rekomendasi tanaman pangan. Penelitian tersebut memberikan gambaran mengenai metode *Random Tree* sebagai metode terbaik dalam melakukan klasifikasi rekomendasi tersebut. Dalam perbandingan ini, data yang diambil adalah sampel data BMKG tahun 2017, meliputi temperatur rata-rata, kelembapan rata-rata, tekanan udara, lama penyinaran matahari, dan curah hujan, dimana hal ini mendukung penggunaan parameter yang digunakan dalam penelitian yang dilakukan, dimana parameter yang digunakan adalah parameter yang menyerupai parameter yang ada pada dataset. Parameter tersebut meliputi kondisi tanah secara fisik seperti temperatur, curah hujan, dan pH.

Penelitian terakhir yang menjadi referensi bagi penulis adalah referensi dari Musyarrofatul (2018). Penelitian ini membahas mengenai sistem pendukung keputusan (SPPK) pemilihan tanaman melalui metode AHP. Dalam penelitian ini digunakan beberapa parameter seperti pH, C Organik, Drainase, Tekstur, dan Temperatur. Hal ini mendukung penggunaan parameter yang sama dalam penelitian yang dilakukan. Penelitian ini menggunakan sistem berbasis website yang juga menjadi salah satu dasar dalam fitur yang ada pada website yang dikembangkan, seperti adanya pembagian hak akses untuk masing-masing pengguna, fitur yang dapat diakses, serta desain sistem seperti alur dari sistem.

BAB 3. METODOLOGI PENELITIAN

Bagian metode penelitian membahas mengenai metode yang digunakan dalam penelitian, objek penelitian, serta tahapan yang dilakukan dalam penelitian ini untuk mencapai tujuan. Bagian ini juga menjelaskan mengenai jaringan syaraf tiruan (ANN) yang telah dikembangkan, serta gambaran umum sistem yang telah dikembangkan.

3.1 Jenis Penelitian

Metode yang digunakan dalam penelitian ini adalah penelitian kuantitatif. Pendekatan penelitian kuantitatif berdasarkan arti secara harfiah adalah segala sesuatu bisa diamati, dan diukur. Pendekatan penelitian kuantitatif adalah penelitian terukur yang menghasilkan angka dan dianalisis dengan statistika deskriptif ataupun inferensial (Surya, 2019). Dalam penelitian yang dilakukan terkait dengan penjabaran angka, mulai dari pengumpulan data, interpretasi data, dan hasil yang diinginkan. Penelitian dilakukan melalui peninjauan lebih lanjut dari objek yang diteliti, dan dilakukan penarikan kesimpulan berdasarkan hasil penelitian yang meliputi hasil pengembangan model dan pengembangan sistem yang dilakukan.

3.2 Objek Penelitian

Data klasifikasi kondisi tanah berdasarkan nama tanaman yang paling sesuai didapatkan dari *Crop Recommendation Dataset* yang diunggah oleh Atharva Ingle pada platform Kaggle. Data tersebut merupakan data hasil augmentasi dari parameter-parameter tanah secara kimiawi maupun fisik dan terdiri dari 8 kolom dan 2200 *record* data. Nama tanaman yang ada dalam dataset adalah apel, pisang, kedelai hitam, kacang chickpea, kelapa, kopi, kapas, anggur, rami, kacang merah, kacang lentil, jagung, mangga, kacang ngengat, kacang hijau, melon, jeruk, papaya, kacang gude, delima, padi, dan semangka. Tabel 3.1 memberikan penjelasan dan detail mengenai fitur dan label dalam *dataset* yang menjadi objek dalam penelitian yang meliputi kondisi tanah secara fisik dan kimiawi, serta nama tanaman.

Tabel 3.1 Penjelasan Kolom Objek Penelitian

No	Nama Atribut	Keterangan	Tipe Data
1.	N	Rasio dari kandungan Nitrogen dalam tanah yang diteliti	Int64
2.	P	Rasio dari kandungan Fosfor dalam tanah yang diteliti	Int64
3.	K	Rasio dari kandungan Kalium dalam tanah yang diteliti	Int64
4.	temperature	Temperatur tanah dalam satuan Celcius	Float64
5.	humidity	Kelembapan dari tanah secara relatif dalam satuan persen	Float64
6.	pH	Nilai pH dalam tanah	Float64
7.	rainfall	Curah hujan secara fisik dari tanah tersebut dalam satuan milimeter	Float64
8.	label	Label yang berisi nama tanaman rekomendasi untuk tanah tersebut	String

Tabel 3.2 memberikan contoh 10 data teratas dari *dataset* yang menjadi objek dalam penelitian yang meliputi kondisi tanah secara fisik dan kimiawi, serta nama tanaman. Nilai dari kondisi tanah secara kimiawi memiliki tipe data bilangan bulat, sedangkan kondisi tanah secara fisik memiliki tipe data *float* atau dapat berupa desimal. Sedangkan, label adalah nama tanaman dengan tipe data *string*.

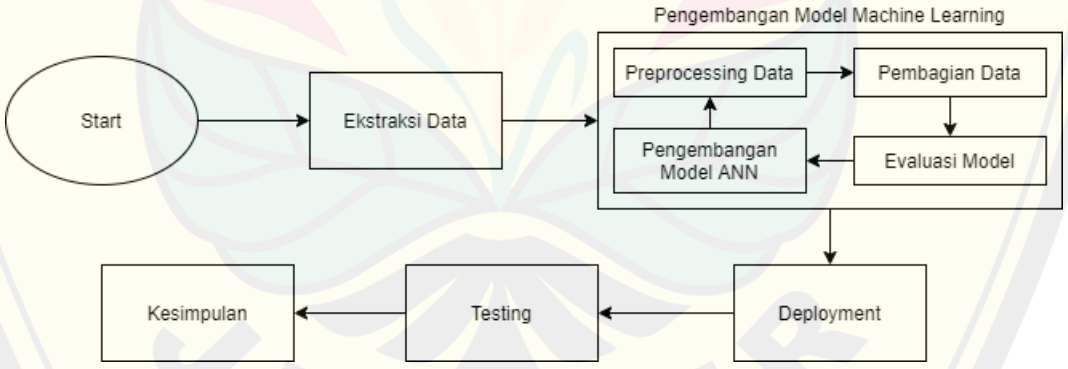
Tabel 3.2 Sepuluh data teratas dari dataset

No.	N	P	K	temperature	humidity	ph	rainfall	label
1	90	42	43	20.890	82.002	6.503	202.936	Rice
2	85	58	41	21.770	80.320	7.038	226.656	Rice
3	60	55	44	23.004	82.321	7.840	263.964	Rice
4	74	35	40	26.491	80.158	6.980	242.864	Rice

No.	N	P	K	temperature	humidity	ph	rainfall	label
5	78	42	42	20.130	81.605	7.628	262.717	Rice
6	69	37	42	23.058	83.370	7.073	251.055	Rice
7	69	55	38	22.709	82.639	5.701	271.325	Rice
8	94	53	40	20.278	82.894	5.719	241.974	Rice
9	89	54	38	24.5159	83.535	6.685	230.446	Rice
10	68	58	38	23.224	83.033	6.336	221.209	Rice

3.3 Tahap Penelitian

Tahap penelitian merupakan tahapan apa saja yang dilalui untuk mendapatkan hasil penelitian, yaitu model jaringan syaraf tiruan, dan akurasi model tersebut. Terdapat beberapa proses yang meliputi proses ekstraksi data, 4 tahapan dalam proses pengembangan model yang akan diiterasikan untuk mendapatkan model dengan skenario terbaik, hingga proses deployment, testing, dan penarikan kesimpulan dari keseluruhan penelitian yang telah dilakukan. Gambar 3.1 adalah *flowchart overview* mengenai proses penelitian yang telah dilalui.



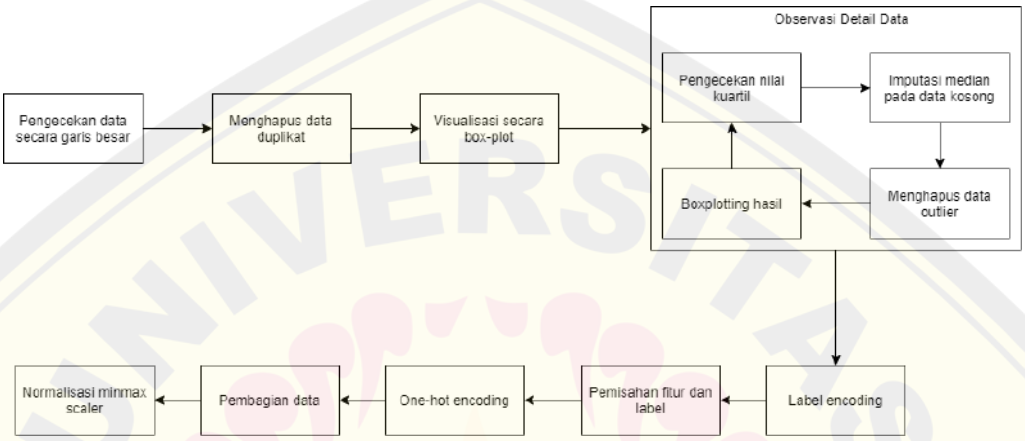
Gambar 3.1 Proses Tahapan Penelitian

3.3.1 Ekstraksi Data

Data dari Kaggle berupa 2200 *record* data dengan 8 fitur diunduh dan diekstrak untuk memudahkan proses pengembangan yang dilakukan. Hasil dari unduhan tersebut diakuisisi lebih lanjut menjadi *file* CSV yang siap diolah lebih lanjut menjadi dataset yang matang dalam proses-proses berikutnya.

3.3.2 Preprocessing Data

Data mentah yang telah didapat dari proses ekstraksi data didapatkan. Data ini tidak dapat digunakan secara langsung dan diperlukan beberapa proses normalisasi agar hasil yang didapatkan menjadi lebih akurat dan mengurangi beban pemrosesan data pada komputer saat pembangunan ANN. Langkah yang dilakukan dalam proses preprocessing data dapat dilihat pada Gambar 3.2.



Gambar 3.2 Tahapan dalam Preprocessing Data

Terdapat beberapa langkah yang dilakukan pada dataset. Pertama, akan dilakukan pengecekan data secara garis besar untuk menentukan kondisi dataset, seperti ada atau tidaknya data yang duplikat, data yang kosong, atau kondisi abnormal lainnya. Setelah itu, data duplikat akan dihapus untuk mengetahui kondisi dataset sebenarnya. Visualisasi boxplot dilakukan untuk mengetahui persebaran data dalam tiap kuartil beserta letak dari data outlier.

Preprocessing akan dilakukan dengan perulangan untuk observasi detail dari data. Nilai dari kuartil data akan diobservasi untuk mengetahui apakah data tersebut sudah memiliki nilai yang penuh, dan bukan merupakan outlier. Apabila terdapat data kosong, maka akan dilakukan imputasi median pada data tersebut. Kemudian, apabila data tersebut merupakan data outlier, maka akan dilakukan penghapusan pada data tersebut. Proses ini akan berlangsung hingga seluruh data telah diobservasi dan akan digunakan dalam proses pengembangan model.

Proses berikutnya adalah proses efisiensi dari data untuk mengurangi biaya komputasi. Proses ini meliputi *label encoding* untuk mengubah label yang

semula memiliki tipe data *string* menjadi *integer* dengan range angka 0 hingga 21 dan proses pemisahan untuk membagi fitur dan label menjadi X dan y, proses *one-hot encoding* untuk mengubah label menjadi *list* biner.

Melalui proses ini, terdapat 1.980 data latih, 110 data validasi, dan 110 data uji. Fitur dari tiap kategori data ini dinormalisasi dengan *MinMax Scaler* untuk mengubah *range* angka dari 0 hingga 1.

3.3.3 Pembagian Data

Dataset dibagi menjadi data latih, data validasi, dan data uji dengan 3 rasio perbandingan. Data dapat dibagi menjadi data latih, validasi, dan uji. Pembagian data yang umum dilakukan adalah 70% data latih, 15% data validasi, dan 15% data uji, 80% data latih, 10% data validasi, dan 10% data uji. Dalam implementasinya, beberapa skenario lain juga dapat diimplementasikan berdasarkan hasil dari performa model (Ghayoumi, 2022). Pembagian 90% data latih dan 10% data uji juga bisa menjadi pilihan yang baik dan rasional (Joseph, 2022). Tabel rasio pembagian data dapat dilhat pada Tabel 3.3.

Tabel 3.3 Pembagian Data

No. Skenario	Rasio Data Latih	Rasio Data Validasi	Rasio Data Uji
1	70%	15%	15%
2	80%	10%	10%
3	90%	5%	5%

Perbandingan pertama adalah 70% data latih, 15% data validasi, dan 15% data uji. Perbandingan kedua adalah 80% data latih, 10% data validasi, dan 10% data uji. Sedangkan, perbandingan terakhir adalah 90% data latih, 5% data validasi, dan 5% data uji. Sebagai contoh pada skenario ketiga, dari 2.200 *record* data yang dimiliki dibagi menjadi 1.980 dataset latih yang digunakan dalam pelatihan model ANN, 110 dataset validasi untuk menguji validitas model dalam setiap epoch, dan 110 dataset uji untuk menguji akurasi model yang telah dikembangkan dalam data

yang belum pernah ditemui. Dalam pembagian data juga dilakukan pengacakan data dengan *seed* yang telah ditentukan sebelumnya, sehingga tidak ada perubahan data latih dan data tes meskipun dilakukan berulang kali pelatihan model. Metode yang digunakan untuk proses pembagian data adalah *method* pada Sci-kit Learn, yaitu *Train Test Split*.

3.3.4 Pengembangan Model ANN

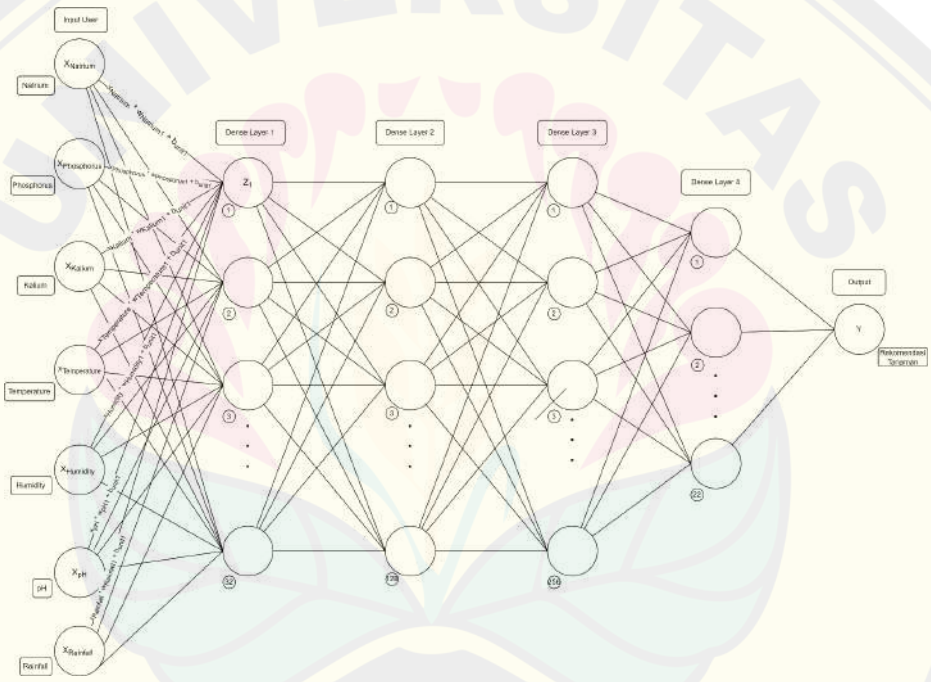
Proses pengembangan model dilakukan dengan menggunakan *library* Python untuk membuat model, yaitu TensorFlow. Gambar 3.2 merupakan rancangan model dan *layer* beserta contoh bobot dan bias yang ada pada *layer* 1 dalam proses pengembangan model ANN. Terdapat 3 skenario dari model yang dibangun berdasarkan jumlah layer yang digunakan. Tabel 3.4 merupakan tabel mengenai seluruh skenario yang dilakukan dalam penelitian ini.

Tabel 3.4 Skenario Penelitian

No. Skenario	Jumlah Layer	Rasio Data Latih	Rasio Data Validasi	Rasio Data Uji
1	3	70%	15%	15%
2	4	80%	10%	10%
3	5	90%	5%	5%
4	3	70%	15%	15%
5	4	80%	10%	10%
6	5	90%	5%	5%
7	3	70%	15%	15%
8	4	80%	10%	10%
9	5	90%	5%	5%

Model yang digunakan adalah model dengan 4 layer. Pemilihan ini didasarkan pada penelitian terdahulu terkait klasifikasi multinomial yang membuktikan bahwa model dengan 4 layer memiliki akurasi tertinggi bila dibandingkan dengan 2 hingga 9 layer (Abu, 2018). Untuk menguji terkait jumlah layer, maka akan dilakukan pengembangan dengan 3 layer dan 5 layer. Model yang digunakan terdiri dari 4 layer dengan fungsi aktivasi ReLU, dan diakhiri dengan 22 unit dengan fungsi aktivasi *softmax*. Alasan penggunaan layer ReLU adalah layer ReLU merupakan layer terbaik dalam proses pelatihan pada layer atas, yang

dibuktikan pada penelitian terdahulu dengan permasalahan yang sama, yaitu klasifikasi multinomial (Ertam, 2017). Layer terakhir adalah layer output yang terdiri dari 22 unit dengan fungsi aktivasi *softmax*. Penelitian terdahulu terkait klasifikasi multinomial juga menggunakan fungsi aktivasi yang sama, yaitu *softmax.* Penggunaan fungsi aktivasi *softmax* didasarkan pada permasalahan yang membutuhkan solusi berupa klasifikasi multinomial, maka dibutuhkan lebih dari 2 label. Pemilihan optimizer berupa RMSProp didasarkan pada penelitian terdahulu oleh Chandra (2017) yang dikutip dari Schaul et. al (2014) menyatakan bahwa RMSProp mengungguli metode adaptif lain seperti Adagrad, Adadelta, bahkan SGD dengan momentum pada tes dalam jumlah besar.



Gambar 3.3 Rancangan Model ANN

Gambar 3.3 merupakan gambar rancangan dari model yang telah dikembangkan. Bila diambil dari X pada Gambar 3.3 dimana terdapat 7 buah fitur, maka contoh dari penghitungan masukan pada Unit 1 pada *Dense Layer* 1 adalah serangkaian perkalian dan penjumlahan seperti pada penghitungan Z pada Persamaan 2.1 dengan poin permisalan sebagai berikut.

1. **X** adalah matriks *input*, yang terdiri dari Natrium (1), Phosphorus (2), dan seterusnya, sehingga setiap input direpresentasikan menjadi x_1, x_2, x_3, x_4, x_5, x_6 dan x_7 yang telah dinormalisasi melalui *MinMax Scaling*.
2. **W** adalah matriks bobot, yang terdiri dari w_1, w_2, w_3, w_4, w_5, w_6, dan w_7 yang merepresentasikan bobot untuk *input* tersebut pada *unit* yang spesifik.
3. **b** adalah bias, yang bernilai b_{unit1} pada unit 1 Dense Layer 1.
4. **f** adalah fungsi aktivasi yang digunakan, yaitu ReLU yang diutilisasikan apabila seluruh 32 unit telah diakumulasikan.

Pertama perlu dihitung Z melalui perkalian matriks dan penambahan dengan bias seperti Persamaan 2.1, dengan formula penghitungan seperti pada Persamaan 3.1.

$$\begin{aligned} Z_1 &= \left(\sum_{n=1}^{n=7} x_n w_n\right) + b \\ &= \left((x_1 * w_1) + (x_2 * w_2) \dots (x_7 * w_7)\right) + b_{unit1} \\ &= Z_1 \end{aligned} \quad (3.1)$$

Masukan pengguna yang berjumlah 7 parameter tanah yaitu kadar Natrium, Fosfor, Kalium, temperatur, humiditas, pH, dan curah hujan dimasukkan pada jaringan syaraf tiruan untuk dikalkulasikan menjadi hasil prediksi yang berupa *list confidence* model terhadap masing-masing label yang ada (22 label). Dari hasil ini, dicari angka *confidence* yang paling besar dari model untuk dijadikan hasil keluaran setelah dilakukan *reverse label encoding*. Hasil akhir yang diharapkan adalah y yang merupakan nama tanaman yang paling sesuai untuk tanah dengan 7 parameter yang telah disebutkan sebelumnya.

3.3.5 Evaluasi Model

Model yang telah dikembangkan dievaluasi menggunakan metrik akurasi. Terdapat dua macam akurasi yang dihasilkan, yaitu akurasi yang diperoleh dari data latih, dan akurasi validasi yang diperoleh dari proses validasi yang dilakukan melalui dataset tes. Apabila akurasi masih dirasa kurang, maka model yang telah

dibuat dikembangkan lebih lanjut dengan mengubah beberapa *hyperparameter* pada ANN tersebut untuk mempelajari lebih banyak pola pada data tersebut, sehingga hasil dari pelatihan dapat menjadi lebih dimaksimalkan.

3.3.6 Deployment

Model terbaik yang didapatkan disimpan dalam bentuk H5 untuk memudahkan dalam proses deployment. Untuk menampung model tersebut, dibuat sebuah website sederhana menggunakan *micro-framework* Flask. Website ini digunakan untuk memudahkan proses utilisasi dari klasifikasi model ANN yang telah dikembangkan.

3.3.7 Testing

W*ebsite* diuji dalam tahap ini untuk mencoba apakah terdapat kesalahan mayor yang membuat aplikasi tidak dapat digunakan, atau model tidak dapat melakukan klasifikasi dengan baik. Terdapat dua macam pendekatan yang testing dilakukan, yaitu melalui pendekatan *white box* dan *black box testing*.

Pendekatan pertama adalah white box testing. White box testing berfungsi untuk meneliti kembali kode yang telah dibuat, dan menelitinya kembali melalui beberapa studi kasus. White box testing digunakan dalam pengujian model yang dibangun. Melalui white box testing, diharapkan kesalahan yang terjadi sebelum proses pengembangan model dapat dimitigasi, sehingga model dapat berkembang sesuai dengan ekspektasi. Testing yang akan dilakukan adalah pembuatan ulang sebuah model dengan arsitektur terbaik dari awal, hingga melakukan klasifikasi pada sebuah contoh parameter. Terdapat 6 test case yang telah diujikan dengan menggunakan bantuan unittest dari Python.

Pendekatan berikutnya adalah *black box testing* yang dilakukan dengan cara mencoba setiap view pada sistem dengan skenario dan *output* yang diharapkan. Pengujian black box testing dilakukan untuk mencoba fitur yang ada pada sistem yang telah ter-deploy. Terdapat 15 skenario yang telah diujikan dimulai dari masuk ke dalam sistem, melakukan klasifikasi kondisi tanah, hingga keluar dari sistem.

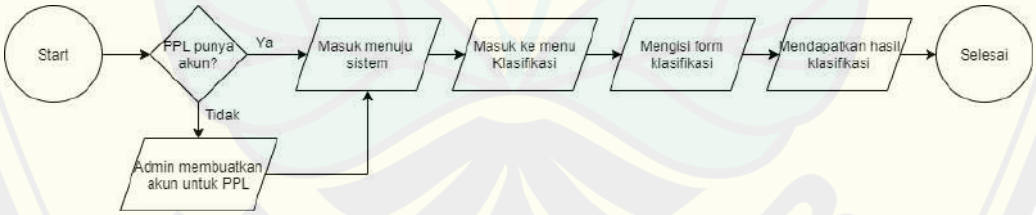
3.3.8 Kesimpulan

Kesimpulan ditarik berdasarkan seluruh hasil penelitian yang telah dilakukan. Hasil dari penelitian yang dilakukan berupa model yang telah dikembangkan, performa dari model, beserta hal-hal menjadi penemuan dalam penelitian. Dalam tahap ini juga ditentukan apakah sistem telah yang dikembangkan dapat menjawab rumusan masalah yang ada.

Seluruh kegiatan yang telah dilakukan selama proses penelitian akan dirangkum sehingga menjadi hasil penelitian. Kesimpulan merupakan hasil akhir dari keseluruhan proses yang dilakukan selama proses penelitian.

3.4 Gambaran Umum Sistem

Website yang dikembangkan merupakan *website* untuk membantu pengguna dalam mengetahui rekomendasi tanaman yang paling sesuai berdasarkan jenis tanah. *Dataset* yang digunakan adalah *Crop Recommendation Dataset* oleh Atharva Ingle. Keluaran yang dihasilkan dari *website* ini adalah nama tanaman yang paling sesuai berdasarkan kondisi tanah yang dimasukkan oleh pengguna. Gambar 3.4 adalah ilustrasi diagram alir untuk menjelaskan proses yang perlu ditempuh untuk melakukan klasifikasi nama tanaman yang paling sesuai berdasarkan kondisi tanah yang dimasukkan oleh pengguna.



Gambar 3.4 Diagram Alir Penggunaan Sistem

Hak akses dari website dibagi menjadi 3 jenis. Hak akses ini terdiri dari admin, Penyuluh Pertanian Lapangan (PPL), dan Petani. Admin dan PPL hanya dapat ditambahkan oleh admin. Petani dapat mendaftarkan diri melalui menu Daftar. Berdasarkan tugas pokok dan fungsi, PPL memiliki tugas dan fungsi memberikan penyuluhan kepada petani melalui pendekatan kelompok tani. Maka

dari itu, yang dapat melakukan klasifikasi hanyalah PPL dari petani yang bersangkutan.

Apabila PPL telah memiliki akun, maka PPL dapat masuk ke dalam sistem. Pengguna akan diarahkan menuju halaman utama dari sistem. Setelah itu, pengguna dapat menekan tombol “Klasifikasi” pada baris navigasi. Setelah itu, sistem akan menampilkan form prediksi rekomendasi tanaman. Pengguna akan diminta untuk mengisi *input* yang dibutuhkan, yaitu nilai Natrium, Fosfor, Kalium, temperatur, kelembapan, pH tanah, dan curah hujan. Sistem akan memulai proses klasifikasi setelah pengguna menekan tombol klasifikasi. Setelah proses klasifikasi selesai, hasil berupa nama tanaman akan diberikan kepada pengguna.

BAB 4. HASIL DAN PEMBAHASAN

Bagian hasil dan pembahasan berisi mengenai hasil dari penelitian yang dilakukan, beserta dengan penjelasan dari masing-masing bagian. Hasil dari penelitian merupakan jawaban atas rumusan masalah yang telah didefinisikan pada bagian pendahuluan.

4.1 Perancangan dan Pengembangan Jaringan Syaraf Tiruan untuk Melakukan Klasifikasi Kondisi Tanah Berdasarkan Rekomendasi Tanaman

Bagian-bagian dari perancangan dan pengembangan jaringan syaraf tiruan untuk melakukan klasifikasi kondisi tanah berdasarkan rekomendasi tanaman menjelaskan mengenai pengembangan sistem membahas mengenai proses yang dilakukan sebelum dilakukan pengembangan sistem. Pengembangan ini meliputi proses pengembangan model yang dilakukan pada proses ekstraksi data hingga *testing*. Penjelasan dari setiap bagian akan dijelaskan sebagai berikut.

4.1.1 Ekstraksi Data

Proses ekstraksi data adalah proses untuk mempersiapkan data yang diteliti. Proses ini dimulai dengan mengunduh data *Crop Recommendation Dataset* dari Kaggle. Data hasil unduhan adalah sebuah berkas dalam format arsip RAR. Berkas tersebut diekstrak untuk mendapatkan isi dari berkas tersebut, yaitu data mentah. File CSV yang berisi dataset adalah berkas yang telah diakuisisi. File CSV yang berhasil diakuisisi tersebut akan diletakkan dalam folder *dataset*. Untuk menganalisis dataset secara lebih lanjut, maka digunakan beberapa perangkat lunak yang akan membantu dalam proses komputasi. Dataset yang diakuisisi berisi 7 kolom fitur yang meliputi kondisi tanah secara fisik dan kimiawi, dan 1 kolom label berupa nama tanaman untuk kondisi tanah tersebut. Tabel 4.1 berisi 10 sampel data teratas dari *dataset*.

Tabel 4.1 Sampel 10 Data Teratas

No.	N	P	K	temperature	humidity	ph	rainfall	label
1	90	42	43	20.890	82.002	6.503	202.936	Rice
2	85	58	41	21.770	80.320	7.038	226.656	Rice
3	60	55	44	23.004	82.321	7.840	263.964	Rice
4	74	35	40	26.491	80.158	6.980	242.864	Rice
5	78	42	42	20.130	81.605	7.628	262.717	Rice
6	69	37	42	23.058	83.370	7.073	251.055	Rice
7	69	55	38	22.709	82.639	5.701	271.325	Rice
8	94	53	40	20.278	82.894	5.719	241.974	Rice
9	89	54	38	24.5159	83.535	6.685	230.446	Rice
10	68	58	38	23.224	83.033	6.336	221.209	Rice

Proses ekstraksi data meliputi proses import dari library yang dibutuhkan dalam proses ekstraksi data hingga proses pengembangan model. Library yang digunakan meliputi Pandas, NumPy, JSON, TensorFlow, dan Matplotlib. Alasan pemilihan library akan dijelaskan sebagai berikut.

Pandas dipilih karena Pandas menyediakan struktur data dan fungsionalitas untuk memanipulasi dan menganalisis data dengan cepat (Brownlee, 2016). NumPy memiliki fondasi dari struktur data berupa N-Dimensional Array (ndarray) yang efisien untuk didefinisikan dan dimanipulasi (Brownlee, 2016). JSON digunakan dalam proses parsing data dalam pemrosesan label dari data, serta pengolahan file dalam bentuk JSON. TensorFlow adalah library utama dalam proses pengembangan model yang akan dilakukan nantinya. Library berikutnya adalah TensorFlow. TensorFlow adalah library yang berfokus pada deep learning dan memiliki beragam library yang kompatibel dengan beragam API yang dapat digunakan oleh seluruh pengembang perangkat lunak. Fitur ini membuat TensorFlow menjadi salah satu platform untuk merancang algoritma deep learning. TensorFlow dapat dieksekusi pada CPU atau GPU pada satu atau lebih mesin (Ghayoumi, 2022). Library berikutnya adalah Sci-kit Learn, yaitu library yang digunakan dalam memudahkan preprocessing dari data, seperti melakukan

MinMax scaling, label encoding, one-hot encoding, dan train-test split. Library terakhir adalah Matplotlib yang digunakan untuk visualisasi dari proses pengembangan model, seperti visualisasi dari akurasi dan loss dari model setelah dilakukan pengembangan dari model. Gambar 4.1 merupakan kode untuk melakukan *import library* dan load dataset dari format CSV menjadi dataframe menggunakan library Pandas.

```
1  #import library yang dibutuhkan
2
3  import pandas as pd
4  import numpy as np
5  import json
6  import tensorflow as tf
7  import matplotlib.pyplot as plt
8
9  from sklearn.preprocessing import MinMaxScaler
10 from sklearn.preprocessing import LabelEncoder
11 from sklearn.model_selection import train_test_split
12
13 #pembuatan dataframe
14 file_name = "dataset/Crop_recommendation.csv"
15 df = pd.read_csv(file_name)
```

Gambar 4.1 Kode Akuisisi Dataset

Hasil ekstraksi data pada Gambar 4.1 yang tertampung pada variabel *df*, akan diproses terlebih lanjut untuk mendapatkan data statistikal dari data terkait *preprocessing* yang akan dilakukan. Proses ini dilakukan untuk mengetahui apakah terdapat data yang hilang, atau bernilai NaN. Melalui statistik dari data juga dapat diketahui statistika dasar dari dataset, seperti rata-rata, standar deviasi, nilai minimal, kuartil 1, median, kuartil 3, dan nilai maksimal. Kode untuk mendapatkan data statistikal dari *dataframe* menggunakan Pandas disediakan pada Gambar 4.2.

```
1 #mendapatkan deskripsi statistikal data
2 describe = df.describe()
3 print(describe)
```

Gambar 4.2 Kode untuk Mendapatkan Data Statistikal Dataset

Tabel 4.2 merupakan data statistikal dari dataset berdasarkan hasil kode pada Gambar 4.2. Berdasarkan data statistikal pada Tabel 4.2 dapat dilihat bahwa count dari seluruh kolom bernilai 2200. Hal ini menandakan bahwa tidak ada data yang hilang, dan data siap untuk diolah pada proses berikutnya. Penggunaan data statistikal juga membantu untuk mengetahui persebaran data.

Tabel 4.2 Data Statistikal Dataset

	N	P	K	temperature	humidity	ph	rainfall
count	2.200	2.200	2.200	2.200	2.200	2.200	2.200
mean	50,552	53,363	48,149	25,616	71,482	6,469	103,464
std	36,917	32,986	50,648	5,064	22,264	0,774	54,958
min	0,000	5,000	5,000	8,826	14,258	3,505	20,211
25%	21,000	28,000	20,000	22,769	60,262	5,972	64,552
50%	37,000	51,000	32,000	25,599	80,473	6,425	94,868
75%	84,250	68,000	49,000	28,562	89,949	6,924	124,268
max	140,000	145,000	205,000	43,675	99,982	9,935	298,560

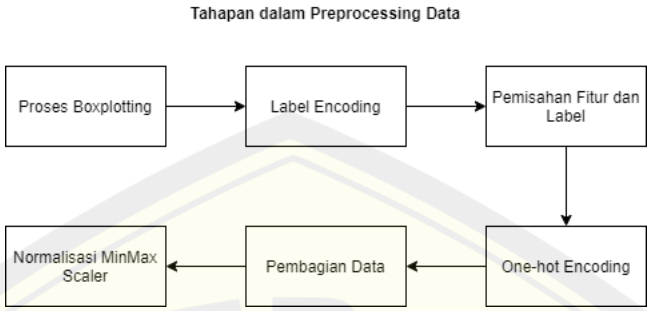
4.1.2 Pengembangan Model *Machine Learning*

Proses pengembangan model machine learning terdiri dari 4 sub-proses. Apabila dalam evaluasi model sebelumnya masih terdapat ruang pengembangan model, maka 4 sub-proses tersebut akan diiterasikan kembali untuk mendapatkan hasil model terbaik. Berikut adalah penjelasan mengenai 4 sub-proses pada tahap pengembangan model machine learning.

a. *Preprocessing* Data

Preprocessing data adalah proses untuk menyiapkan data menjadi data matang yang siap untuk diberikan pada model. Dalam penelitian ini, terdapat 6 proses yang dilakukan dalam preprocessing data. Proses tersebut adalah Boxplot, Label Encoding, pemisahan fitur dan label, One-hot Encoding, pembagian data, dan Normalisasi MinMax Scaling. Gambar 4.3 merupakan diagram mengenai proses

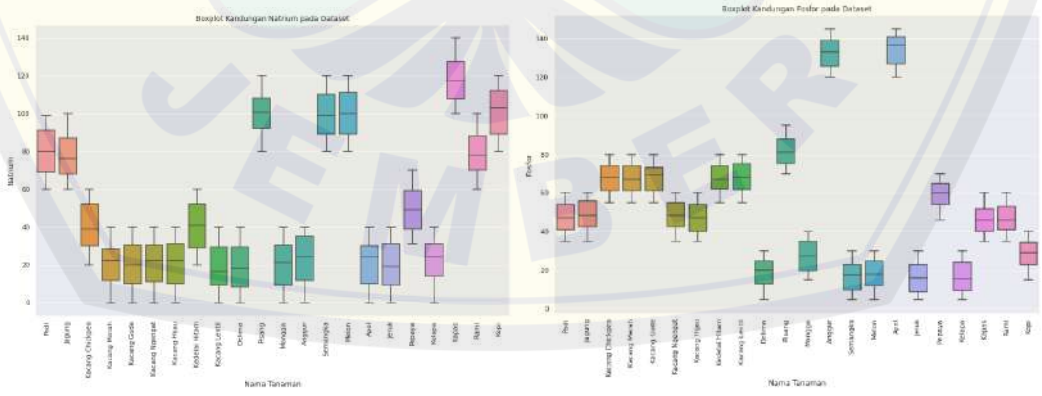
yang dilakukan dalam proses *preprocessing* data yang akan dijelaskan secara lebih rinci dalam penjelasan berikut.

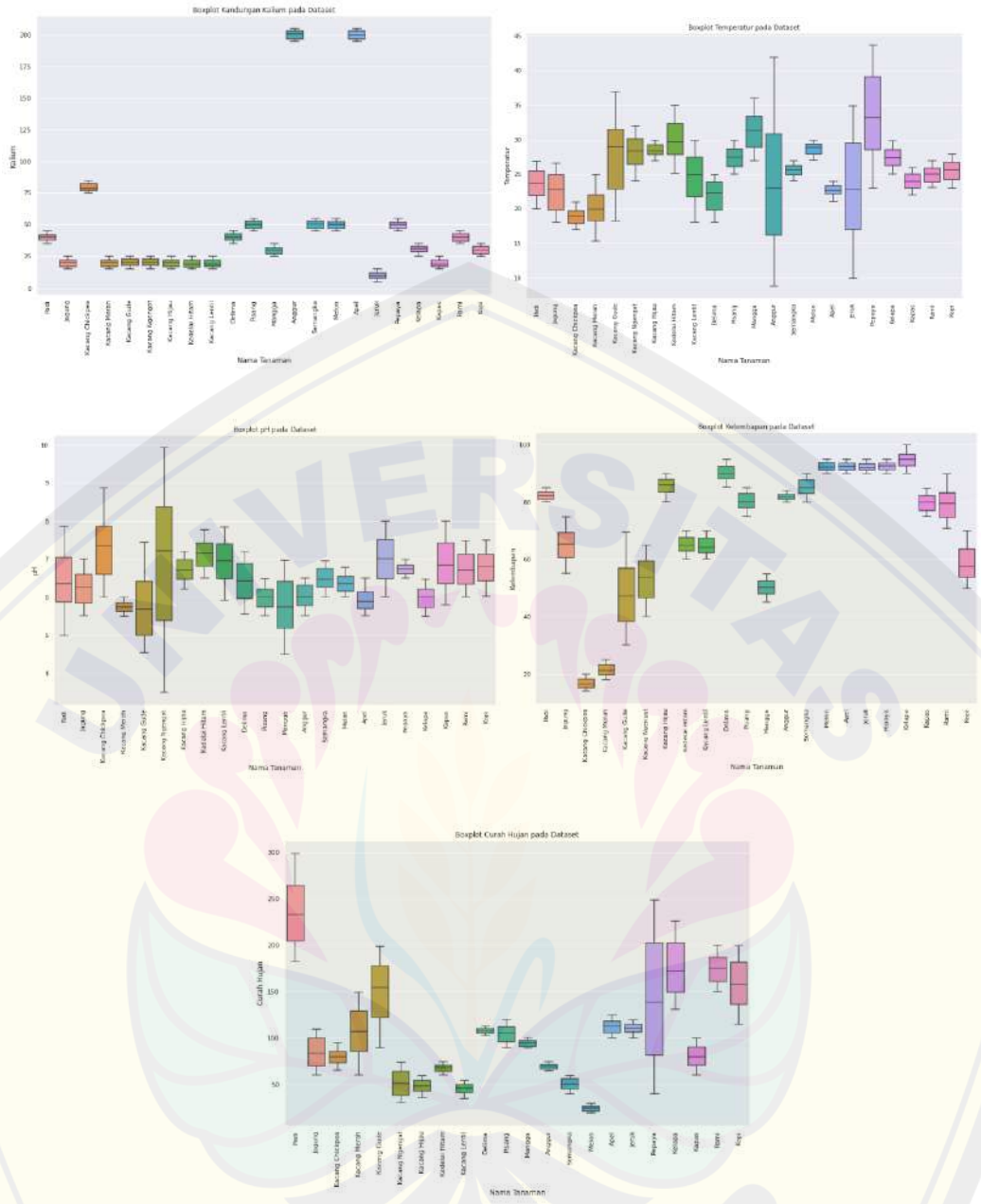


Gambar 4.3 Tahapan Pre-processing Data

1) Proses Boxplotting

Persebaran data dalam sebuah dataset dapat didapatkan secara lebih mendalam dengan melakukan boxplot untuk mengetahui adanya data yang memiliki karakteristik berbeda dalam data. Boxplot adalah ringkasan distribusi sampel yang disajikan secara grafis yang bisa menggambarkan bentuk distribusi data, ukuran tendensi sentral dan ukuran penyebaran pada data pengamatan. Gambar 4.4 merupakan boxplot pada setiap fitur pada dataset bila dibandingkan dengan setiap label pada dataset. Nilai setiap fitur pada Gambar 4.4 menunjukkan bahwa terdapat persebaran dengan nilai terkecil dan nilai terbesar dengan rentang nilai yang berpola. Dengan demikian, proses penelitian dapat dilanjutkan menuju proses berikutnya.





Gambar 4.4 Hasil Bloxpotting

2) *Label Encoding*

Proses *label encoding* yang dilakukan bertujuan untuk memudahkan model dalam melakukan komputasi data dengan efisien. Hasil dari proses ini akan disimpan ke dalam sebuah file untuk JSON (JavaScript Object Notation) untuk memudahkan proses pemanggilan label saat proses *reverse label encoding*.

Langkah pertama yang dilakukan dalam proses label encoding adalah mendefinisikan method yang akan digunakan, yaitu LabelEncoder dari Sci-kit Learn. Setelah itu, digunakan method fit pada label encoder dengan parameter berupa kolom label dari dataframe agar LabelEncoder dari Sci-Kit Learn dapat mengetahui label apa saja yang tersedia pada dataset. Hasil dari proses pada line 5 adalah sebuah dictionary dengan key berupa label dari dataset, sedangkan value bernilai angka yang berkisar antara 0 hingga 22. JSON hanya menerima tipe data string, maka dari itu perlu dilakukan casting dari value hasil label encoding yang semula integer menjadi string. Maka dari itu, digunakan fungsi casting dari Python yang dikombinasikan dengan struktur data dictionary. Setelah itu, library JSON digunakan untuk menyimpan hasil dari dictionary hasil label encoding ke dalam file JSON dalam file dataset/labelencoder_dict dengan ekstensi JSON. Lalu, langkah terakhir adalah mengubah label pada dataframe sesuai dengan angka yang merupakan hasil dari proses fitting yang telah dilakukan. Gambar 4.5 merupakan kode yang digunakan dalam *label encoding* pada data.

```
1  #encode label menjadi integer, dan menyimpannya dalam JSON
2
3  labelEncoder = LabelEncoder()
4  labelEncoder.fit(df["label"])
5  labelEncoder_key_value = dict(zip(labelEncoder.classes_,
   labelEncoder.transform(labelEncoder.classes_)))
6
7  keys_values = labelEncoder_key_value.items()
8  output_key_val = {str(key): str(value) for key, value in
   keys_values}
9
10 with open('dataset/labelEncoder_dict.json', 'w') as
   output_file:
11     json.dump(output_key_val, output_file, indent=4)
12
13 #pengaplikasian labelEncoder pada dataset
14 df["label"] = labelEncoder.transform(df["label"])
15 print(df.head(10))
```

Gambar 4.5 Kode Label Encoding

3) Pemisahan Fitur dan Label

Proses pembagian data dimulai dengan memisahkan fitur dan label menjadi 2 variabel yang berbeda, yaitu X dan y. Fitur berupa *list* dimana tiap anggotanya merupakan *list* dengan 7 anggota fitur, sedangkan label merupakan *list* bilangan bulat. Pemisahan pada dataframe menggunakan method iloc dari dataframe. Method ini memerlukan 2 parameter. Parameter pertama adalah penanda baris yang akan diambil. Titik dua adalah tanda bahwa seluruh baris akan diambil. Parameter kedua adalah penanda kolom yang akan diambil. Pada variabel X, yaitu fitur, kolom yang akan digunakan adalah kolom pertama hingga kolom terakhir sebelum label. Sedangkan, variabel y adalah label, maka kolom yang digunakan hanyalah kolom terakhir. Gambar 4.6 adalah kode untuk melakukan pemisahan fitur dan label pada dataframe.

```
1  #pemisahan fitur dan label
2  X = df.iloc[:, 0:-1]
3  y = df.iloc[:, -1]
```

Gambar 4.6 Kode Pemisahan Fitur dan Label

4) *One-hot Encoding*

One-hot encoding bertujuan untuk mengubah label bilangan bulat menjadi list bilangan biner 22 anggota. Setiap anggota merepresentasikan label yang ada pada dataset, dengan nilai standar 0, dan satu buah label bernilai 1. Untuk melakukan one-hot encoding, dipanggil method to_categorical dengan parameter berupa variabel y yang menampung label dari dataset. Gambar 4.7 merupakan kode untuk melakukan one-hot encoding pada variabel y.

```
1  #one-hot encoding label
2  y = tf.keras.utils.to_categorical(y)
```

Gambar 4.7 Kode Pemisahan Fitur dan Label

5) Pembagian Data

Proses pembagian data yang dilakukan dalam proses penelitian adalah membagi data menjadi 3 macam pembagian. Skenario pertama yaitu 70% data latih, 15% data validasi, dan 15% data uji. Skenario kedua adalah pembagian 80% data latih, 10% data validasi, dan 10% data uji. Skenario terakhir adalah pembagian 90% data latih, 5% data validasi, dan 5% data uji. Dalam proses ini juga dilakukan pengacakan data dengan *seed random* bernilai 42 dengan bantuan *method* pada Sci-kit Learn, yaitu *Train Test Split*.

Pemisahan dilakukan sejumlah 2 kali, karena *Train Test Split* hanya dapat memisahkan 1 dataset menjadi 2 dataset. Maka dari itu, pemisahan pertama dilakukan untuk memisahkan data latih dengan data validasi dan data uji. Pemisahan kedua dilakukan untuk memisahkan data validasi dan data uji. Pemisahan dilakukan melalui 3 skenario seperti pada Tabel 3.2.

Gambar 4.8 merupakan potongan kode yang digunakan dalam melakukan proses pembagian data pada variabel X dan y dengan rasio pembagian data 90% data latih, 5% data validasi, dan 5% data uji.

```
1  #train-val-test split
2  X_train, X_val_test, y_train, y_val_test = train_test_split(X, y, test_size=0.1, shuffle=True, random_state=42)
3  X_val, X_test, y_val, y_test = train_test_split(X_val_test, y_val_test, test_size=0.5, shuffle=True, random_state=42)
```

Gambar 4.8 Kode Pembagian Data

6) Normalisasi *MinMax Scaler*

Proses MinMax Scaler bertujuan untuk mengurangi biaya komputasi yang dibutuhkan. Melalui proses ini, akurasi akan meningkat dan waktu yang dibutuhkan dalam proses pelatihan akan berkurang. Proses MinMax Scaler dilakukan melalui bantuan salah satu *method* pada Sci-kit Learn, yaitu *MinMax Scaler*.

Proses pertama adalah mendefinisikan MinMaxScaler. Proses berikutnya adalah melakukan fitting dari MinMaxScaler yang telah didefinisikan pada data latih menggunakan method fit. Setelah mendapatkan range data dari data latih,

maka proses normalisasi dilakukan pada data latih, data validasi, dan data uji. Gambar 4.9 merupakan potongan kode untuk melakukan proses MinMax Scaling.

```
1  #scaling untuk mengurangi range data
2  minmaxScaler = MinMaxScaler()
3  minmaxScaler.fit(X_train)
4
5  X_train = minmaxScaler.transform(X_train)
6  X_val = minmaxScaler.transform(X_val)
7  X_test = minmaxScaler.transform(X_test)
```

Gambar 4.9 Kode Normalisasi MinMax Scaling

Tabel 4.3 merupakan hasil dari 10 record teratas data latih hasil akhir tahap *preprocessing*. Data ini adalah data matang yang akan digunakan dalam proses pengembangan model machine learning.

Tabel 4.3 Hasil 10 Record Teratas Data Latih

Idx	**N**	**P**	**K**	**temperature**	**humidity**	**ph**	**rainfall**
1	0,171	0,443	0,085	0,324	0,101	0,329	0,303
2	0,186	0,229	0,125	0,763	0,411	0,414	0,277
3	0,086	0,157	0,105	0,582	0,365	0,283	0,273
4	0,150	0,107	0,130	0,481	0,997	0,366	0,523
5	0,414	0,329	0,210	0,956	0,903	0,506	0,637
6	0,193	0,950	0,980	0,426	0,929	0,381	0,306
7	0,157	0,521	0,355	0,259	0,056	0,796	0,202
8	0,157	0,029	0,195	0,456	0,870	0,348	0,330
9	0,286	0,379	0,350	0,280	0,006	0,570	0,249
10	0,714	0,064	0,220	0,593	0,896	0,445	0,021

4.1.3 Pengembangan Model ANN

Proses pengembangan model ANN dimulai dengan mendefinisikan *callback*. *Callback* dalam TensorFlow merupakan fungsi atau blok kode yang akan dieksekusi pada suatu kejadian secara spesifik ketika melakukan pelatihan model ANN. Digunakan dua *callback* dalam model yang akan dikembangkan, yaitu *callback Reminder* untuk memberi tanda pada akhir sebuah epoch dimana akurasi pada data latih dan data validasi mencapai 99%. Selain itu, dibuat sebuah *callback* kedua untuk menyimpan model yang mencapai *loss* terkecil pada data validasi. Gambar 4.10 merupakan potongan kode yang digunakan untuk mendefinisikan kedua *callback* yang merupakan turunan dari *Callback* TensorFlow. Pada baris 12, didefinisikan sebuah callback yang digunakan untuk menyimpan model terbaik pada proses pelatihan ini dalam bentuk H5. Sehingga, apabila pelatihan model mendapatkan akurasi validasi tertinggi selama proses pelatihan berlangsung, maka hasil tersebut akan disimpan.

```
1  #mendefinisikan kedua callback
2
3  class Reminder(tf.keras.callbacks.Callback):
4      def on_epoch_end(self, epoch, logs={}):
5          ACCURACY_THRESHOLD = 0.99
6      if(logs.get('val_accuracy') > ACCURACY_THRESHOLD and
           logs.get('accuracy') > ACCURACY_THRESHOLD):
7          print("\nTarget reached %2.2f%%."
   %(ACCURACY_THRESHOLD
                  100))
8
9  reminderCB = Reminder()
10
11 checkpoint_filepath = 'app/data/new_model/model.h5'
12 checkpointCB = tf.keras.callbacks.ModelCheckpoint(
13 filepath=checkpoint_filepath,
14                                      monitor='val_loss',
15                                      mode='min',
16                                      save_best_only=True)
```

Gambar 4.10 Kode Callback dalam TensorFlow

Terdapat sembilan model yang telah dikembangkan, yaitu model dengan 3 skenario perbedaan pada layer yang digunakan, yaitu 3 layer, 4 layer, dan 5 layer dengan rasio pembagian data pada tabel 3.3. Model yang akan digunakan adalah model dengan 4 layer. Pemilihan ini didasarkan pada akurasi dari total 9 model yang telah dikembangkan, dan penelitian terdahulu terkait klasifikasi multinomial yang membuktikan bahwa model dengan 4 layer memiliki akurasi tertinggi bila dibandingkan dengan 2 hingga 9 layer (Abu, 2018). Model yang digunakan terdiri dari 4 layer yang masing-masing terdiri dari 32 unit, 128 unit, dan 256 unit dengan fungsi aktivasi ReLU, dan diakhiri dengan 22 unit dengan fungsi aktivasi *softmax*. Alasan penggunaan layer ReLU adalah layer ReLU merupakan layer terbaik dalam proses pelatihan pada layer atas, yang dibuktikan pada penelitian terdahulu dengan permasalahan yang sama, yaitu klasifikasi multinomial (Ertam, 2017). Layer terakhir adalah layer output yang terdiri dari 22 unit dengan fungsi aktivasi *softmax*. Penelitian terdahulu terkait klasifikasi multinomial juga menggunakan fungsi aktivasi yang sama, yaitu *softmax.* Penggunaan fungsi aktivasi *softmax* didasarkan pada permasalahan yang membutuhkan solusi berupa klasifikasi multinomial, maka dibutuhkan lebih dari 2 label. Pemilihan optimizer berupa RMSProp didasarkan pada penelitian terdahulu oleh Chandra (2017) yang dikutip dari Schaul et. al (2014) menyatakan bahwa RMSProp mengungguli metode adaptif lain seperti Adagrad, Adadelta, bahkan SGD dengan momentum pada tes dalam jumlah besar. Gambar 4.11 merupakan kode yang digunakan untuk melakukan pemodelan.

```
1  #mendefinisikan model dan proses pelatihan model
2
3  model = tf.keras.Sequential([
4      tf.keras.layers.Dense(32, activation='relu'),
5      tf.keras.layers.Dense(128, activation='relu'),
6      tf.keras.layers.Dense(256, activation='relu'),
7      tf.keras.layers.Dense(22, activation='softmax')
8  ])
9
10 model.compile(loss='categorical_crossentropy',optimizer='rmsprop'
                ,metrics=['accuracy'])
```

```
11
12 fitting_history = model.fit(
13                         X_train,
14                         y_train,
15                         epochs=200,
16                         steps_per_epoch = 50,
17                         validation_data=(X_test, y_test),
18                         callbacks = [reminderCB,checkpointCB],
19                     )
```

Gambar 4.11 Kode Pendefinisian Model dan Proses Pelatihan

Contoh pada Gambar 4.11 dengan perbandingan 90% data latih, 5% data validasi, dan 5% data uji menghasilkan akurasi tertinggi dibandingkan dengan 8 skenario lain. Skenario ini menghasilkan akurasi latih sebesar 99.30%, akurasi validasi sebesar 99.24%, dan akurasi uji sebesar 98.93%. Hasil dari seluruh skenario dapat dilihat pada Tabel 4.25, dan Tabel 4.26.

4.1.4 Deployment

Model terbaik yang diperoleh dari pelatihan akan di-deploy dengan menggunakan micro-framework berbasis Python, yaitu Flask. Melalui penggunaan Flask, model dapat di-deploy menuju sebuah website sederhana untuk memudahkan proses penggunaan model oleh pengguna. Hasil dari proses deployment adalah website dengan 16 fitur yang mencakup sistem yang dibutuhkan, seperti halaman masuk, daftar, klasifikasi kondisi tanah berdasarkan nama tanaman, dan melihat hasil klasifikasi yang telah dilakukan. Berikut adalah penjelasan mengenai cara untuk mengembangkan dan hasil dari masing-masing halaman pada proses deployment.

a. Halaman beranda

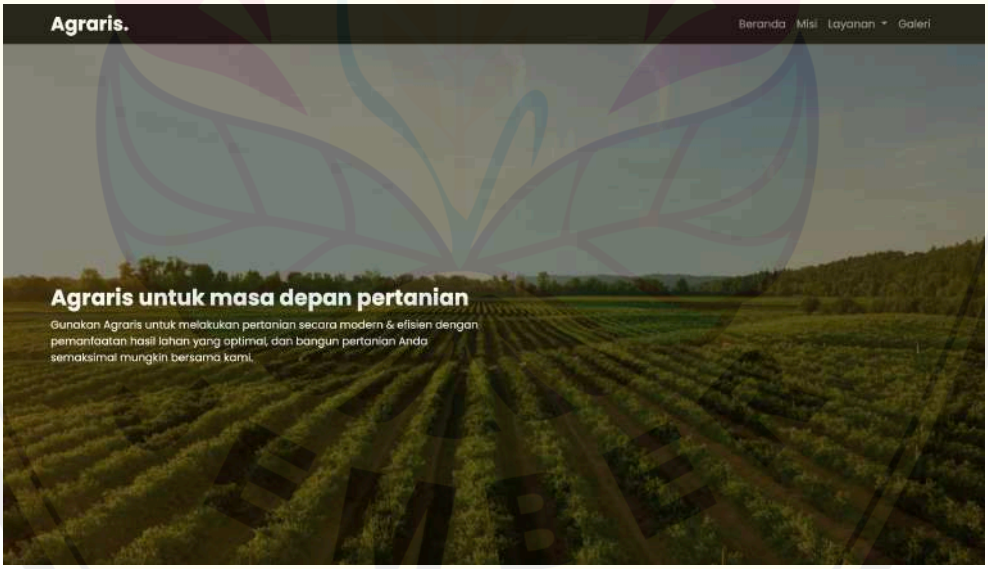
Pengembangan dari halaman ini membutuhkan beberapa library, yaitu Flask, dan App yang merupakan decorator dari Flask. Gambar 4.12 adalah kode yang digunakan untuk menampilkan Halaman Beranda. Fungsi method render_template adalah menampilkan halaman HTML dari file index.html yang berada pada templates/index.html pada base route, yaitu “/”. Decorator app juga

membutuhkan 1 parameter lagi, yaitu method. Apabila tidak dispesifikasikan, maka app akan memanggil method default, yaitu GET. Untuk menjalankan server dari Flask, digunakan method run.

```
1  # Import library yang dibutuhkan
2  from flask import render_template
3  from app import app
4
5
6  # Render halaman homepage
7  @app.route('/')
8  def home():
9    return render_template('index.html')
10
11 if __name__ == '__main__':
12   app.run()
```

Gambar 4.12 Kode untuk Menampilkan Halaman Beranda

Gambar 4.13 merupakan halaman beranda dari hasil deployment pada kode. Halaman beranda adalah halaman yang berisi informasi mengenai website yang dikembangkan, beserta layanan apa saja yang disediakan dalam website ini.



Gambar 4.13 Halaman Beranda

b. Halaman Form Pengisian Parameter Kondisi Tanah dan Hasil Klasifikasi

Pengembangan dari halaman ini membutuhkan library seperti Flask dan App yang merupakan decorator dari Flask. Gambar 4.14 adalah kode yang

digunakan untuk menampilkan Halaman Form Pengisian Parameter Kondisi Tanah. Fungsi method render_template adalah menampilkan halaman HTML dari file soil-classification.html yang berada pada templates/soil-classification.html pada route yang telah didefinisikan sebelumnya, yaitu "/klasifikasi-kondisi-tanah". Lalu, untuk menjalankan server dari Flask, digunakan method run.

```
@app.route('/klasifikasi-kondisi-tanah', methods=["GET", "POST"])
def soil_classification():
  if current_user.is_authenticated:
    if (current_user.role == "2"):
      form = SoilConditionForm()
      user_list = load_all_users()
      form.petani.choices = [(user.id, user.name) for user in user_list if user.role == "3" and user.ppl == current_user.id]
      if form.validate_on_submit():
        created_at = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        user_input = np.array([[form.natrium.data, form.phosphorus.data, form.kalium.data, form.temperature.data, form.humidity.data, form.ph.data, form.rainfall.data]])
        prediction = utils.classify_soil_condition(user_input)
        result = utils.recommendation_label(prediction)


        new_soil_condition = SoilConditions(user_id=form.petani.data, natrium=form.natrium.data, phosphorus=form.phosphorus.data, kalium=form.kalium.data, temperature=form.temperature.data, humidity=form.humidity.data, ph=form.ph.data, rainfall=form.rainfall.data, created_at=created_at, result=result)
        db.session.add(new_soil_condition)
        db.session.commit()

        last_id = SoilConditions.query.order_by(desc(SoilConditions.id)).first().id

        return redirect(url_for('classification_report_detail_user', report_id=last_id))
      return render_template('soil-classification.html', form=form)
    else:
```

```
        return redirect(url_for('landing_page'))
    else:
        return redirect(url_for('login'))

```

Gambar 4.14 Kode untuk Halaman Form Pengisian Parameter Kondisi Tanah

Halaman form pengisian parameter kondisi tanah merupakan halaman untuk memasukkan parameter dari pengguna berupa kondisi tanah secara fisik dan kimiawi yang dirumuskan dalam 7 parameter, yaitu Natrium (N), Potassium (P), Kalium (K), temperatur (temperature), kelembapan (humidity), pH, dan curah hujan (rainfall). Gambar 4.15 merupakan kode form pengisian parameter kondisi tanah sebagai masukan dari pengguna untuk menggunakan model yang telah dikembangkan. Halaman ini hanya dapat diakses apabila pengguna masuk sebagai PPL. Apabila pengguna yang masuk bukan PPL, maka pengguna tersebut akan diarahkan menuju halaman utama. Apabila pengguna belum masuk, maka pengguna tersebut akan diarahkan menuju halaman untuk masuk. Pengguna yang telah mengisi form dengan benar dan menekan tombol “Lakukan Klasifikasi” akan menuju proses klasifikasi.

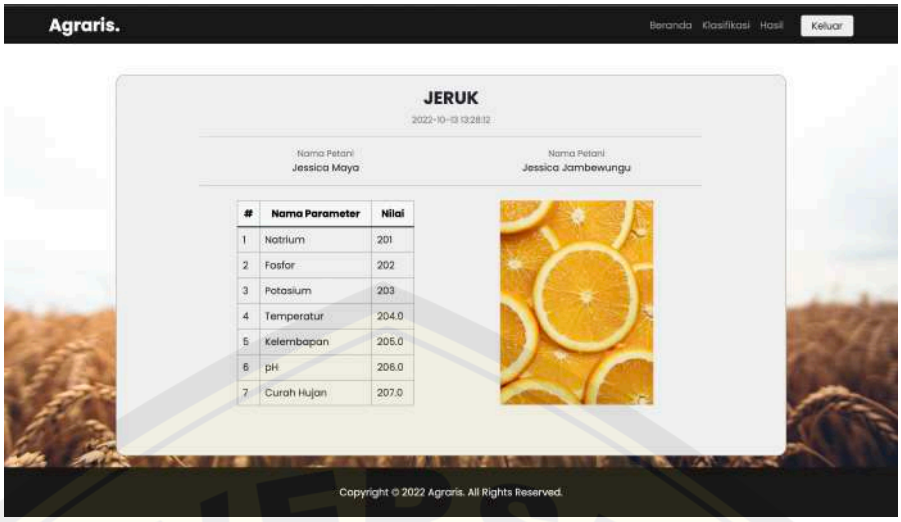
```
def classify_soil_condition(array_param):
    path = ("app/data/new_model/model.h5")
    X_predict = normalization(array_param)
    model = tf.keras.models.load_model(path)
    prediction = model.predict(X_predict)
    return prediction

def normalization(array_parameter):
  min_val = [0, 5, 5, 8.825675, 14.258040, 3.504752,
20.211267]
  max_val = [140, 145, 205, 43.675493, 99.981876, 9.935091,
298.560117]

def recommendation_label(prediction):
  result = np.argmax(prediction)

  with
open('app/data/labels/dict_crop_recommendation.json', 'r')
as file:
    labels = json.load(file)

```

```
18    for key, value in labels.items():
19                                if str(result) == value:
20        return key
```

Gambar 4.15 Kode dan Fungsi untuk Melakukan Klasifikasi Kondisi Tanah

Hal pertama yang akan dilakukan adalah mencatat waktu dilakukannya klasifikasi. Lalu, masukan dari pengguna berupa 7 kondisi tanah akan dikumpulkan ke dalam sebuah NumPy Array. Kumpulan masukan ini akan menjadi masukan untuk dilakukannya proses klasifikasi pada Gambar 4.15 yang dihubungkan melalui fungsi classify_soil_condition(). Pada fungsi classify soil condition, NumPy Array akan dinormalisasikan menggunakan MinMax Scaling. Nilai minimal dan maksimal dalam tiap parameter telah disatukan dalam variabel min_val dan max_val untuk mengurangi biaya komputasi yang diperlukan. Setiap nilai pada array masukan akan diiterasikan untuk melakukan normalisasi pada data. Prediksi dilakukan setelah proses normalisasi. Hasil prediksi yang masih berupa array dari confidence model akan diproses menggunakan method argmax dari NumPy untuk mendapatkan 1 label angka yang merupakan index dengan nilai confidence tertinggi.

Fungsi recommendation_label() adalah fungsi untuk melakukan reverse label encoding. Fungsi ini dilakukan pada hasil prediksi model untuk mendapatkan hasil akhir berupa kata-kata yang mewakili tanaman tersebut. Hasil dari klasifikasi akan disimpan dalam database, sehingga hasil klasifikasi dapat dilihat oleh pengguna lainnya yang berwenang. Setelah klasifikasi selesai, hasil ini akan dikembalikan dalam halaman detail hasil klasifikasi. Halaman ini meminta sebuah parameter berupa hasil klasifikasi terbaru dari database. Gambar 4.16 merupakan gambar contoh hasil keluaran dari keseluruhan proses klasifikasi untuk menampilkan tanaman terbaik hasil dari klasifikasi kondisi tanah.



Gambar 4.16 Halaman Rekomendasi Tanaman

4.1.5 Testing

Terdapat 2 macam testing yang dilakukan. Testing pertama dilakukan pada proses pengembangan model untuk memastikan bahwa proses pengembangan model berlangsung dengan baik. Test case yang diambil pada ekstraksi data hingga proses pengembangan model yang mengacu pada Bagian 4.1.1 hingga bagian 4.1.3. Tabel 4.4 hingga Tabel 4.9 adalah hasil testing yang dilakukan pada pengembangan model.

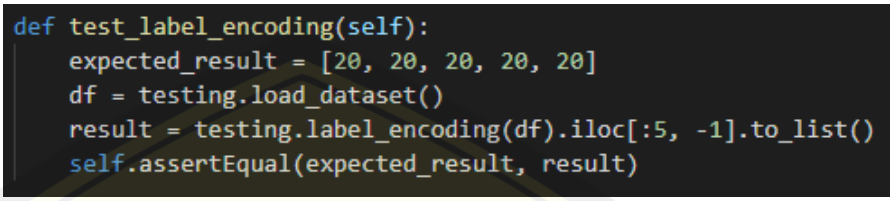
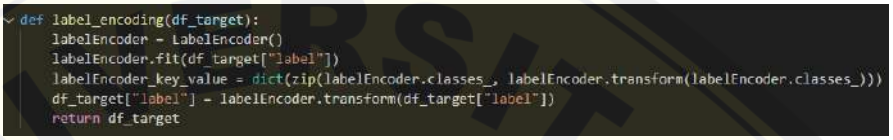
Tabel 4.4 Whitebox Testing Fitur Ekstraksi Data

ID	WX-001
Fitur	Ekstraksi Data
Pre-Condition	-
Test Step	1. Menyiapkan dataset pada dataset/crop_recommendation.csv 2. Import library yang dibutuhkan (Pandas) 3. Mendefinisikan dataframe dengan dataset Crop Recommendation 4. Mendefinisikan shape dari dataframe yang diharapkan, yaitu (2200, 8)

	5. Melakukan pengecekan melalui method AssertEqual dari library Unit Test untuk mengetahui kesamaan antara shape dari dataframe dengan shape yang diharapkan
Ekspektasi	Shape dari dataframe sama dengan shape yang diharapkan, yaitu (2200, 8)
Rules	Dataset berada pada path yang tepat
Status	BERHASIL
Keterangan	`class AgrarisTesting(unittest.TestCase):` `  def test_dataset_load(self):` `  expected_length = (2200, 8)` `  result = testing.load_dataset().shape` `  self.assertEqual(expected_length, result)` `def load_dataset():` `  file_name = "dataset/Crop_recommendation.csv"` `  df = pd.read_csv(file_name)` `  return df`

Tabel 4.5 Whitebox Testing Fitur Label Encoding

ID	WX-002
Fitur	Label Encoding
Pre-condition	Telah melakukan ekstraksi data
Test Step	1. Mendefinisikan LabelEncoder dari library Sci-kit Learn 2. Melakukan fitting dari LabelEncoder pada dataframe 3. Mencatat hasil dari LabelEncoder 4. Melakukan label encoding pada dataframe sesuai dengan hasil fitting 5. Mencatat hasil yang diharapkan dari 5 baris data teratas dataframe, yaitu [20, 20, 20, 20, 20] 6. Melakukan pengecekan melalui AssertEqual dari library Unit Test untuk mengetahui kesamaan antara 5 baris data pada hasil Label Encoding dari dataframe dengan hasil yang diharapkan

Ekspektasi	Hasil label encoding pada 5 baris teratas dari dataframe sama dengan hasil yang diharapkan, yaitu [20, 20, 20, 20, 20]
Rules	Dataset berada pada path yang tepat
Status	BERHASIL
Keterangan	 

```
def test_label_encoding(self):
    expected_result = [20, 20, 20, 20, 20]
    df = testing.load_dataset()
    result = testing.label_encoding(df).iloc[:5, -1].to_list()
    self.assertEqual(expected_result, result)
```

```
def label_encoding(df_target):
    labelEncoder = LabelEncoder()
    labelEncoder.fit(df_target["label"])
    labelEncoder_key_value = dict(zip(labelEncoder.classes_, labelEncoder.transform(labelEncoder.classes_)))
    df_target["label"] = labelEncoder.transform(df_target["label"])
    return df_target
```

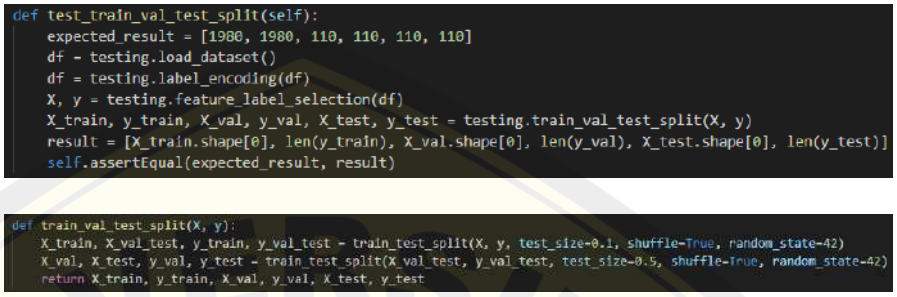
Tabel 4.6 Whitebox Testing Fitur Pemisahan Fitur dan Label

ID	WX-003
Fitur	Pemisahan Fitur dan Label
Pre-Condition	Telah melakukan Label Encoding
Test Step	1. Mendefinisikan dataframe X yang merupakan fitur dari dataset 2. Mendefinisikan Series y yang merupakan label dari dataset 3. Melakukan one-hot encoding pada Series y 4. Mendefinisikan hasil yang diharapkan yaitu panjang dari tiap baris data dalam X dan y, yaitu [7, 22] 5. Melakukan pengecekan melalui AssertEqual dari library Unit Test untuk mengetahui kesamaan antara panjang tiap baris data pada hasil pemisahan fitur dan label dengan panjang yang diharapkan
Ekspektasi	Panjang dari tiap baris data pada hasil pemisahan fitur dan label dari dataframe sama dengan panjang yang diharapkan, yaitu [7, 22]
Rules	Dataset berada pada path yang tepat
Status	BERHASIL

Keterangan	``` def test_feature_label_selection(self): expected_result = [7, 22] df = testing.load_dataset() df = testing.label_encoding(df) X, y = testing.feature_label_selection(df) result = [len(X.columns), len(y[0])] self.assertEqual(expected_result, result) ``` ``` def feature_label_selection(df): X = df.iloc[:, 0:-1] y = df.iloc[:, -1] y = tf.keras.utils.to_categorical(y) return X, y ```

Tabel 4.7 Whitebox Testing Fitur Pemisahan Data Latih, Validasi dan Uji

ID	WX-004
Fitur	Pemisahan Data Latih, Data Validasi, dan Data Uji
Pre-Condition	Telah melakukan pemisahan fitur dan label
Test Step	1. Mendefinisikan X latih, X validasi dan uji, y latih, dan y validasi dan uji 2. Membagi data dengan proporsi 90% data latih, 10% data validasi dan uji 3. Mendefinisikan X validasi, X uji, y validasi, dan y uji 4. Memisahkan data dengan proporsi 50% data validasi, 50% data uji 5. Mendefinisikan hasil pemisahan yang diharapkan pada X latih, y latih, X validasi, y validasi, X uji, dan y uji, yaitu [1980, 1980, 110, 110, 110, 110] 6. Melakukan pengecekan melalui AssertEqual dari library Unit Test untuk mengetahui kesamaan antara jumlah hasil pemisahan setiap fitur dan label dengan hasil pemisahan yang diharapkan

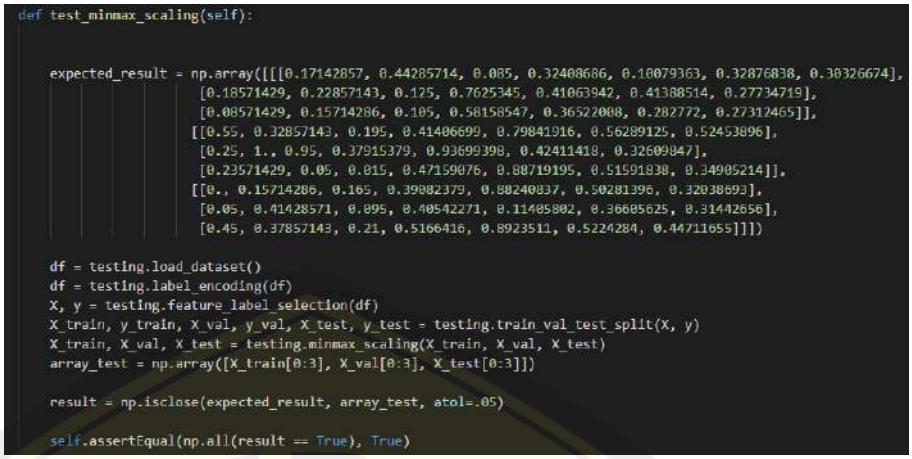
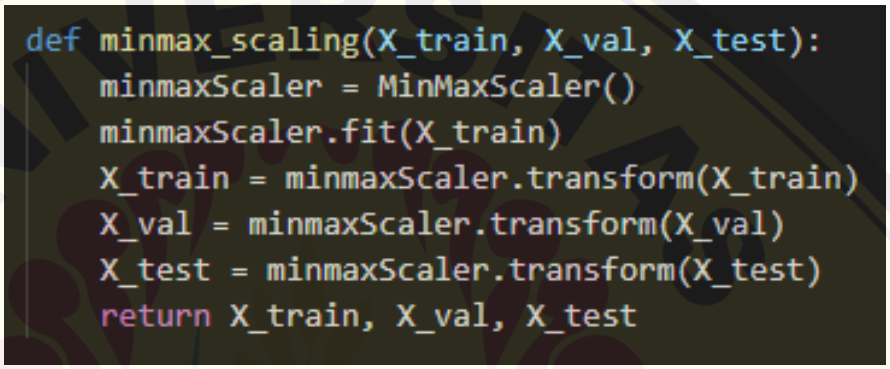
Ekspektasi	Jumlah hasil pemisahan tiap fitur dan label sama dengan hasil pemisahan yang diharapkan
Rules	Dataset berada pada path yang tepat
Status	BERHASIL
Keterangan	

```
def test_train_val_test_split(self):
    expected_result = [1980, 1980, 110, 110, 110, 110]
    df = testing.load_dataset()
    df = testing.label_encoding(df)
    X, y = testing.feature_label_selection(df)
    X_train, y_train, X_val, y_val, X_test, y_test = testing.train_val_test_split(X, y)
    result = [X_train.shape[0], len(y_train), X_val.shape[0], len(y_val), X_test.shape[0], len(y_test)]
    self.assertEqual(expected_result, result)
```

```
def train_val_test_split(X, y):
    X_train, X_val_test, y_train, y_val_test = train_test_split(X, y, test_size=0.1, shuffle=True, random_state=42)
    X_val, X_test, y_val, y_test = train_test_split(X_val_test, y_val_test, test_size=0.5, shuffle=True, random_state=42)
    return X_train, y_train, X_val, y_val, X_test, y_test
```

Tabel 4.8 Whitebox Testing Fitur MinMax Scaling

ID	WX-005
Fitur	MinMax Scaling
Pre-Condition	Telah melakukan pemisahan data latih, data validasi, dan data uji
Test Step	1. Mendefinisikan MinMaxScaling dari library Sci-kit Learn 2. Melakukan fitting dari MinMaxScaling pada data latih 3. Melakukan MinMax scaling pada fitur latih, fitur validasi, dan fitur uji sesuai dengan hasil fitting (label tidak di-scaling) 4. Mencatat hasil yang diharapkan dari 3 baris data teratas fitur latih, fitur validasi, dan fitur uji. 5. Melakukan pengecekan melalui AssertEqual dari library Unit Test untuk mengetahui kemiripan dengan toleransi 0.05 antara 3 baris data pada fitur latih, fitur validasi, dan fitur latih, fitur validasi, dan fitur uji hasil MinMax scaling dari fitur latih, fitur validasi, dan fitur uji dengan hasil yang diharapkan
Ekspektasi	Hasil 3 baris teratas fitur latih, fitur validasi, dan fitur uji MinMax scaling yang dilakukan memiliki kemiripan dengan hasil yang diharapkan denga batas toleransi 0.05
Rules	Dataset berada pada path yang tepat
Status	BERHASIL

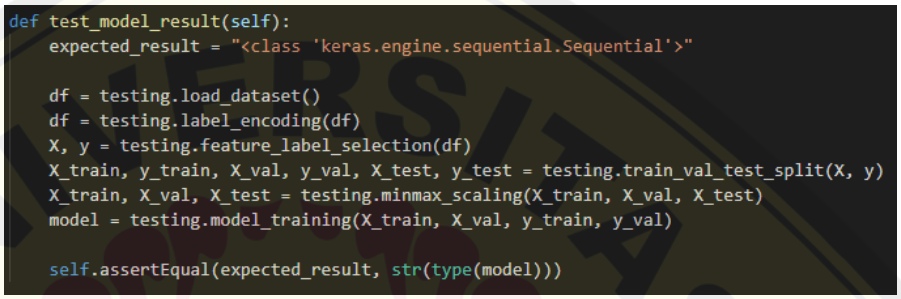
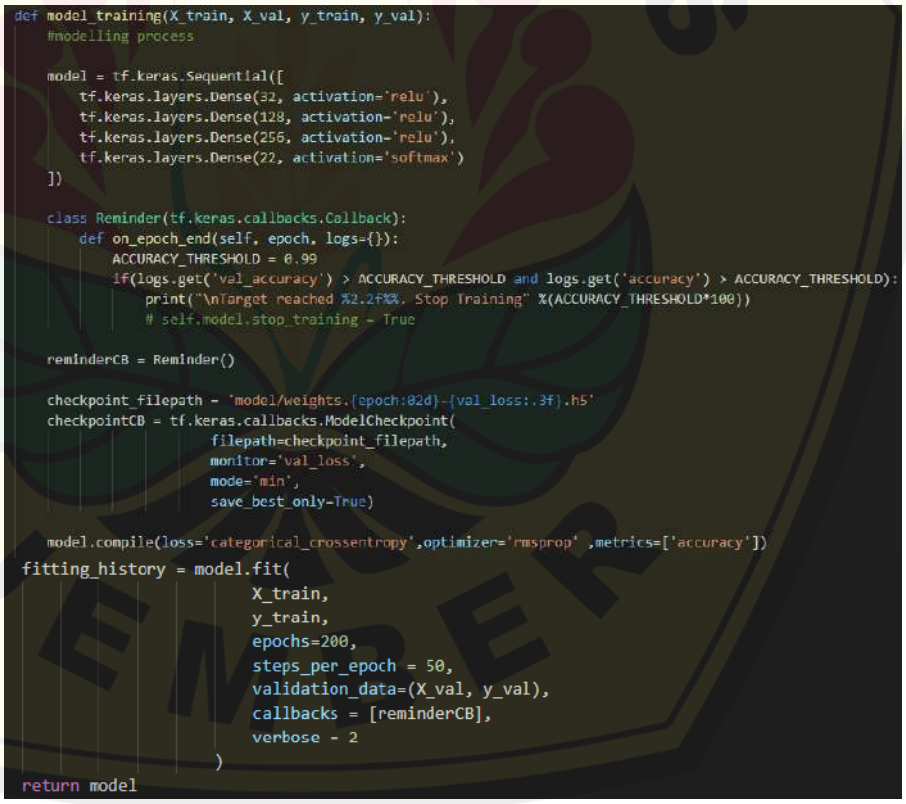
Keterangan	 

```
def test_minmax_scaling(self):

    expected_result = np.array([[[0.17142857, 0.44285714, 0.085, 0.32408686, 0.10079363, 0.32876838, 0.30326674],
                                 [0.18571429, 0.22857143, 0.125, 0.7625345, 0.41063942, 0.41388514, 0.27734719],
                                 [0.08571429, 0.15714286, 0.105, 0.58158547, 0.36522088, 0.282772, 0.27312465]],
                                [[0.55, 0.32857143, 0.195, 0.41406699, 0.79841916, 0.56289125, 0.52453896],
                                 [0.25, 1., 0.95, 0.37915379, 0.93699398, 0.42411418, 0.32609847],
                                 [0.23571429, 0.05, 0.015, 0.47159076, 0.88719195, 0.51591838, 0.34905214]],
                                [[0., 0.15714286, 0.165, 0.39082379, 0.88240837, 0.50281396, 0.32038693],
                                 [0.05, 0.41428571, 0.095, 0.40542271, 0.11405802, 0.38605625, 0.31442656],
                                 [0.45, 0.37857143, 0.21, 0.5166416, 0.8923511, 0.5224284, 0.44711655]]])

    df = testing.load_dataset()
    df = testing.label_encoding(df)
    X, y = testing.feature_label_selection(df)
    X_train, y_train, X_val, y_val, X_test, y_test = testing.train_val_test_split(X, y)
    X_train, X_val, X_test = testing.minmax_scaling(X_train, X_val, X_test)
    array_test = np.array([X_train[0:3], X_val[0:3], X_test[0:3]])

    result = np.isclose(expected_result, array_test, atol=.05)

    self.assertEqual(np.all(result == True), True)
```

```
def minmax_scaling(X_train, X_val, X_test):
    minmaxScaler = MinMaxScaler()
    minmaxScaler.fit(X_train)
    X_train = minmaxScaler.transform(X_train)
    X_val = minmaxScaler.transform(X_val)
    X_test = minmaxScaler.transform(X_test)
    return X_train, X_val, X_test
```

Tabel 4.9 Whitebox Testing Fitur Pengembangan Model ANN

ID	WX-006
Fitur	Pengembangan Model ANN
Pre-Condition	Telah melakukan MinMax Scaling
Test Step	1. Mendefinisikan model dan arsitektur model yang akan dikembangkan 2. Mendefinisikan Callback yang dibutuhkan 3. Mendefinisikan checkpoint dari model terbaik 4. Melakukan proses compile dari model 5. Proses pelatihan model 6. Mendefinisikan tipe data ciri khas dari model TensorFlow, yaitu <class 'keras.engine.sequential.Sequential'>

	7. Melakukan pengecekan melalui AssertEqual dari library Unit Test untuk mengetahui kesamaan antara tipe data dari model yang telah dikembangkan dengan tipe data yang diharapkan
Ekspektasi	Tipe data dari model yang telah dikembangkan sama dengan tipe data yang diharapkan
Rules	• Dataset berada pada path yang tepat • Model telah dilatih terlebih dahulu hingga selesai
Status	BERHASIL
Keterangan	 

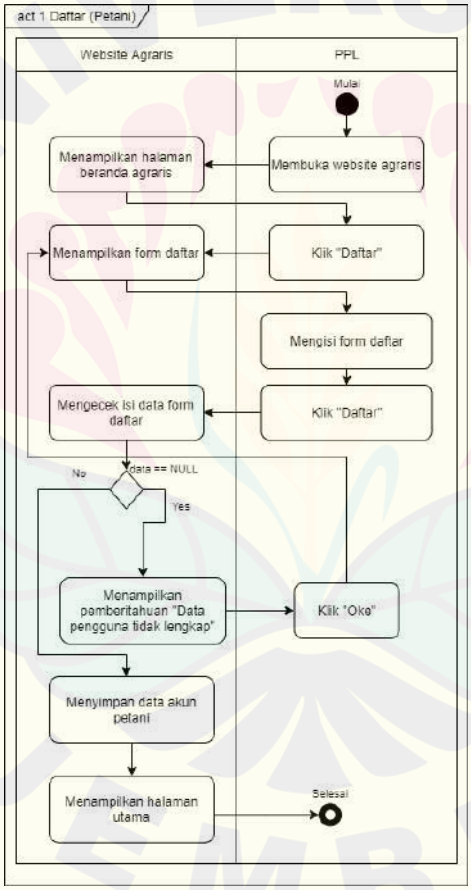
```
def test_model_result(self):
    expected_result = "<class 'keras.engine.sequential.Sequential'>"

    df = testing.load_dataset()
    df = testing.label_encoding(df)
    X, y = testing.feature_label_selection(df)
    X_train, y_train, X_val, y_val, X_test, y_test = testing.train_val_test_split(X, y)
    X_train, X_val, X_test = testing.minmax_scaling(X_train, X_val, X_test)
    model = testing.model_training(X_train, X_val, y_train, y_val)

    self.assertEqual(expected_result, str(type(model)))
```

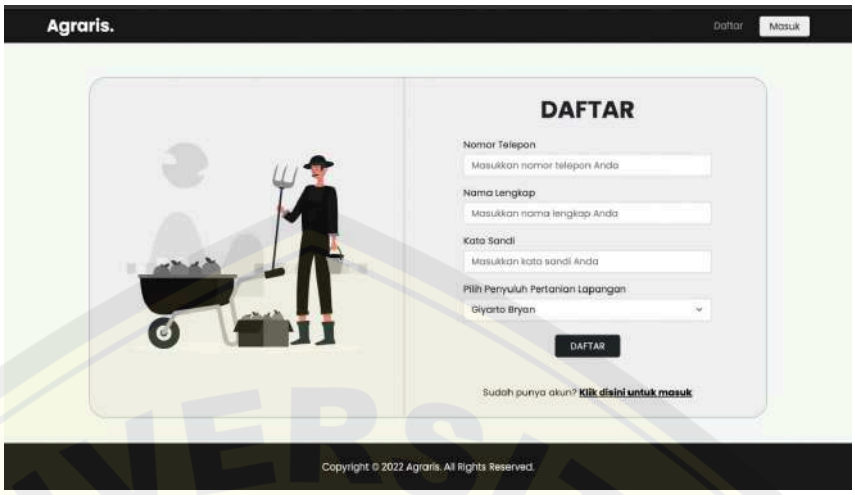
```
def model_training(X_train, X_val, y_train, y_val):
    #modelling process

    model = tf.keras.Sequential([
        tf.keras.layers.Dense(32, activation='relu'),
        tf.keras.layers.Dense(128, activation='relu'),
        tf.keras.layers.Dense(256, activation='relu'),
        tf.keras.layers.Dense(22, activation='softmax')
    ])

    class Reminder(tf.keras.callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            ACCURACY_THRESHOLD = 0.99
            if(logs.get('val_accuracy') > ACCURACY_THRESHOLD and logs.get('accuracy') > ACCURACY_THRESHOLD):
                print("\nTarget reached %2.2f%%. Stop Training" %(ACCURACY_THRESHOLD*100))
                # self.model.stop_training = True

    reminderCB = Reminder()

    checkpoint_filepath = 'model/weights.{epoch:02d}-{val_loss:.3f}.h5'
    checkpointCB = tf.keras.callbacks.ModelCheckpoint(
                    filepath=checkpoint_filepath,
                    monitor='val_loss',
                    mode='min',
                    save_best_only=True)

    model.compile(loss='categorical_crossentropy',optimizer='rmsprop' ,metrics=['accuracy'])
fitting_history = model.fit(
                        X_train,
                        y_train,
                        epochs=200,
                        steps_per_epoch = 50,
                        validation_data=(X_val, y_val),
                        callbacks = [reminderCB],
                        verbose = 2
                    )
return model
```

Testing kedua dilakukan pada website hasil deployment, dimana terdapat 3 halaman yang telah dikembangkan. Tampilan ini merupakan halaman beranda,

Halaman Form Pengisian Parameter Kondisi Tanah, dan Halaman Form Hasil Pengisian Parameter Kondisi Tanah. Test case dari testing kedua mengacu pada Bagian 4.1.4. Tabel 4.10 hingga Tabel 4.24 adalah hasil dari proses testing dari 3 halaman hasil proses deployment.

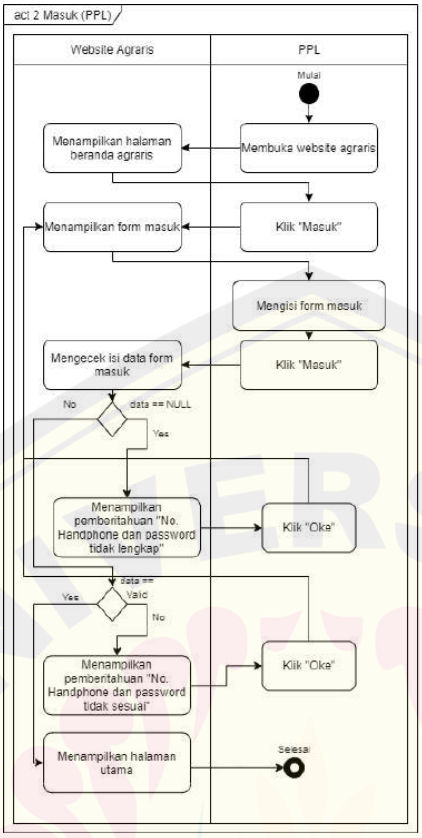
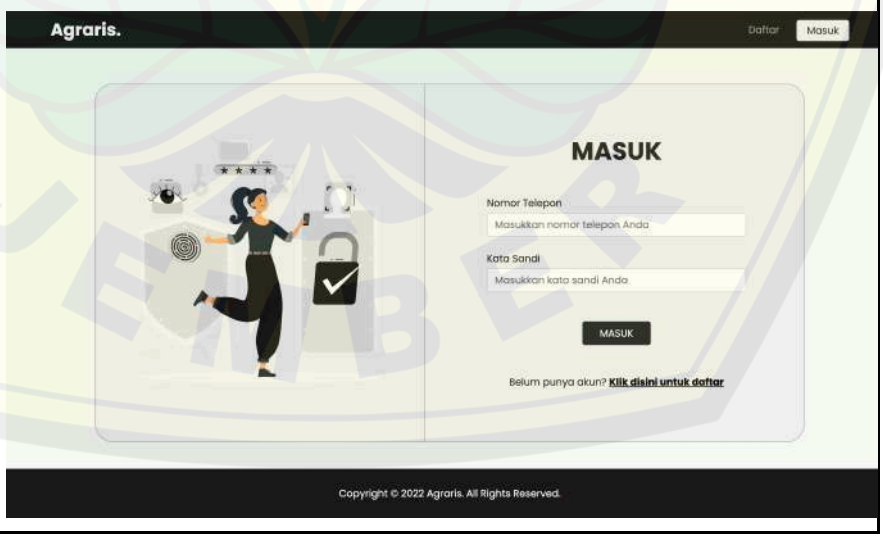
Tabel 4.10 Blackbox Testing Fitur Daftar (Petani)

ID	BX-001
Fitur	Daftar
User	Petani
Test Step	
Ekspektasi	User berhasil diarahkan menuju Halaman Beranda dari website
Rules	• Nomor telepon harus diawali dengan 62 • Nama lengkap harus diisi setidaknya 4 huruf • Password harus diisi setidaknya 8 karakter

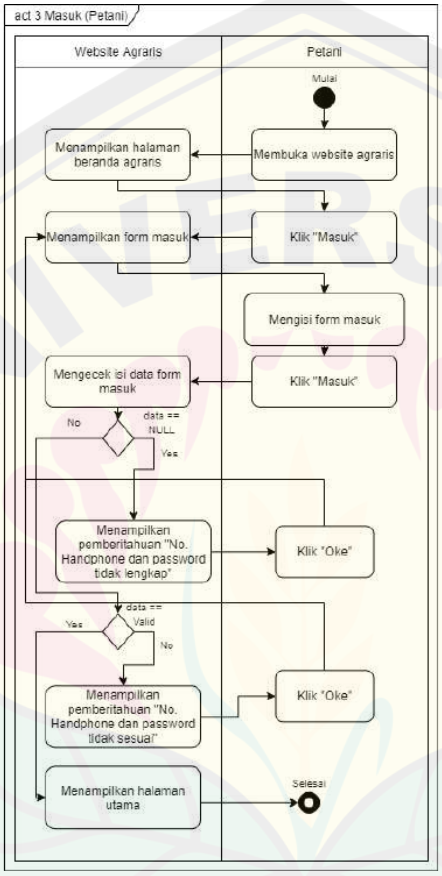
Status	BERHASIL
Keterangan	

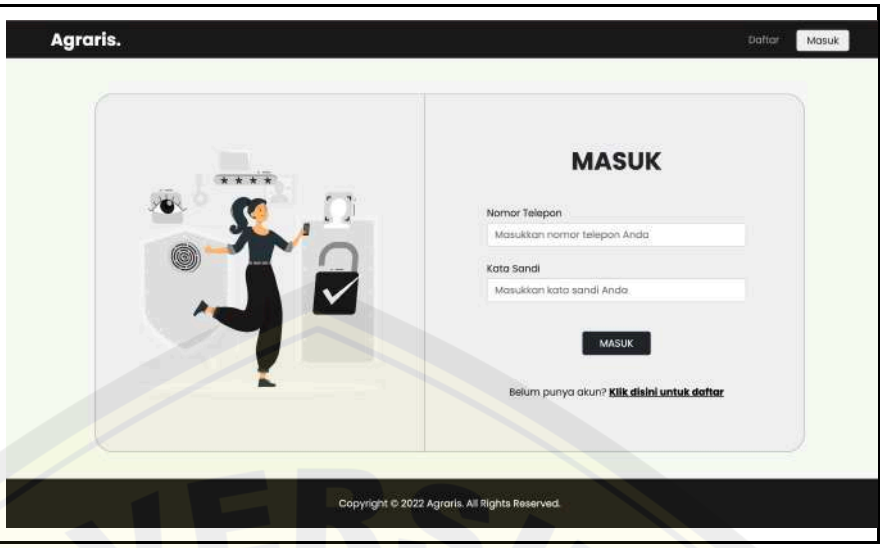
Tabel 4.11 Blackbox Testing Fitur Masuk (PPL)

ID	BX-002
Fitur	Masuk
User	PPL

Test Step	
Ekspektasi	User berhasil masuk dan diarahkan ke halaman utama Agraris
Rules	• Nomor telepon harus diawali dengan 62 • Password harus diisi setidaknya 8 karakter
Status	BERHASIL
Keterangan	

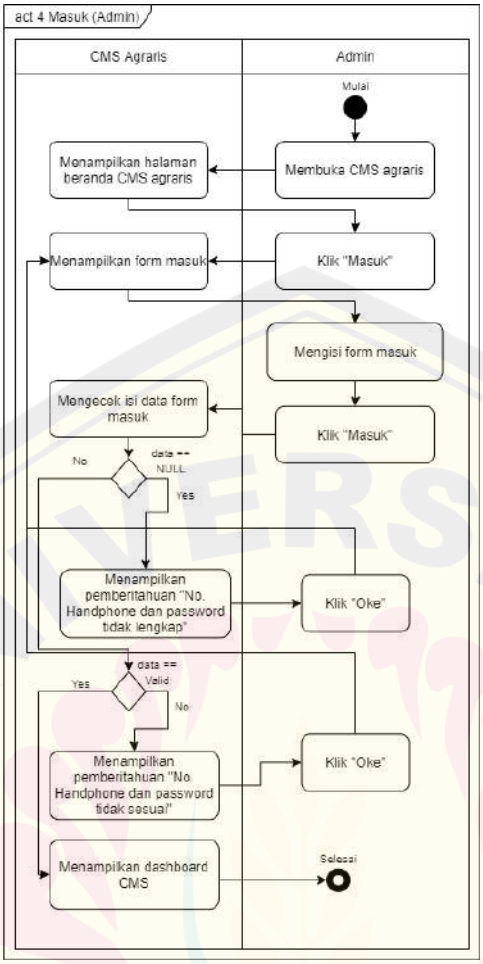
Tabel 4.12 Blackbox Testing Fitur Masuk (Petani)

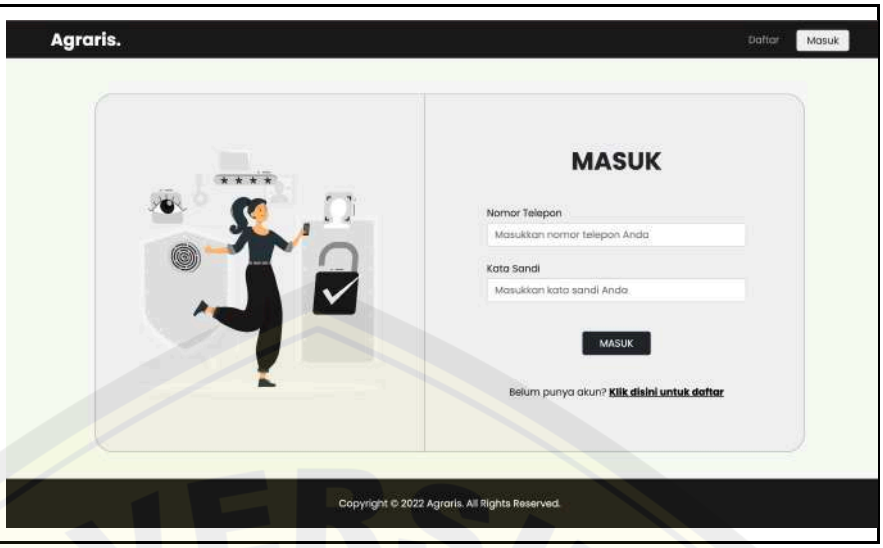
ID	BX-003
Fitur	Masuk
User	Petani
Test Step	
Ekspektasi	User berhasil masuk dan diarahkan ke halaman utama Agraris
Rules	• User harus memiliki akses internet • Nomor telepon harus diawali dengan 62 • Password harus diisi setidaknya 8 karakter
Status	BERHASIL

Keterangan	

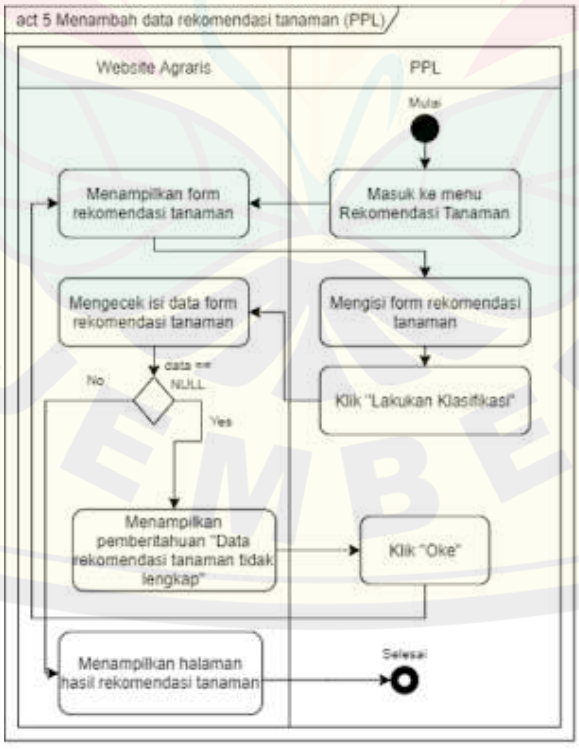
Tabel 4.13 Blackbox Testing Fitur Masuk (Admin)

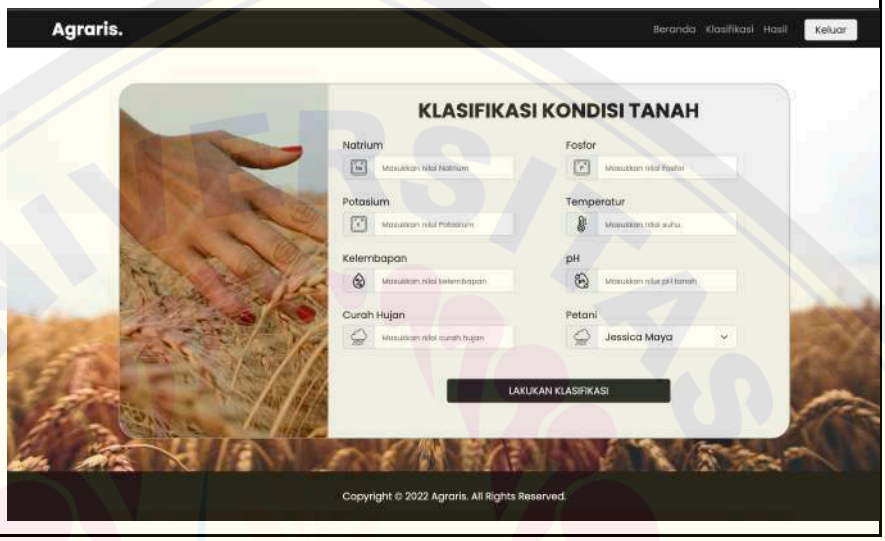
ID	BX-004
Fitur	Masuk
User	Admin

Test Step	
Ekspektasi	User berhasil masuk dan diarahkan ke dashboard CMS Agraris
Rules	• User harus memiliki akses internet • Nomor telepon harus diawali dengan 62 • Password harus diisi setidaknya 8 karakter
Status	BERHASIL

Keterangan	

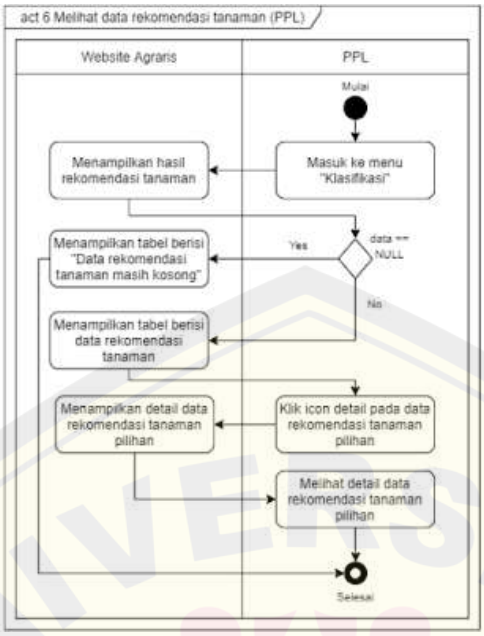
Tabel 4.14 Blackbox Testing Fitur Menambah Data Rekomendasi Tanaman (PPL)

ID	BX-005
Fitur	Menambah Data Rekomendasi Tanaman
User	PPL
Test Step	

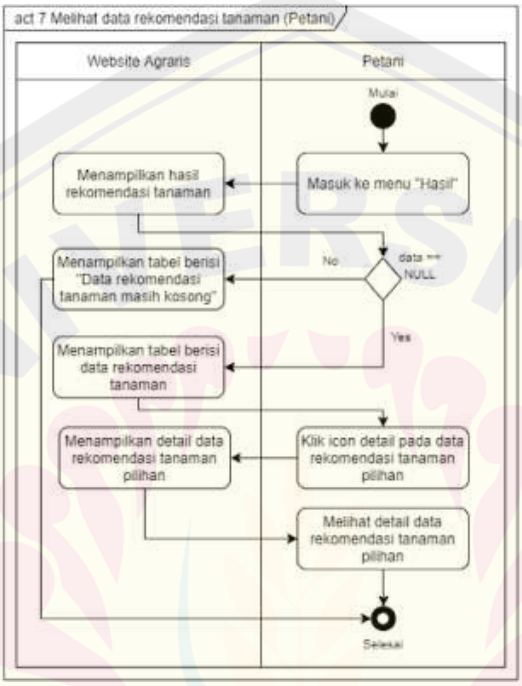
Ekspektasi	Data Hasil Rekomendasi Tanaman berhasil ditambahkan
Rules	• Seluruh field wajib diisi dengan benar • Isi field Natrium, Fosfor, Kalium wajib bernilai bilangan bulat • Isi field Temperatur, Humiditas, pH, Curah Hujan diperbolehkan desimal
Status	BERHASIL
Keterangan	

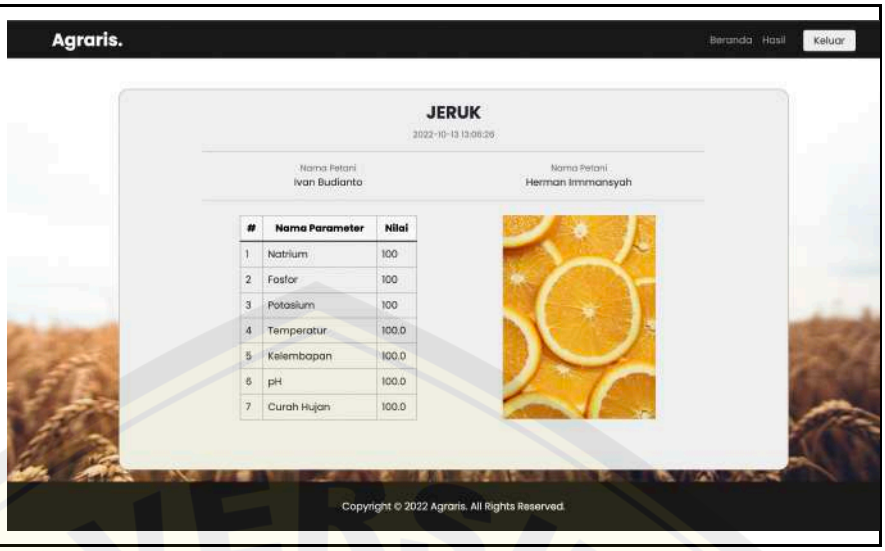
Tabel 4.15 Blackbox Testing Fitur Melihat Data Rekomendasi Tanaman (PPL)

ID	BX-006
Fitur	Melihat Data Rekomendasi Tanaman
User	PPL

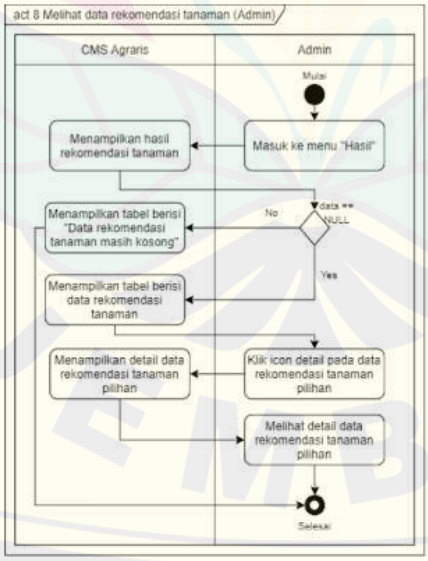
Test Step	
Ekspektasi	Data Hasil Rekomendasi Tanaman berhasil dilihat
Rules	• Data hasil rekomendasi tanaman telah ditambahkan sebelumnya oleh PPL
Status	BERHASIL
Keterangan	

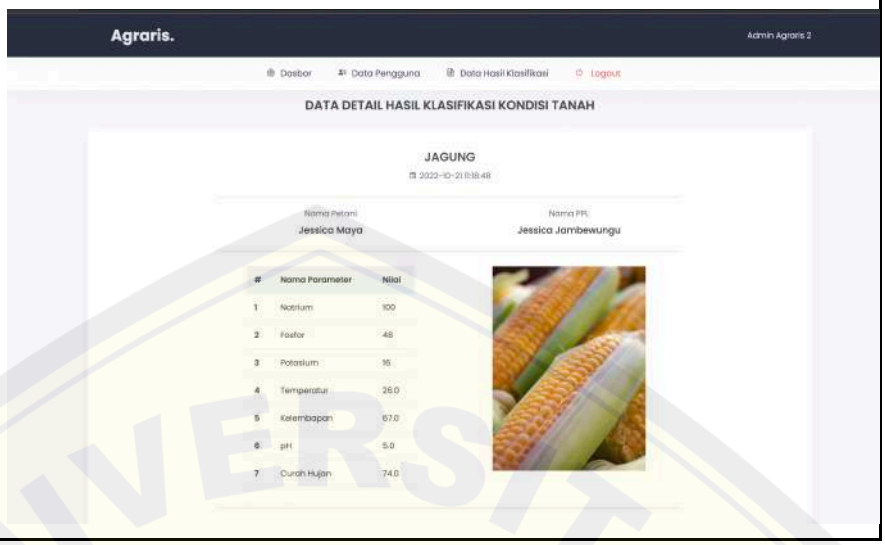
Tabel 4.16 Blackbox Testing Fitur Melihat Data Rekomendasi Tanaman (Petani)

ID	BX-007
Fitur	Melihat Data Rekomendasi Tanaman
User	Petani
Test Step	
Ekspektasi	Data Hasil Rekomendasi Tanaman berhasil dilihat
Rules	• Data hasil rekomendasi tanaman telah ditambahkan sebelumnya oleh PPL
Status	BERHASIL

Keterangan	

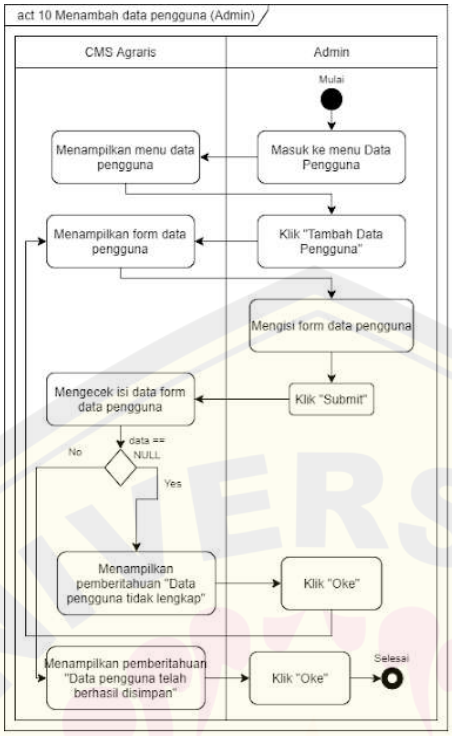
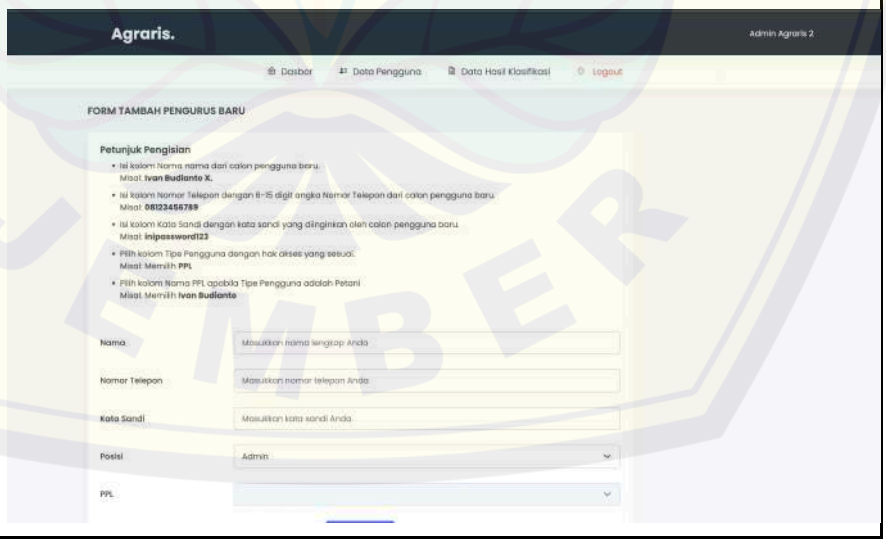
Tabel 4.17 Blackbox Testing Fitur Melihat Data Rekomendasi Tanaman (Admin)

ID	BX-008
Fitur	Melihat Data Rekomendasi Tanaman
User	Admin
Test Step	
Ekspektasi	Data Hasil Rekomendasi Tanaman berhasil dilihat
Rules	• Data hasil rekomendasi tanaman telah ditambahkan sebelumnya oleh PPL

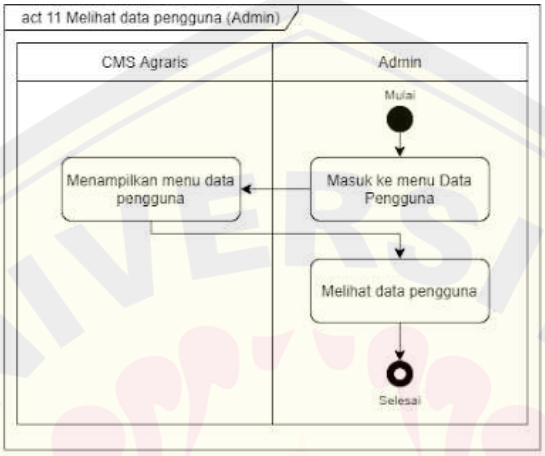
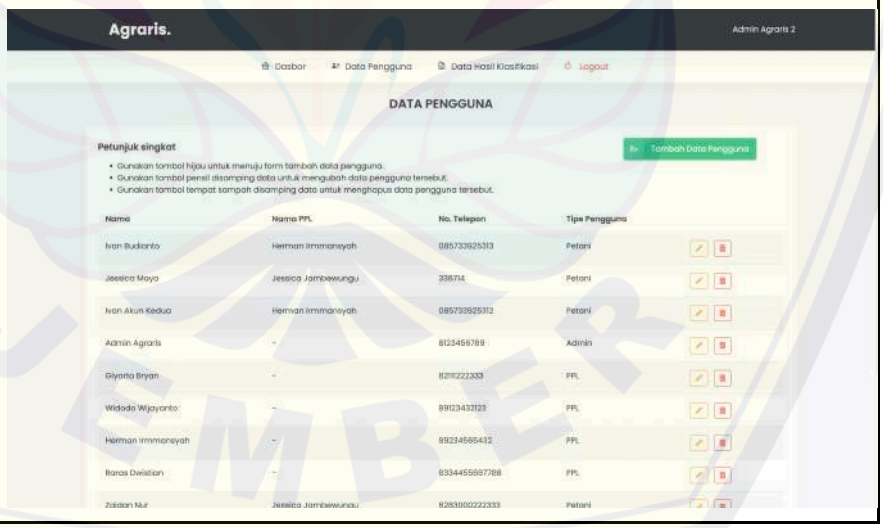
Status	BERHASIL
Keterangan	

Tabel 4.18 Blackbox Testing Menambah Data Pengguna (Admin)

ID	BX-009
Fitur	Menambah Data Pengguna
User	Admin

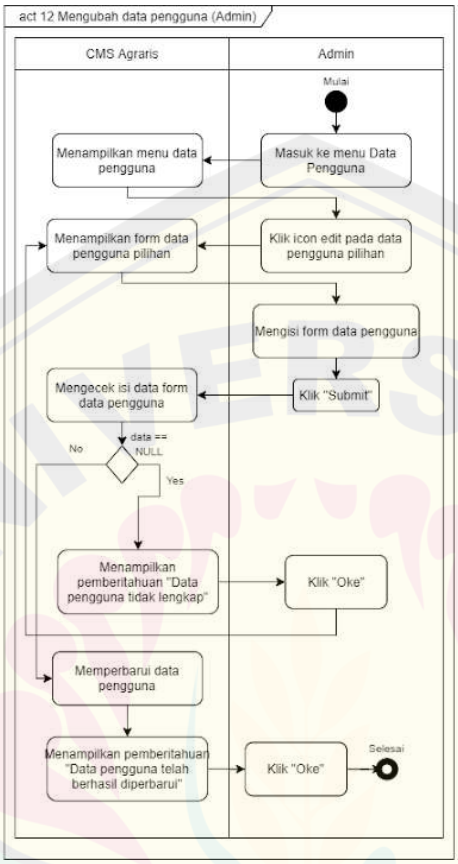
Test Step	
Ekspektasi	Data Pengguna berhasil ditambah
Rules	• Nomor telepon harus diawali dengan 62 • Nama lengkap harus diisi setidaknya 4 huruf • Password harus diisi setidaknya 8 karakter • Kategori Admin dan PPL tidak perlu mengisi field PPL
Status	BERHASIL
Keterangan	

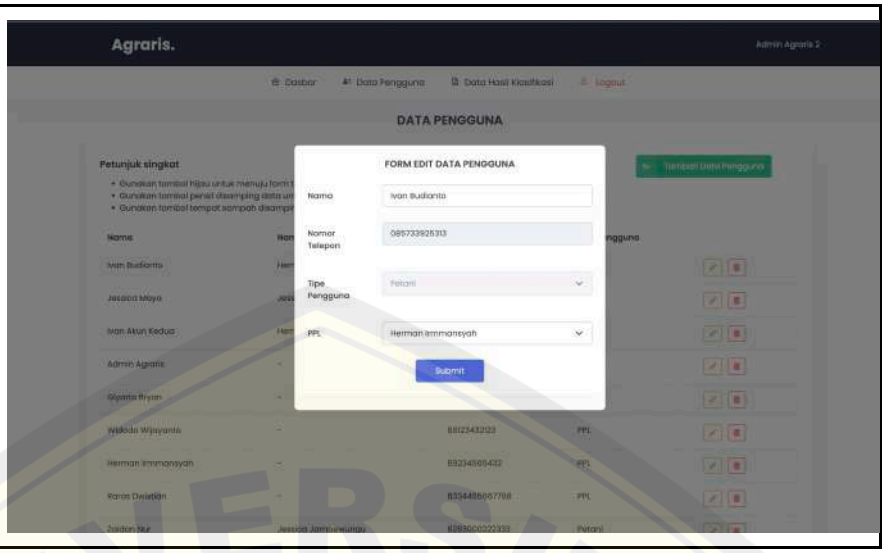
Tabel 4.19 Blackbox Testing Melihat Data Pengguna (Admin)

ID	BX-010
Fitur	Melihat Data Pengguna
User	Admin
Test Step	
Ekspektasi	Data Pengguna berhasil dilihat
Rules	• Data pengguna telah ditambahkan sebelumnya
Status	BERHASIL
Keterangan	

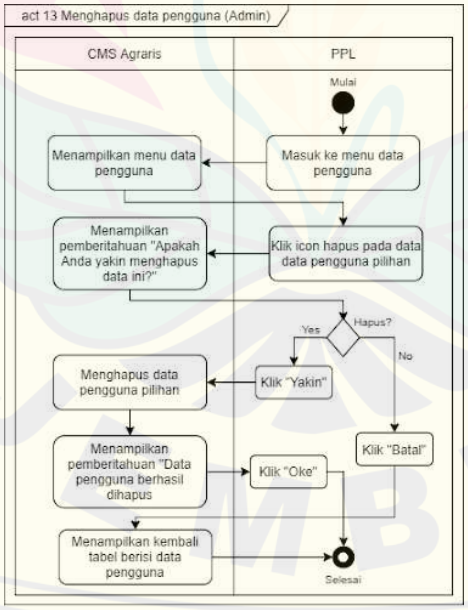
Tabel 4.20 Blackbox Testing Mengubah Data Pengguna (Admin)

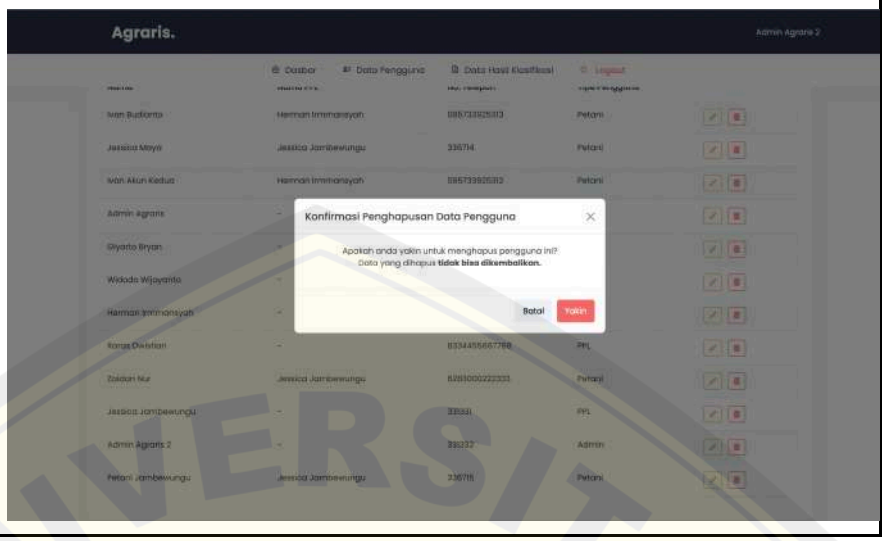
ID	BX-011

Fitur	Mengubah Data Pengguna
User	Admin
Test Step	
Ekspektasi	Data Pengguna berhasil diubah
Rules	• Data pengguna telah ditambahkan sebelumnya • Hanya field nama dan PPL yang dapat diubah • Nama lengkap harus diisi setidaknya 4 huruf
Status	BERHASIL

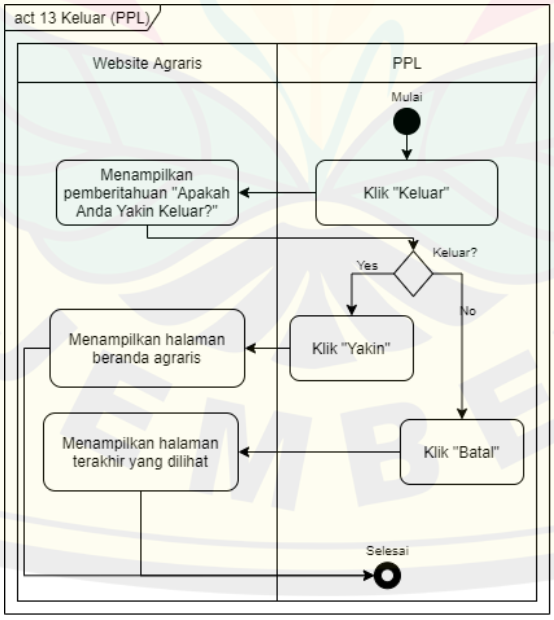
Keterangan	

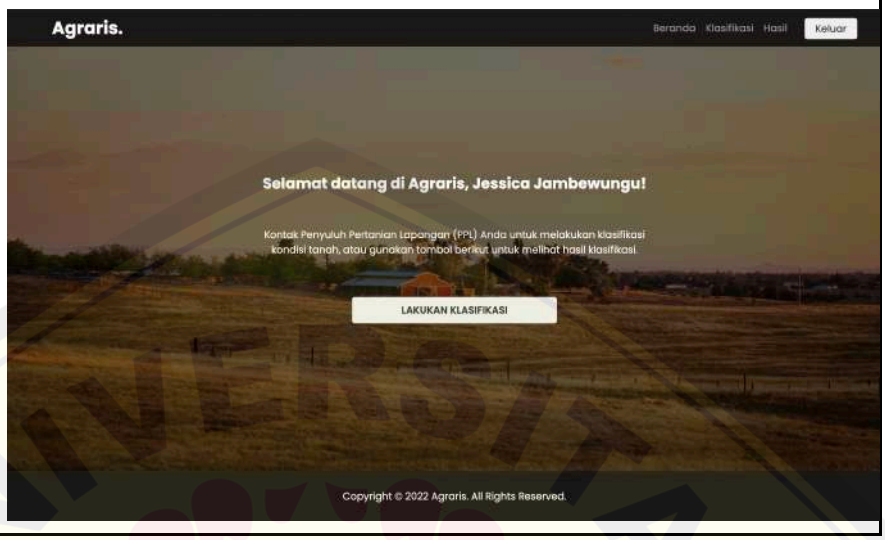
Tabel 4.21 Blackbox Testing Menghapus Data Pengguna (Admin)

ID	BX-012
Fitur	Menghapus Data Pengguna
User	Admin
Test Step	
Ekspektasi	Data Pengguna berhasil dihapus
Rules	• Data pengguna telah ditambahkan sebelumnya

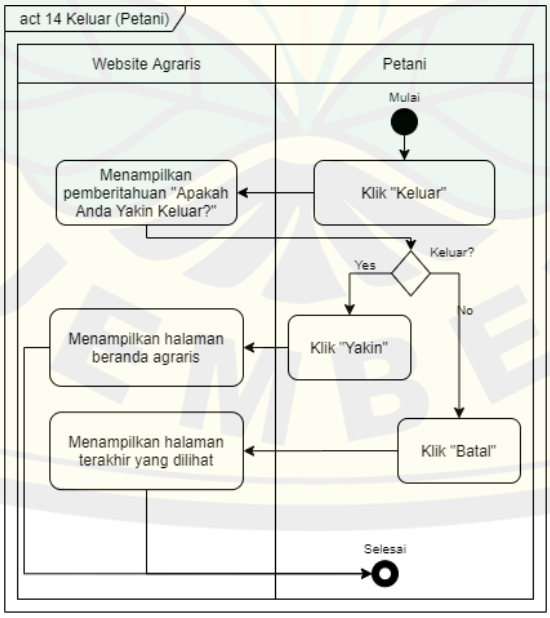
Status	BERHASIL
Keterangan	

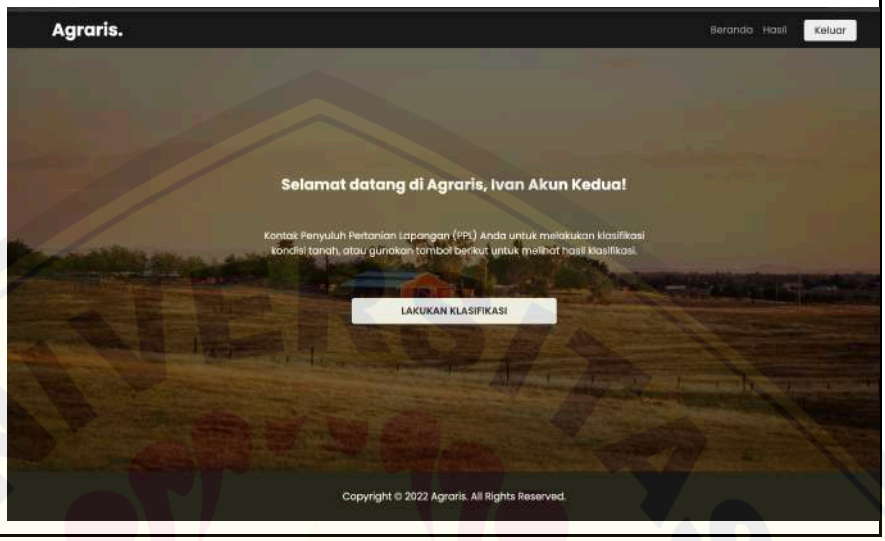
Tabel 4.22 Blackbox Testing Keluar (PPL)

ID	BX-013
Fitur	Keluar
User	PPL
Test Step	
Ekspektasi	Pengguna berhasil keluar dari website Agraris

Rules	-
Status	BERHASIL
Keterangan	

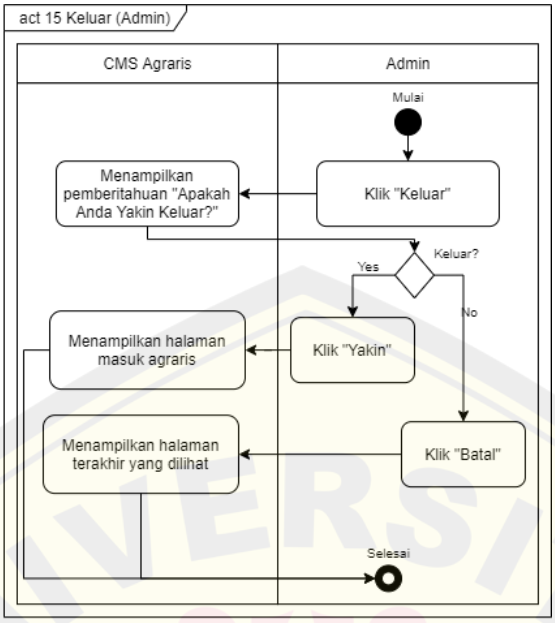
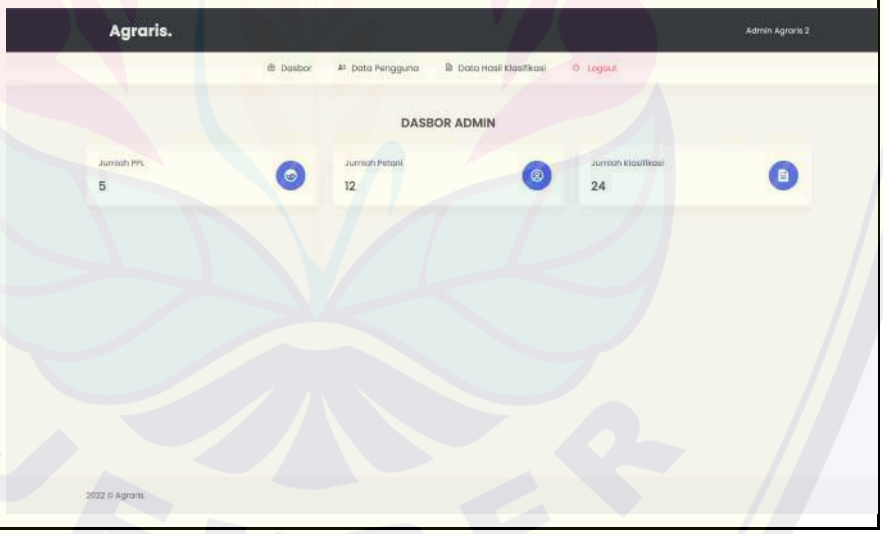
Tabel 4.23 Blackbox Testing Keluar (Petani)

ID	BX-014
Fitur	Keluar
User	Petani
Test Step	

Ekspektasi	Pengguna berhasil keluar dari website Agraris
Rules	-
Status	BERHASIL
Keterangan	

Tabel 4.24 Blackbox Testing Keluar (Admin)

ID	BX-015
Fitur	Keluar
User	Admin

Test Step	
Ekspektasi	Pengguna berhasil keluar dari website Agraris
Rules	-
Status	BERHASIL
Keterangan	

Berdasarkan Tabel 4.10 hingga Tabel 4.24, diketahui bahwa 3 *test case* yang dibuat telah berhasil untuk dilakukan. Maka dari itu, didapatkan kesimpulan bahwa sistem dapat berjalan dengan baik, dan telah siap untuk digunakan.

4.2 Tingkat Akurasi dan Hasil Evaluasi pada Klasifikasi Kondisi Tanah berdasarkan Rekomendasi Tanaman

Bagian ini menjelaskan mengenai tingkat akurasi dan hasil evaluasi dari model yang telah dikembangkan pada bagian 4.1. Berikut adalah hasil tingkat akurasi dan hasil evaluasi model.

4.2.1 Tingkat Akurasi

Terdapat beberapa skenario pada model yang dikembangkan. Skenario tersebut meliputi jumlah hidden layer yang digunakan, serta pembagian dari data. Terdapat 3 skenario pada jumlah hidden layer yang digunakan, yaitu penggunaan 3 hidden layer, 4 hidden layer, dan 5 hidden layer. Sedangkan, pada proses pembagian data, terdapat 3 skenario, yaitu 70% data latih, 15% data validasi, dan 15% data uji. Skenario kedua adalah pembagian 80% data latih, 10% data validasi, dan 10% data uji. Skenario terakhir adalah pembagian 90% data latih, 5% data validasi, dan 5% data uji. Tabel 4.25 adalah hasil perbandingan dari 9 skenario tersebut dalam bentuk nilai akurasi dari model.

Tabel 4.25 Hasil Akurasi Latih, Akurasi Validasi, dan Akurasi Uji Tiap Skenario

No.	Jumlah Hidden Layer	Pembagian Dataset	Akurasi Latih	Akurasi Validasi	Akurasi Uji
1	3	70% : 15% : 15%	98.96%	98.48%	96.66%
2	4	70% : 15% : 15%	99.18%	98.18%	95.75%
3	5	70% : 15% : 15%	99.29%	98.79%	96.25%
4	3	80% : 10% : 10%	98.96%	98.18%	97.02%
5	4	80% : 10% : 10%	98.88%	97.97%	97.72%
6	5	80% : 10% : 10%	99.25%	96.36%	93.63%
7	3	90% : 5% : 5%	98.96%	98.79%	97.57%
8	4	90% : 5% : 5%	99.30%	99.24%	98.93%
9	5	90% : 5% : 5%	99.29%	97.27%	98.18%

Hasil dari pengujian dengan 9 skenario tersebut, maka hasil yang dipilih adalah skenario ke-8 dengan nilai akurasi dan akurasi uji yang tertinggi bila dibandingkan dengan skenario lainnya. Hasil pelatihan yang dilakukan pada skenario ke-8 mencapai akurasi latih sebesar 99.30%, dengan akurasi validasi sebesar 99.24%, dan dibuktikan dengan validasi uji sebesar 98.97%. Akurasi tertinggi kedua adalah skenario ke-9 dengan akurasi sebesar 99.29% dengan akurasi validasi sebesar 97.27%, dan akurasi uji sebesar 98.18%. Berdasarkan hasil pada Tabel 4.25, hal ini berbanding lurus dengan penelitian yang dilakukan oleh Abu pada tahun 2018 mengenai perbandingan jumlah hidden layer pada neural network dengan permasalahan yang sama, yaitu klasifikasi multinomial.

Hasil dari pengembangan model juga didukung dengan skor akurasi pendukung, seperti nilai presisi, recall, dan F1 Score. Tabel 4.26 adalah nilai dari presisi, recall, dan F1 Score dari setiap model yang dikembangkan.

Tabel 4.26 Hasil Precision, Recall, dan F1 Score Tiap Skenario

No	Jumlah Hidden Layer	Pembagian Dataset	Precision	Recall	F1 Score
1	3	70% : 15% : 15%	97.17%	97.31%	97.15%
2	4	70% : 15% : 15%	96.78%	96.55%	96.41%
3	5	70% : 15% : 15%	97.13%	96.56%	96.63%
4	3	80% : 10% : 10%	97.96%	98.37%	98.07%
5	4	80% : 10% : 10%	98.03%	98.49%	98.00%
6	5	80% : 10% : 10%	98.11%	98.26%	98.00%
7	3	90% : 5% : 5%	97.42%	97.68%	97.05%
8	4	90% : 5% : 5%	99.24%	99.49%	99.32%
9	5	90% : 5% : 5%	97.58%	97.58%	97.58%

Berdasarkan Tabel 4.26, didapatkan bahwa dari 9 skenario yang ada dengan perubahan pada hidden layer dan proporsi pembagian data yang dilakukan, hasil ini mendukung hasil yang didapatkan pada Tabel 3.3. Kedua hasil menunjukkan bahwa skenario ke-8 memiliki hasil terbaik dibandingkan skenario

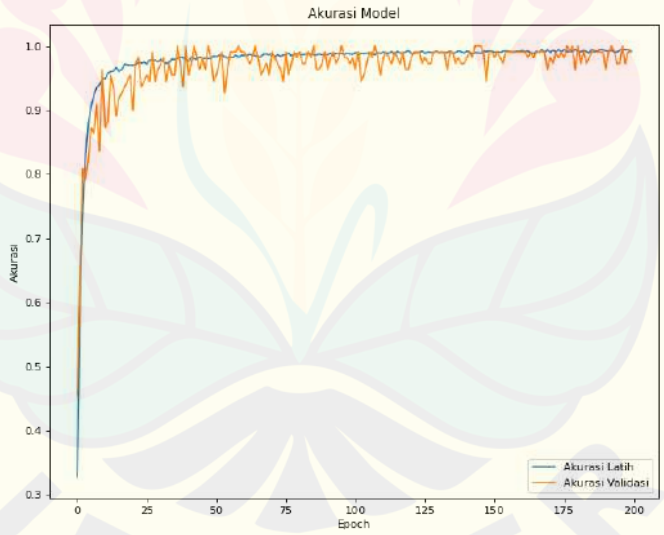
lainnya. Skenario ini merupakan skenario dengan 4 hidden layer dan pembagian data menjadi 90% data latih, 5% data validasi, dan 5% data uji. Hasil yang didapatkan adalah 99.24% precision, 99.49% Recall, dan 99.32% F1 Score.

4.2.2 Hasil Evaluasi Model

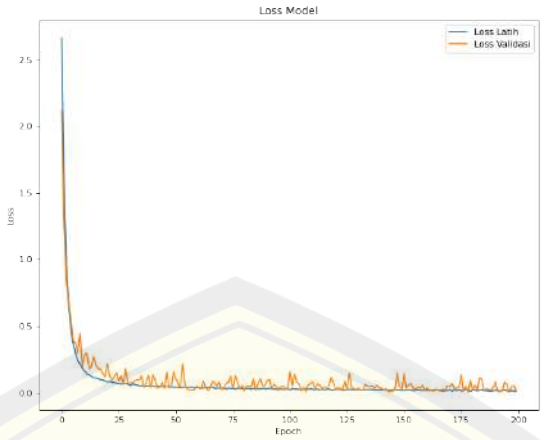
Hasil evaluasi model merupakan bagian untuk mengetahui efektivitas model yang digunakan. Bagian hasil evaluasi model mencakup 2 macam evaluasi. Evaluasi pertama adalah evaluasi model tanpa adanya pemberhentian model hingga epoch ke-200, sedangkan evaluasi kedua adalah dengan pemberhentian pelatihan model ketika telah mencapai konvergensi maksimal.

a. Hasil Evaluasi Model Hingga Epoch Terakhir

Hasil evaluasi ini meliputi grafik plotting dari model mengenai grafik plotting akurasi dan loss model selama proses pelatihan, serta confusion matrix dari model. Skala yang digunakan dalam proses plotting adalah desimal, dimana range dari data yang dihasilkan merupakan desimal berada diantara nilai nol dan satu.



Gambar 4.17 Grafik Plotting Akurasi dan Loss dari Model



Gambar 4.18 Loss Latih dan Validasi Pelatihan Model

Gambar 4.17 merupakan gambar hasil pelatihan dari model yang berlangsung selama 200 epoch. Dari iterasi selama 200 epoch tersebut, model berhasil untuk mendapatkan akurasi yang optimal pada epoch ke-25, dan dapat dikatakan telah mencapai konvergensi maksimal. Bila ditinjau dalam grafik, maka proses konvergensi dari model berjalan dengan optimal, dibuktikan dengan peningkatan dari akurasi latih dan akurasi validasi yang meningkat secara beriringan. Fakta ini didukung oleh Gambar 4.18 yang menunjukkan bahwa loss latih dan loss uji dari model yang berkurang secara beriringan hingga hampir mencapai nol.

b. Hasil Evaluasi Model pada Konvergensi Maksimal

Hasil pada model ini memiliki tingkat akurasi yang sedikit lebih rendah dibandingkan dengan model pada poin sebelumnya. Model ini memiliki akurasi latih sebesar 98.84% dengan selisih sebesar 0.46%. Namun, penggunaan model ini dapat mengurangi biaya komputasi sebesar 87.5%, dimana epoch yang digunakan hanya 25, dibandingkan dengan model sebelumnya yang menggunakan 200 epoch.

4.2.3 Confusion Matrix Model

Confusion Matrix dari model bertujuan untuk mengetahui letak kesalahan dari model dalam proses prediksi yang dilakukan. Dalam confusion matrix, ditemukan bahwa terdapat 1 kesalahan prediksi yang terjadi, yaitu pada label ke-13, yaitu Kacang Hijau, yang salah diprediksi menjadi Jagung. Gambar 4.19

merupakan confusion matrix dari model terbaik yang telah dikembangkan. Pada Gambar 4.19, hasil yang diharapkan atau hasil sempurna adalah seluruh angka terisi lebih dari 0 hanya pada pola struktur diagonalnya saja.

```
Out[27]: array([[5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4]])
```

Gambar 4.19 Confusion Matrix Model

4.2.4 Hasil Model yang Dikembangkan

Model terbaik adalah model skenario kedelapan. Hasil evaluasi yang didapatkan dari keseluruhan skenariopengembangan model adalah 3 jenis akurasi, yaitu akurasi latih, akurasi validasi, dan akurasi uji. Model terbaik memiliki akurasi latih sebesar 99.30%, akurasi validasi sebesar 99.24%, dan akurasi uji sebesar 98.93%. Model ini juga memiliki nilai Precision, Recall, dan F1 Score tertinggi bila dibandingkan dengan model lainnya. Nilai Precision yang didapatkan sebesar 99.24%, sementara nilai Recall yang didapat adalah 99.49%, dan nilai F1 Score yang didapat sebesar 99.32%.

Proses konvergensi model dapat dilihat pada Gambar 4.15 dan Gambar 4.16, dimana model terus mengelami perkembangan secara signifikan hingga mencapai epoch ke-15. Loss dari model menunjukkan hasil yang serupa, dimana seiring dengan akurasi yang meningkat tajam hingga epoch ke-15, loss dari model juga mengalami penurunan yang signifikan hingga epoch ke-15.

Hasil dari proses pengujian dapat dilihat dari akurasi uji sebesar 98.93%. Saat dilakukan validasi melalui confusion matrix, didapatkan bahwa hanya terdapat 1 kesalahan uji yang didapatkan, yaitu misklasifikasi pada label ke-13, yaitu Kacang Hijau, yang salah diprediksi menjadi label ke 11, yaitu Jagung.

BAB 5. KESIMPULAN

Bagian kesimpulan berisi mengenai kesimpulan yang diringkas, beserta saran untuk kedepannya. Berikut adalah kesimpulan dan saran.

5.1 Kesimpulan

Terdapat beberapa kesimpulan dari penelitian yang telah dilakukan. Penarikan kesimpulan dilakukan berdasarkan rumusan masalah, dan hasil yang didapat selama proses penelitian. Berikut adalah kesimpulan dari penelitian yang dilakukan.

1. Proses untuk membuat model machine learning terdiri dari 5 langkah. Langkah pertama adalah melakukan ekstraksi data untuk mendapatkan DataFrame dari data mentah. Langkah berikutnya adalah proses keseluruhan pengembangan model jaringan syaraf tiruan yang dimulai pada preprocessing data. Pada bagian ini dilakukan proses Label Encoding untuk mengubah label yang semula berupa kata-kata menjadi bilangan bulat. Lalu, fitur dan label akan dipisahkan. Akan dilakukan proses one hot encoding untuk mengubah bilangan bulat menjadi list bilangan biner. Pada proses pembagian data, terdapat 3 skenario. Data ini akan dinormalisasi sebelum dibuat menjadi 3 skenario model. Skenario model dapat dilihat pada Tabel 5.1. Hasil berupa 9 model akan dievaluasi untuk mencari model terbaik. Model terbaik akan di-deploy menuju website, dan dilakukan proses testing untuk mencari permasalahan yang terjadi pada sistem, dan mencoba apakah sistem telah berjalan sesuai dengan ekspektasi. Kesimpulan akan ditarik berdasarkan keseluruhan proses yang telah dilakukan.
2. Model terbaik adalah model skenario ke-8 dengan rasio pembagian data 90% data latih, 5% data validasi, dan 5% data uji. Model ini dibuat dengan 4 layer Dense dari TensorFlow dengan fungsi aktivasi ReLU dan Softmax. Optimizer yang digunakan adalah RMSProp. Model ini menghasilkan akurasi latih sebesar 99.30%, akurasi validasi sebesar 99.24%, akurasi uji sebesar 98.97%.

3. Model terbaik pada skenario kedelapan juga mencapai 99.24% precision, 99.49% Recall, dan 99.32% F1 Score. Tabel 5.2 merupakan tabel yang berisi hasil metrik yang diuji dari seluruh skenario yang dilakukan.
4. Model yang dihasilkan memiliki akurasi yang lebih tinggi dibandingkan dengan penelitian sebelumnya oleh Wibowo (2021) yang mencapai akurasi sebesar 99% dengan metode *Random Forest* pada penelitian tersebut.

5.2 Saran

Beberapa saran sebagai pengembangan dan/atau penyempurnaan penelitian ini adalah sebagai berikut.

1. Penelitian berikutnya dapat mencoba penelitian dengan pengubahan pada beberapa parameter dalam pelatihan, seperti variasi epoch yang dilakukan, optimizer yang digunakan, serta penggunaan layer kompleks seperti layer konvolusi.
2. Penelitian dapat diarahkan untuk membuat pipeline dari sistem yang dimulai dari pencarian parameter dari kondisi tanah untuk mencari parameter berupa kondisi tanah, baik secara kimiawi maupun fisik melalui penggunaan Internet of Things (IoT).

DAFTAR PUSTAKA

Alassadi, Abdulrahman, Tadas Ivanauskas. (2019). *Classification Performance Between Machine Learning and Traditional Programming in Java*. WEKA 2019.

Babu, Satish. (2013). *A Software Model for Precision Agriculture for Small and Marginal Farmers*. Trivandrum: IEEE.

Bahfein, Suhaiela. (2020). *Luas Baku Tanah Sawah Nasional 7,46 Juta Hektar*. Diakses melalui https://bit.ly/CSD074-Ref-Kompas.

Brownlee, J. (2020). *Imbalanced classification with Python: better metrics, balance skewed classes, cost-sensitive learning*. Machine Learning Mastery.

Brownlee, Jason. (2019). *Deep Learning for Computer Vision*. (v1.4)

Brownlee, Jason. (2019). *Better Deep Learning*. (v1.3)

Brownlee, Jason. (2016). *Machine Learning Mastery with Python: Understand Your Data, Create Accurate Models and Work Projects End-To-End.* (v1.3)

Brownlee, J. (2016). *Machine Learning Algorithms from Scratch with Python*. Machine Learning Mastery.

Chandra, Mahesh Mukkamala, Matthias Hein. (2017). *Variants of RMSProp and Adagrad with Logarithmic Regret Bounds*. ICML 2017.

CNN Indonesia. (2021). *Mentan: Ada 8 Juta Petani Baru di Indonesia karena Pandemi*. Diakses melalui https://bit.ly/CSD074-Ref-CNN.

Darol, Hosea Valentina (2021). Pengaruh Jenis Tanah Terhadap Pertumbuhan dan Hasil Produksi Tanaman Terong Asam

Darmawan, Didit, et.al. (2021). Tanaman Perkebunan Prospektif Indonesia. Jakarta: Qiara Media.

de Lima, Dominggus, J.S.A. Lamerkabel, Ingrid Welerubun. (2017). *Inventarisasi Jenis-Jenis Tanaman Penghasil Nektar dan Polen sebagai Pakan Lebah Madu Apis Mellifera di Kecamatan Kairatu Kabupaten Seram Bagian Barat.* Agrinimal Jurnal Ilmu Ternak Dan Tanaman, 7(2), 77-82.

Ertam, F., & Aydın, G. (2017, October). Data classification with deep learning using Tensorflow. In 2017 international conference on computer science and engineering (UBMK) (pp. 755-758). IEEE.so

Ghayoumi, M. (2021). Deep Learning in Practice. Chapman and Hall/CRC.

Hasnira, H., Windarko, N. A., Tjahjono, A., Nugroho, M. A. B., & Jati, M. P. (2020). Efficient Maximum Power Point Estimation Monitoring of Photovoltaic Using Feed Forward Neural Network. Jurnal Integrasi, 12(2), 92-104.

J., Stuart Russell, Peter Norvig. (2009). *Artificial Intelligence: A Modern Approach 3rd Edition*. New Jersey: Pearson Education, Inc.

Jääskeläinen, Antti, Mika Katara. (2012). Model-Based GUI Testing. Advances in Computers Volume 85, 65–122. doi:10.1016/b978-0-12-396526-4.00002-3.

Joseph, V. R. (2022). Optimal ratio for data splitting. *Statistical Analysis and Data Mining: The ASA Data Science Journal*.

Jung, Alexander. (2022). *Machine Learning: Foundations, Methodologies, and Applications*. Singapore: Springer Nature Singapore Pte Ltd.

Karamina, Hidayati, W Fikrinda, A. T. Murti (2017). *Kompleksitas Pengaruh Temperatur dan Kelembaban Tanah terhadap Nilai pH Tanah di Perkebunan Jambu Biji Varietas Kristal (Psidium Guajava L.) Bumiaji, Kota Batu*. (Vol 16, No 3).

Khomsatun, K., Ikhsan, D., Ali, M., & Kursini, K. (2020). Sistem Pengambilan Keputusan Pemilihan Lahan Tanam Di Kabupaten Wonosobo Dengan K-Means Clustering Dan Topsis. *Jurnal Nasional Pendidikan Teknik Informatika: JANAPATI*, *9*(1), 55-62.

Kominfo. (2019). Dominasi Sektor Pertanian, Jumlah Orang Bekerja Naik 2,50 Juta.
Diakses melalui https://www.kominfo.go.id/content/detail/22584/dominasi-sektor-pertanian-jumlah-orang-bekerja-naik-250-juta/0/berita.

Kulkarni, N. H., Srinivasan, G. N., Sagar, B. M., & Cauvery, N. K. (2018, December). Improving crop productivity through a crop recommendation system using ensembling technique. In 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS) (pp. 114-119). IEEE.

Korner, Christoper, Marcel Alsdorf. (2022). *Mastering Azure Machine Learning*. Birmingham: Packt Publishing Ltd.

Kumar, Rohit Rajak, Ankit Pawar, Mitalee Pendke, et.al. (2017). *Crop Recommendation System to Maximize Crop Yield using Machine Learning Technique*. India: Sinhagad Academy of Engineering.

Lonetti, Francesca, Eda Marchetti. (2018). Emerging Software Testing Technologies. Advances in Computers, 91–143. doi:10.1016/bs.adcom.2017.11.003.

Mala, N. M., Muhibuddin, A., & Sifaunajah, A. (2018). Sistem Pendukung Keputusan Pemilihan Penggunaan Jenis Tanaman dengan Metode Analytic Hierarchy Process (AHP). SAINTEKBU, 10(1), 64-74.

Mire, Archana, Vinayak Elangovan, Shailaja Patil. (2022). *Advances in Deep Learning for Medical Image Analysis*. Boca Raton: CRC Press.

Musyarrofatul, Nur Mala, Anton Muhibuddin, Agus Sifaunajah. (2018). *Sistem Pendukung Keputusan Pemilihan Penggunaan Jenis Tanaman Dengan Metode Analytic Hierarchy Process (AHP).* SAINTEKBU Jurnal Sains dan Teknologi (Vol. 10 No. 1).

Palmas, Alessandro, Emanuele Ghelfi, DR. Alexandra Galina Petre, et.al. (2020). *The Reinforcement Learning Workshop*. Birmingham: Packt Publishing Ltd.

Ping, David. (2022). *The Machine Learning Solutions Architect Handbook*. Birmingham: Packt Publishing Ltd.

Pricylia et al. 2018. Potensi Pengembangan Sumber Daya Manusia Penyuluh Pertanian Di Kabupaten Sigi. J. Agroland 25.

Raff Edward (2022). Inside Deep Learning: Math, Algorithms, Models. Manning.

Rajak, R. K., Pawar, A., Pendke, M., Shinde, P., Rathod, S., & Devare, A. (2017). Crop recommendation system to maximize crop yield using machine learning technique. International Research Journal of Engineering and Technology, 4(12), 950-953.

Raju, Nedunchezhian, et.al. (2022). *Empowering Artificial Intelligence Through Machine Learning*. Burlington: Apple Academic Press, Inc.

Russel, Alfred Conklin. (2005). *Introduction to Soil Chemistry: Analysis and Instrumentation*.

Sagala, D., Ningsih, H., Koryati, T., Ramdan, E. P., Indarwati, I., Herawati, J., ... & Septariani, D. N. (2021). Dasar-Dasar Agronomi. Yayasan Kita Menulis.

Sandiwantoro, Riko Tri. (2016). *Pengaruh Sistem Olah Tanah dan Pemberian Biochar pada Pertumbuhan dan Hasil Tanaman Jagung Manis (Zea mays saccharata Sturt.).* Universitas Brawijaya.

Schaul, T., et.al. (2014). Unit tests for stochastic optimization. *ICLR*, 2014

Siddique, M. A. B., Khan, M. M. R., Arif, R. B., & Ashrafi, Z. (2018, September). Study and observation of the variations of accuracies for handwritten digits recognition with various hidden layers and epochs using neural network algorithm. In 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT) (pp. 118-123). IEEE.

Sri, Bagus Mulyanto. (2013). *Kajian Rekomendasi Pemupukan Berbagai Jenis Tanah pada Tanaman Jagung, Padi dan Ketela Pohon di Kabupaten Wonogiri*. Fakultas Pertanian Universitas Sebelas Maret Surakarta.

Sumarni, N., Rosliani, R., & Duriat, A. S. (2010). Pengelolaan fisik, kimia, dan biologi tanah untuk meningkatkan kesuburan lahan dan hasil cabai merah. Jurnal Hortikultura, 20(2).

Surya, Dameis Anggara, Candra Abdillah. (2019). *Modul Metode Penelitian*. Tangerang Selatan: UNPAM PRESS

So, Anthony, William So, Zsolt Nagy (2020). *The Applied Artificial Intelligence Workshop*. Birmingham: Packt Publishing Ltd.

Vintarno, J., Sugandi, Y. S., & Adiwisastra, J. (2019). Perkembangan penyuluhan pertanian dalam mendukung pertumbuhan pertanian di Indonesia. Responsive: Jurnal Pemikiran Dan Penelitian Administrasi, Sosial, Humaniora Dan Kebijakan Publik, 1(3), 90-96.

Wibowo, Merlinda, Rafian Ramadhani. (2021). *Perbandingan Metode Klasifikasi Data Mining Untuk Rekomendasi Tanaman Pangan*. Jurnal Media Informatika Budidarma (Vol. 5 No. 3).

Wolf, Andrew. (2022). *Machine Learning Simplified: A Gentle Introduction to Supervised Learning*. THEMLSBOOK.COM.