UNIVERSITAS BHAYANGKARA JAKARTA RAYA
FAKULTAS ILMU KOMPUTER
Kampus I: Jl. Harsono RM No. 67, Ragunan, Pasar Minggu, Jakarta Selatan, 12550
Telepon: (021) 27808121 – 27808882
Kampus II: Jl. Raya Perjuangan, Marga Mulya, Bekasi Utara, Jawa Barat, 17142
Telepon: (021) 88955882, Fax.: (021) 88955871
Web: *fasilkom.ubharajaya.ac.id*, E-mail: fasilkom@ubharajaya.ac.id

SURAT TUGAS
Nomor: ST/389/III/2024/FASILKOM-UBJ

Tentang

PENUGASAN DOSEN PEMBIMBING KERJA PRAKTEK MAHASISWA PROGRAM STUDI INFORMATIKA FAKULTAS ILMU KOMPUTER SEMESTER GENAP TAHUN AKADEMIK 2023/2024

Menimbang : Bahwa dalam rangka pelaksanaan Sidang Kerja Praktek sebagai salah satu persyaratan untuk mencapai Gelar Sarjana Ilmu Komputer (S1), maka dipandang perlu mengeluarkan Surat Tugas.

Mengingat :
1. Undang-Undang No. 20 Tahun 2003 tentang Sistem Pendidikan Nasional;
2. Undang-Undang No. 14 Tahun 2005 tentang Guru dan Dosen;
3. Undang-Undang No. 12 Tahun 2012 tentang Pendidikan Tinggi;
4. Permendikbud No. 3 Tahun 2020 tentang Standar Nasional Pendidikan Tinggi;
5. Keputusan Menteri Pendidikan dan Kebudayaan Republik Indonesia Nomor: 830/M/2020 tentang Perubahan Nama Program Studi Teknik Informatika Program Sarjana pada Universitas Bhayangkara Jakarta Raya di Jakarta yang diselenggarakan oleh Yayasan Brata Bhakti;
6. Kalender Akademik Universitas Bhayangkara Jakarta Raya Semester Genap T.A. 2023/2024;
7. Kalender Akademik Fakultas Ilmu Komputer Universitas Bhayangkara Jakarta Raya Semester Genap T.A. 2023/2024, mengenai Kerja Praktek Mahasiswa Jenjang Pendidikan S-1.

DITUGASKAN

Kepada : **Aida Fitriyani, S.Kom., M.M.S.I.**
NIDN. 0302078508

Untuk :
1. Melaksanakan tugas sebagai Dosen Pembimbing Kerja Praktek Mahasiswa Program Studi Informatika Fakultas Ilmu Komputer Universitas Bhayangkara Jakarta Raya pada Semester Genap Tahun Akademik 2023/2024 (daftar nama terlampir).
2. Melaporkan hasil pelaksanaan kepada Dekan Fakultas Ilmu Komputer Universitas Bhayangkara Jakarta Raya.
3. Melaksanakan tugas ini dengan penuh tanggung jawab.

Ditetapkan di : Jakarta
Pada tanggal : 04 Maret 2024

DEKAN FAKULTAS ILMU KOMPUTER

Dr. Dra. Tyastuti Sri Lestari, M.M.
NIP. 1408206

LAMPIRAN
SURAT TUGAS PEMBIMBING KERJA PRAKTEK FASILKOM UBJ
NOMOR : ST/389/III/2024/FASILKOM-UBJ
TANGGAL : 04 MARET 2024

NO.	NAMA MAHASISWA	NPM
1.	Putri Mentari Rahmatillah	202110715237
2.	Rizky Aditia Purnomo	202110715250

Ditetapkan di : Jakarta
Pada tanggal : 04 Maret 2024
DEKAN FAKULTAS ILMU KOMPUTER

Dr. Dra. Tyastuti Sri Lestari, M.M.
NIP. 1408206

PT Mitra Talenta Grup
The Manhattan Square Building, Mid Tower, 12th Floor.
Jl. T.B. Simatupang kav 1-S. Cilandak Timur, Jakarta Selatan
Celerates.com
contact@celerates.com
021 8064 1000

Nomor : 211/LOA-MTG/KM-DS/I/2024

LETTER OF ACCEPTANCE

Saya yang bertanda tangan di bawah ini:

Nama Lengkap	: Prayoga Pangudhi Nugroho
Jabatan	: Head of Celerates School
Nama Perusahaan	: PT Mitra Talenta Grup (Celerates)

Selaku penanggung jawab Program Magang dan Studi Independen Bersertifikat (MSIB) Kampus Merdeka Angkatan 6 periode tahun 2024, dengan ini menyatakan bahwa nama yang terlampir bersama dengan surat ini:

Nama Lengkap	: Putri Mentari Rahmatillah
Perguruan Tinggi	: Universitas Bhayangkara Jakarta Raya
Program Studi/Jurusan	: Teknik Informatika
NIM	: 202110715237
Nomor Telp	: 89682431808

diterima untuk menjadi peserta di program **Studi Independen (Data Science)** di **PT Mitra Talenta Grup (Celerates)** dengan pelaksanaan **16 Februari – 30 Juni 2024**.

Demikian surat pernyataan ini kami sampaikan dan mohon digunakan dengan sebaik-baiknya.

Jakarta, 16 Februari 2024

Prayoga Pangudhi Nugroho

Head of Celerates School

SERTIFIKAT

MAGANG DAN STUDI INDEPENDEN BERSERTIFIKAT

Diberikan Kepada :

Putri Mentari Rahmatillah

ID Kegiatan : 9320812 - Universitas Bhayangkara Jakarta Raya - Teknik Informatika

Sebagai :

Peserta MSIB Angkatan 6

Telah berhasil menyelesaikan tugasnya di PT Mitra Talenta Grup dalam **program Studi Independen - Digital Creative, Data, Cybersecurity dan Artificial Intelligence** dengan kegiatan **Celerates Acceleration Program - Data Science Basics** yang diselenggarakan pada **tanggal 16 Februari-30 Juni 2024**.

Jakarta 30 Juni 2024
Chief Operating Officer
PT Mitra Talenta Grup,

Muhamad Risyad Ganis

CAPAIAN PEMBELAJARAN PROGRAM

No.	Kompetensi	Definisi Kompetensi	Jam	Nilai Capaian	Deskripsi Nilai Capaian
1.	Database & SQL	- Pada pembelajaran individu, peserta akan mengikuti pembelajaran synchronous dan asynchronous, serta praktikum. - Pembelajaran ini berfokus pada memberikan kemampuan pengertian dasar terhadap data. - Peserta diharapkan sudah mampu dan mengetahui cara-cara menggunakan bahasa pemrograman SQL untuk melakukan analisa data.	90 jam	86,56	Mampu melakukan query terhadap data terstruktur. Mampu melakukan fungsi Select, Join, Aggregate dan CTE. Mampu membuat query optimal. Mampu membuat query plan. Mampu melakukan Exploratory Data untuk mengambil insight dari data mentah yang ada pada database.
2.	Data Warehouse & ETL	- Pada pembelajaran individu, peserta akan mengikuti pembelajaran synchronous dan asynchronous, serta praktikum. - Pembelajaran ini berfokus pada memberikan kemampuan tentang penggunaan Data Warehouse dan pembentukan data warehouse menggunakan ETL Tools. - Peserta diharapkan sudah mampu dan mengetahui cara membuat sebuah Data Warehouse menggunakan ETL Tools. - Peserta akan diberikan metode, karakteristik dan konsep untuk mengembangkan Data Warehouse dan pengetahuan terkait tools ETL	90 jam	89,74	Mampu memahami konsep dan karakteristik Data Warehouse. Mampu membedakan OLTP dan OLAP. Mampu menggunakan ETL tools. Mampu melakukan cleansing, standarisasi, dan transform data pada ETL tools. Mampu memahami konsep Data Model. Mampu memahami Dimensi, Measure, dan fact Table Mampu memahami serta membuat skema pada Data Warehouse menggunakan ETL tools.
3.	Python for Data Science	- Pada pembelajaran individu, peserta akan mengikuti pembelajaran synchronous dan asynchronous, serta praktikum. - Kegiatan pembelajaran berfokus pada penggunaan bahasa pemrograman Python untuk kepentingan analisis dan modeling Data. - Peserta akan diberikan satu set data dan dibimbing langkah tiap langkah untuk bisa mendapatkan pengertian terhadap data yang diolah pada aplikasi Python.	135 jam	75,19	Mampu melakukan analisis data dengan pendekatan programming python. Mampu melakukan fungsi aritmatika dan statistik terhadap data menggunakan python. Mampu menggunakan modul-modul popular seperti Pandas, Numpy, Matplotlib. Mampu menerapkan Data Wrangling pada aplikasi Python. Mampu melakukan Exploratory Data Analytics dengan berbagai teknik.
4.	Statistic	- Peserta belajar mengenai dasar statistik. - Peserta belajar mengenai penggunaan statistik untuk Data Analytic. - Peserta belajar mengenai kesalahan-kesalahan yang sering terjadi dalam Data Analytic. - Peserta akan belajar mengenai metode sampling dan uji hipotesis. - Peserta kemudian akan melakukan eksplorasi data menggunakan python dengan menggunakan teknik teknik statistik. - Pembelajaran akan didukung menggunakan modul-modul yang ada pada aplikasi Python.	135 jam	75,00	Mampu memahami dasar-dasar statistika. Mampu memahami penggunaan ilmu statistika dalam analisis data. Mampu melakukan validasi data. Mampu memahami kesalahan-kesalahan statistik. Mampu melakukan sampling dengan metode yang benar. Mampu melakukan uji hipotesis dengan metode statistik. Mampu menerapkan kemampuan statistik pada aplikasi Python.
5.	Data Visualization (Tableau)	- Peserta akan melakukan pembelajaran secara synchronous dan asynchronous. - Pembelajaran ini berfokus pada memberikan kemampuan dasar melakukan Data Visualization menggunakan Tableau. Peserta diharapkan sudah mampu dan mengetahui cara membangun sebuah dashboard dari beberapa tipe sumber data serta menghasilkan sebuah insight guna pengambilan sebuah keputusan yang lebih baik.	90 jam	74,50	Mampu memahami fungsi Data Visualization. Mampu memahami terminology pada Tableau. Mampu melakukan pemanggilan data source. Mampu membedakan tipe data,dimensi,dan measure. Mampu membuat basic Chart. Mampu membuat Dashboard dan insight.
6.	Machine Learning	- Peserta akan melakukan pembelajaran secara synchronous dan asynchronous. - Peserta dengan dibekali ilmu statistik sebelumnya akan mempelajari teknik-teknik Machine Learning tidak hanya dari cara penggunaannya saja namun juga dari bagaimana cara kerjanya. - Peserta belajar mengenai teknik-teknik ML. - Peserta belajar cara melakukan cleansing data sebelum membentuk sebuah model. - Peserta belajar mengenai penggunaan teknik ML menggunakan Python. - Peserta belajar cara melakukan evaluasi terhadap model yang sudah dibentuk. - Peserta melakukan praktek dalam mini projek diakhir."	135 jam	69,57	Mampu memahami Bayesian Probability. Mampu melakukan data cleansing sebelum membentuk sebuah model. Mampu memahami teknik-teknik Machine Learning (Random Forest, Naive Bayes, dll). Mampu memahami cara melakukan evaluasi terhadap model (Akurasi, Presisi, Recall, F1-Score). Mampu memahami teknik ML terhadap data terstruktur. Mampu melakukan teknik ML (klasifikasi, regresi) menggunakan Python.
7.	Deep Learning	- Peserta akan melakukan pembelajaran secara synchronous dan asynchronous. - Peserta dengan dibekali ilmu Machine Learning akan melakukan prediksi namun menggunakan data tidak terstruktur. - Peserta belajar mengenai perbedaan antara Machine Learning dengan Deep Learning. - Peserta belajar mengenai Neural Networks. - Peserta belajar mengenai teknik-teknik Deep Learning. - Peserta belajar menggunakan teknik-teknik Deep Learning menggunakan Python dan Tensorflow.	135 jam	70,18	Mampu memahami perbedaan antara Machine Learning dengan Deep Learning. Mampu memahami teknik-teknik Deep Learning (CNN, RNN, LSTM). Mampu menggunakan teknik-teknik Deep Learning untuk melakukan Klasifikasi terhadap data tidak terstruktur. Mampu menggunakan Tensorflow dalam melakukan praktek Deep Learning.
8.	Final Project	- Peserta akan diminta untuk menulis report harian tentang apa yang mereka rasakan dan alami sepanjang final projek, peserta juga akan melakukan check point task per-minggu bersama mentor untuk monitoring progress pekerjaan. - Peserta melakukan final project dengan dataset yang sudah disediakan. - Peserta diharapkan sudah mampu menerapkan end-to-end proses dalam pengembangan model Deep Learning dari dataset yang sudah diberikan.	90 jam	92,07	Mampu menerapkan end-to-end proses dalam pengembangan model prediktif pada sebuah case.
Nilai rata-rata				86,14	A

Chief Operating Officer
PT Mitra Talenta Grup,

Muhamad Risyad Ganis

LAPORAN AKHIR

MAGANG & STUDI INDEPENDEN BERSERTIFIKAT

Studi Independen

Di PT Mitra Talenta Group

Celerates Acceleration Program

Putri Mentari Rahmatillah
202110715237

Nama Dosen Pendamping Program (DPP) :
Zahedi

PROGRAM STUDI INFORMATIKA
UNIVERSITAS BHAYANGKARA JAKARTA RAYA
2024

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga saya dapat menyelesaikan laporan akhir program Studi Independen Bersertifikat Kampus Merdeka ini. Laporan ini disusun sebagai salah satu persyaratan untuk menyelesaikan program Studi Independen Bersertifikat Kampus Merdeka di PT Mitra Talenta Group (Celerates).

Saya mengucapkan terima kasih kepada semua pihak yang turut serta dalam membantu kelancaran penyelesaian kegiatan dan penyusunan laporan Studi Independen Bersertifikat ini. Terima kasih kepada:

1. Allah SWT, Tuhan Yang Maha Esa, yang memberikan rahmat dan petunjuk-Nya, memandu langkah-langkah penulis sehingga proses penyusunan laporan dapat terselesaikan dengan lancar.
2. Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbud dan Ristekdikti) atas kesempatan melalui Program Kampus Merdeka.
3. Orang tua penulis, Ayah, Ibu, dan keluarga yang senantiasa memberikan doa dan materi untuk mencapai tahap ini dalam menyelesaikan Studi Independen Bersertifikat.
4. Kak Alya, mentor Akademik Data Science Celerates Acceleration Program , yang sabar memberikan bimbingan dan arahan sehingga laporan akhir dapat memuaskan.
5. Seluruh panitia dosen/mentor Studi Independen di PT Mitra Talenta Group (Celerates) yang memberikan bimbingan dan arahan selama pelaksanaan Studi Independen Bersertifikat ini.

Saya menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis menerima kritik dan saran yang membangun demi perbaikan laporan ini untuk kedepannya.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada era digital yang terus berkembang pesat, perusahaan dihadapkan pada tantangan yang semakin kompleks dalam mengelola sumber daya manusia (SDM). Salah satu tantangan utama adalah turnover karyawan yang tinggi, yang dapat berdampak negatif terhadap produktivitas, budaya organisasi, dan biaya operasional. Turnover karyawan tidak hanya mengganggu alur kerja yang sudah ada, tetapi juga memerlukan biaya tambahan untuk proses rekrutmen, pelatihan, dan adaptasi karyawan baru.

Pengelolaan data karyawan secara efektif menjadi kunci untuk mengatasi tantangan ini. Dengan memanfaatkan data historis karyawan, perusahaan dapat menganalisis pola dan faktor yang mempengaruhi keputusan karyawan untuk meninggalkan perusahaan. Analisis ini dapat membantu perusahaan dalam merumuskan strategi yang lebih tepat untuk mempertahankan karyawan berkualitas dan mengurangi tingkat turnover.

Dalam konteks ini, program Studi Independen yang diselenggarakan oleh PT Mitra Talenta Grup melalui Celerates Acceleration Program memberikan kesempatan bagi peserta untuk mempelajari dan mengimplementasikan teknik-teknik Data Science dalam dunia industri. Program ini tidak hanya menekankan pada pembelajaran teoretis, tetapi juga pada aplikasi praktis melalui proyek akhir yang melibatkan analisis data karyawan.

Penelitian yang dilakukan dalam program ini berfokus pada pengembangan model prediktif untuk memprediksi apakah seorang karyawan akan meninggalkan perusahaan atau tidak (LeaveOrNot). Dataset yang digunakan mencakup berbagai atribut karyawan seperti pendidikan, tahun bergabung, kota, tingkat pembayaran, usia, jenis kelamin, pengalaman di domain saat ini, dan apakah karyawan pernah diberhentikan sementara atau tidak. Dengan menggunakan algoritma machine learning seperti Decision Tree, Random Forest, dan XGBoost serta algoritma deep learning seperti Artificial Neural Network (ANN), penelitian ini bertujuan untuk mengidentifikasi pola dan faktor yang signifikan dalam memprediksi turnover karyawan.

Penelitian ini juga menunjukkan bagaimana penggunaan teknologi machine learning dan deep learning dapat memberikan nilai tambah yang signifikan dalam analisis data karyawan. Algoritma-algoritma ini mampu mengolah data dalam jumlah besar dan kompleks dengan cepat dan akurat, menghasilkan model prediktif yang dapat diandalkan.

1.2 Tujuan

Tujuan utama dari penelitian ini adalah untuk mengembangkan model prediksi yang dapat mengidentifikasi karyawan yang berpotensi untuk meninggalkan perusahaan. Dengan demikian, perusahaan dapat mengambil tindakan preventif untuk mengurangi tingkat turnover karyawan.

1.3 Manfaat

Penelitian ini memberikan beberapa manfaat, antara lain.

1. Pemahaman Mendalam tentang Karyawan: Analisis data memungkinkan perusahaan untuk memahami karakteristik karyawan yang berpotensi untuk meninggalkan perusahaan.
2. Tindakan Preventif: Dengan model prediksi yang dikembangkan, perusahaan dapat mengambil tindakan preventif untuk mengurangi tingkat turnover karyawan.
3. Optimasi Sumber Daya Manusia: Dengan mengetahui faktor-faktor yang mempengaruhi keputusan karyawan untuk pergi, perusahaan dapat mengoptimalkan manajemen sumber daya manusia mereka.

1.4 Metodologi Penelitian

Metode yang digunakan dalam penelitian ini meliputi pengolahan data, eksplorasi data, dan pembangunan model prediksi menggunakan beberapa algoritma machine learning dan deep learning. Algoritma yang digunakan untuk machine learning antara lain Decision Tree, Random Forest, dan XGBoost. Alasan pemilihan algoritma ini adalah sebagai berikut:

a. Decision Tree: Sederhana untuk dipahami dan diinterpretasikan, serta mampu menangani data non-linear.
b. Random Forest: Memperbaiki kelemahan Decision Tree dengan membangun sejumlah pohon keputusan dan menggabungkan hasilnya untuk meningkatkan akurasi dan mengurangi overfitting.
c. XGBoost: Algoritma boosting yang kuat dan efisien dalam hal waktu dan performa, serta sering kali memberikan hasil terbaik dalam kompetisi machine learning.

Setiap model memiliki kelebihan dan kekurangan masing-masing. Decision Tree mudah diinterpretasikan tetapi rentan terhadap overfitting. Random Forest lebih stabil tetapi kurang interpretatif. XGBoost biasanya memberikan performa terbaik tetapi memerlukan waktu lebih lama untuk training dan tuning.

Sedangkan algoritma deep learning yang digunakan, yaitu ANN (Artificial Neural Network) yang kerjanya meniru cara kerja otak manusia (lapisan antar neuron terhubung) untuk memproses data dan membuat keputusan. ANN merupakan algoritma yang cukup sederhana dan lumayan mudah dipahami, ANN juga menawarkan fleksibilitas dan kekuatan dalam menangani masalah klasifikasi yang kompleks, akurat, dapat digeneralisasi. Penggunaan ANN pada klasifikasi data table mungkin cukup umum digunakan dan dapat membantu proses analisa LeaveOrNot karyawan menggunakan jaringan syaraf.

BAB II

TINJAUAN PUSTAKA

A. Penjelasan Log Book

Pada Bagian ini berisi aktivitas mingguan yang dapat diambil dari laporan mingguan yang sudah dibuat di dalam platform dengan format sebagai berikut

Bulan	Kegiatan
1	Laporan Bulan ke-1 (16 feb-15 mar 2024) Selama bulan pertama ini, aktivitas mentoring dan koordinasi dengan Mentor & DPP dilakukan melalui sesi zoom yang diadakan pada pagi hari. Pertemuan ini dimulai dengan salam hangat dan penjelasan singkat tentang agenda hari itu. Setelah itu, Mentor & DPP memberikan materi pelatihan yang disesuaikan dengan kebutuhan kami sebagai peserta MSIB Celerates. Mereka juga memberikan arahan dan panduan tentang bagaimana mengatasi tantangan yang mungkin timbul selama proses pembelajaran. Kegiatan yang dilakukan selama minggu ini terutama berfokus pada pembelajaran SQL. Kami memulai dengan materi dasar seperti pengenalan SQL dan kemudian berkembang ke topik-topik yang lebih kompleks seperti penggunaan operator logika dan teknik join dengan multiple tables. Selama proses pembelajaran, kami diberi tugas-tugas untuk menguji pemahaman kami dan melakukan latihan-latihan praktis. Perkembangan kami dalam memahami konsep-konsep SQL secara bertahap terlihat dari kemampuan kami dalam menyelesaikan tugas-tugas tersebut. Tantangan yang Dihadapi dan Solusi Alternatif: harus mencerna dengan baik materi yang disampaikan cepat oleh pelatih, namun masalah ini dapat diminimalisir dengan adanya komunikasi antar siswa dan pelatih mengenai masalah ini. Salah satu tantangan yang dihadapi adalah kompleksitas beberapa konsep dalam SQL, terutama ketika kita mencoba menerapkannya dalam konteks kasus nyata. Untuk mengatasi tantangan ini, kami dapat mengusulkan untuk melakukan lebih banyak latihan praktis dan studi kasus yang relevan. Selain itu, berdiskusi dengan sesama peserta atau meminta bantuan tambahan dari Mentor & DPP juga bisa menjadi solusi alternatif yang baik. Pengembangan Kompetensi yang Didapat: Siswa akan memahami dasar-dasar database Siswa akan memahami cara mengelola data (dengan query) menggunakan Postgresql dan DBeaver Mahasiswa memahami proses transformasi data dengan menggunakan software pentaho.

2	Logbook Pertemuan Bulan ke-2 (16 Mar - 15 Apr 2024) Kegiatan pendampingan dengan mentor dan DPP, komunikasi antar mentee, mentor dan DPP berjalan secara kondusif dan komunikatif melalui grup WhatsApp. Selain komunikasi di Grup WhatsApp, ada beberapa kegiatan lain yang dilakukan antara mahasiswa dengan mentee atau DPP: - Pertemuan sinkron dengan peserta melalui pertemuan zoom untuk menginformasikan pelajar menilai unsur-unsur dalam proses pembelajaran seperti (pelatih, diri sendiri, teman belajar, msib dan mentee) melalui aplikasi - Rapat sinkron dengan DPP melalui zoom meeting sebagai bentuk pemberian DF wadah bagi mahasiswa untuk melakukan konsultasi mengenai pembelajaran, konversi SKS dan lain sebagainya. DPP kita melayaninya dengan sangat baik. Kegiatan yang dilakukan pelajar di bulan kedua ini adalah memperdalam pemahamannya tentang Python bahasa pemrograman untuk menerapkan analisis data yang efektif, serta membuat catatan rinci pada seluruh materi yang disampaikan oleh pelatih (Putra Gema). Rekaman telah dilakukan 100% selesai dan telah diserahkan sebagai tugas. Tantangannya yang dihadapi oleh pelajar adalah ketidaklengkapan materi yang disampaikan, yang mengharuskan siswa melakukan eksplorasi tambahan untuk memperoleh pemahaman lebih dalam dari materi. Kegiatan yang saya lakukan adalah mencari referensi tambahan lainnya website atau menonton video pembelajaran di platform YouTube terkait Pemrograman python. Hal ini merupakan salah satu alternatif solusi untuk menghadapi permasalahan tersebut
3	Loogbook Pertemuan Bulan ke-3 (16 Apr - 15 Mei 2024) Kegiatan program studi independen berjalan lancar. Saya melakukan mentoring bersama dengan mentor dan DPP pun dilakukan di bulan ini. Mentoring bersama mentor membahas pembelajaran yang sudah di berikan dan juga pembahasan tugas yang sudah di kerjakan. Dan juga bersama dengan DPP pembahasan mengenai perkembangan kegiatan studi independen. Selama di bulan ke 3, pembelajaran yang dipelajari mulai dari materi statistik, marticula, marchine learning, dan deep learning. Saya mengikuti pembelajaran dengan baik, dan menambah ilmu pengetahuan saya dengan informasi-informasi baru mengenai materi-materi tersebut. Sehingga saya dapat mempraktikkan nya sendiri setelah materi berakhir. Pada materi yang dipelajari ada beberapa tantangan yang saya hadapi, seperti di awal materi yang ada saya masih agak sulit untuk memahami materi, namun dengan adanya mentor dan juga video pemahaman materi membantu saya untuk dapat memahami materi tersebut lebih mendalam. Materi statistik yang sebelumnya pernah saya pelajari, pada sesi ini saya mendapatkan kembali informasi yang lebih banyak sehingga saya mendapatkan wawasan baru dalam pengelolaan data di statistik. Pada materi marchine learning dan deep learning ini membantu saya untuk pengelolaan data dengan model-model. Perkembangan selama saya melakukan studi indepen ini, informasi yang saya dapatkan sangat meningkat. Sehingga saya dapat memahami pembelajaran dengan baik.
4	Logbook Bulan Ke-4 (16 Mei - 15 Jun 2024)

	Aktivitas Mentoring dan Koordinasi dengan Mentor & DPP Selama dibulan ini, aktivitas yang dilakukan yakni, disesi mentoring ini digunakan untuk membahas perkembangan project, memberikan informasi mengenai project, dan memastikan bahwa semua anggota memahami dengan project yang telah ditetapkan. Apa yang telah kamu kerjakan dan bagaimana perkembangannya? 16-21 Mei 2024 Mempelajari materi LangChain, Artificial Neural Network, dan metodologi analitik, serta pengenalan tableau. Pembahasan yang diberi yakni, tentang pemahaman konsep, instalasi perangkat lunak, dan eksplorasi fitur tableau. 22-27 Mei 2024 Mempelajari visualisasi data dengan tableau dan membuat diagram dan dashboard dari dataset yang berbeda. 29-30 Mei 2024 Mempelajari cara menghubungkan tableau dengan database postgres dan membuat berbagai diagram dan peta dari database Northwind. 3-14 Juni 2024 Berkontribusi dalam kelompok untuk final project, mencakup preprocessing data, analisis data eksploratori, optimasi model, dan analisis hasil. Tantangan apa yang dihadapi dan berikan alternatif solusi untuk menghadapinya? Tantangannya yakni, memahami dan menerapkan konsup-konsep yang begitu rumit seperti Langchain, Prompt Engineering, Artificial Neural Networks, dan teknik meng analisis data dengan Tableau. Dan beberapa kesulitan di integrasi dan menganalisis data yang efektif, pembuatan visualisasi yang informatif dan interaktif. Untuk mengatasinya, solusi yang bisa dilakukan yakni: •Meningkatkan sesi mentoring •Memanfaatkan sesi online dan forum diskusi. Apa saja dan jelaskan pengembangan kompetensi yang telah didapat? Selama dibulan ini, saya mendapatkan banyak pembelajaran tentang analisis data. Saya memahami apa itu LangChain dan teknik prompt engineering untuk Large Language Models (LLMs), lalu membangun dan mengintegrasikan aplikasi yang lebih rumit, mempelajari tentang Artificial Neural Network (ANN), penggunaan Tableau, mulai dari instalasi, membuat koneksi data langsung, hingga membuat dashboard interaktif, cara menghubungkan Tableau dengan database Postgres untuk menganalisis data dari Northwind. Dan kegiatan kerja kelompok membuat final project, yang mencakup preprocessing data, optimasi model, dan evaluasi kinerja model, dengan bantuan mentor di checkpoint.
5	Logbook Bulan ke-5 (16 - 30 Jun 2024) Pada bulan ke-5 ini, aktivitas mentoring dilakukan secara rutin melalui sesi Zoom. Lalu dipetemuan ini dimulai dengan penjelasan singkat tentang persiapan final project. Mentor memberi arahan mengenai masalah yang mungkin ada selama proses pembelajaran. Sesi-sesi mentoring ini juga membahas evaluasi kemajuan final project. Selama bulan ini, kegiatan diisi dengan pembuatan final project, laporan, dan vidio presentasi. Dalam final project ini, kami menerapkan konsep-konsep yang telah diajarkan selama kegiatan msib ini. Salah satu tantangan utama yang dihadapi adalah penerapan SQL dan Python, serta kesulitan dalam menghubungkan Tableau dengan database Postgres. Selain itu, waktu yang terbatas dalam sering kali sulit untuk mendalami materi secara menyeluruh.

	Untuk mengatasi tantangan ini, berikut beberapa solusi alternatif: -Menyediakan lebih banyak latihan dan studi kasus yang relevan akan membantu saya menerapkan teori ke dalam praktik. - Mengadakan sesi mentoring tambahan untuk membantu memecahkan masalah dan memperdalam pemahaman. Selama periode ini, saya telah mengembangkan beberapa kompetensi penting: - Penguasaan SQL Saya telah belajar membuat query yang baik dan efektif serta mengatasi tantangan dalam menganalisis data menggunakan SQL. - Kemampuan Pemrograman Python Saya sudah memahami dasar-dasar Python, mulai dari sintaks, tipe data, hingga penggunaan Pandas dan NumPy. - Kemampuan Visualisasi Data dengan Tableau Saya telah belajar membuat berbagai jenis visualisasi data dan dashboard interaktif, serta menghubungkan Tableau dengan database. - Soft Skills Saya juga sudah mengembangkan kemampuan komunikasi dan kerja sama tim melalui diskusi dan kerjasama dengan teman-teman dan mentor. Ini juga termasuk kemampuan presentasi, public speaking, dan leadership yang lebih baik.

BAB III
DASAR TEORI

Dalam dunia machine learning dan deep learning, algoritma seperti Random Forest, Decision Tree, XGBoost, dan Artificial Neural Network (ANN) sering digunakan untuk melakukan klasifikasi dan prediksi dalam berbagai konteks data, termasuk data karyawan. Algoritma-algoritma ini memiliki kemampuan signifikan dalam menangani dataset yang kompleks dan memberikan hasil yang akurat.

Berbagai penelitian terkait dengan penerapan atau penggunaan machine learning dan deep learning untuk analisa juga telah banyak dilakukan. Random Forest adalah metode ensemble yang menggabungkan banyak pohon keputusan (Decision Trees) untuk meningkatkan akurasi dan mengurangi overfitting. Dengan menggabungkan hasil dari berbagai pohon keputusan yang dibuat dari subset data dan fitur yang berbeda, Random Forest menghasilkan model yang lebih robust dan stabil. Penelitian menunjukkan bahwa Random Forest efektif dalam berbagai aplikasi, termasuk prediksi arah harga saham dan estimasi porositas efektif dari data log sumur, dengan akurasi yang sangat tinggi.

Decision Tree adalah algoritma pembelajaran mesin yang menggunakan model berbasis pohon untuk membuat keputusan dari data input. Pohon keputusan membagi data menjadi subset yang lebih kecil berdasarkan fitur tertentu hingga mencapai keputusan akhir. Meskipun sederhana, pohon keputusan dapat menjadi dasar bagi metode ensemble yang lebih kompleks seperti Random Forest dan XGBoost.

XGBoost (Extreme Gradient Boosting) adalah algoritma boosting yang meningkatkan kinerja model dengan menggabungkan hasil dari banyak model lemah untuk membentuk model yang kuat. XGBoost terkenal karena kecepatan dan efisiensinya dalam mengatasi data besar serta akurasinya yang tinggi dalam berbagai aplikasi prediksi. Penelitian menunjukkan bahwa XGBoost lebih cepat dalam proses pembelajaran data dan memiliki akurasi yang tinggi dalam berbagai kasus, seperti prediksi harga saham dan analisis nasabah perbankan.

Untuk Deep Learning juga misalnya, dalam penelitian berjudul "Klasifikasi Penyakit Jantung Menggunakan Metode Artificial Neural Network," metode ANN digunakan untuk melakukan klasifikasi penyakit jantung dengan hasil akurasi 73,77%, presisi 80,43%, recall 84,09%, dan f1-score sebesar 82,22% (ETD Repository). Penelitian lain, "Analisa Kelulusan Mahasiswa Teknik Informatika Tepat Waktu Menggunakan Algoritma Artificial Neural Network (ANN)," menunjukkan bahwa algoritma ANN mampu memprediksi kelulusan mahasiswa dengan akurasi 90%, presisi 93%, dan recall 86% (ETD Repository). Selain itu, penelitian "Prediksi Kemiskinan di Provinsi Papua Menggunakan Algoritma Neural Network" menggunakan data dari tahun 2010-2021 dan menunjukkan bahwa model neural network dapat menghasilkan RMSE sebesar 0.072, menunjukkan kemampuan prediksi yang baik dalam konteks sosial ekonomi.

Dataset karyawan yang mencakup informasi tentang pendidikan, tahun bergabung, kota, tier pembayaran, usia, jenis kelamin, pengalaman kerja, dan status pernah diistirahatkan atau tidak, dapat dianalisis menggunakan algoritma-algoritma ini untuk menentukan faktor-faktor yang mempengaruhi keputusan karyawan untuk tetap atau meninggalkan perusahaan. Misalnya, Random Forest dan XGBoost dapat digunakan untuk memprediksi apakah seorang karyawan akan meninggalkan perusahaan berdasarkan fitur-fitur tersebut, dengan memanfaatkan kemampuan mereka dalam menangani data yang kompleks dan memberikan hasil prediksi yang akurat. Model deep learning seperti ANN juga dapat diterapkan untuk klasifikasi yang lebih kompleks dan memperoleh insight yang lebih mendalam dari data karyawan.

Dengan menggunakan Random Forest, Decision Tree, XGBoost, dan ANN, kita dapat membangun model yang tidak hanya membantu dalam memahami faktor-faktor yang mempengaruhi retensi karyawan tetapi juga memberikan wawasan yang dapat digunakan untuk strategi manajemen sumber daya manusia yang lebih efektif.

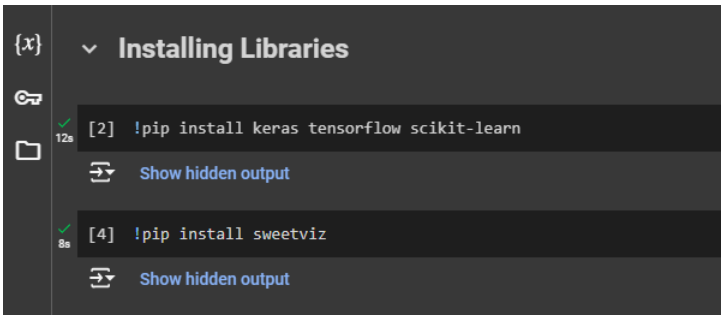
BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengolahan Data

4.1.1 Installing Libraries

Langkah pertama dalam proses pengolahan data adalah memastikan bahwa semua pustaka yang diperlukan telah diinstal. Pustaka-pustaka ini adalah alat yang akan digunakan untuk analisis, pemodelan, dan visualisasi data.

1. `!pip install keras tensorflow scikit-learn menginstal library `keras`, `tensorflow`, dan `scikit-learn`.
2. !pip install sweetviz menginstal library sweetviz untuk visualisasi data yang komprehensif.



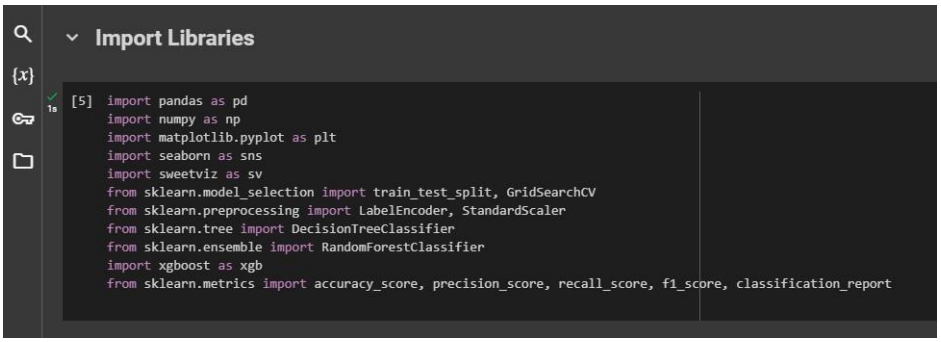
Gambar 4. 1 Instal Librariesn

4.1.2 Import Libraries

Setelah pustaka diinstal, langkah berikutnya adalah mengimpor pustaka tersebut ke dalam environment Python. Pustaka yang digunakan mencakup:

1. `pandas` dan `numpy` digunakan untuk manipulasi data.
2. `matplotlib.pyplot` dan `seaborn` untuk visualisasi data.
3. `sweetviz` untuk pembuatan laporan eksplorasi data otomatis.
4. `sklearn.model_selection` untuk pembagian data dan pencarian hyperparameter.

5. `sklearn.preprocessing` untuk encoding label dan standardisasi fitur.
6. `sklearn.tree` dan `sklearn.ensemble` untuk membangun model Decision Tree dan Random Forest.
7. `xgboost` untuk membangun model XGBoost.
8. `skle4arn.metrics` untuk evaluasi model.



Gambar 4. 2 Import Libraries 1

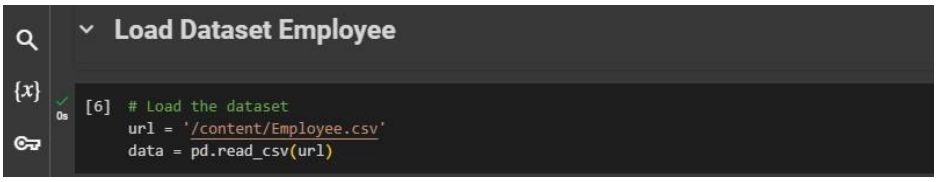
9. ‘sequential’ dari keras untuk membuat lapisan bertumpuk secara berurutan (membangun jaringan syaraf).
10. ‘Layer Dense’ untuk mengimpor lapisan dense yang merupakan lapisan jaringan syaraf sepenuhnya terhubung pada model ANN (neuron lapisan 1-neuron lapisan2-dll).
11. ‘Layer Dropout’ untuk mencegah over vitting dan membantu generalisasi model(dengan menambah lapisan dropout).
12. ‘sklearn.preprocessing import standartscaler’ digunakan untuk normalisasi/standarisasi fitur numeric sebelum melatih model.
13. ’keras.utils import to_categorical’ untuk mengubah label kelas menjadi represetasi one-hot encoding dalam klasifikasi multikategori jaringan syaraf(untuk pelatihan ANN).
14. ‘sklearn.metrics import confusion_matrix’ untuk menghitung matrix kebingungan, mengevaluasi kinerja algoritma klasifikasi dalam memprediksi benar dan salah

```
[146] from keras.models import Sequential
      from keras.layers import Dense, Dropout
      from sklearn.preprocessing import StandardScaler
      from keras.utils import to_categorical
      from sklearn.metrics import confusion_matrix
```

Gambar 4. 3 Import Libraries 2

4.1.3 Load Data

Memuat dataset dari file CSV ke dalam DataFrame pandas. Data karyawan dimuat menggunakan pandas. Dataset berisi informasi mengenai pendidikan, tahun bergabung, kota, tingkat pembayaran, usia, jenis kelamin, pernah diberhentikan sementara, pengalaman di domain saat ini, dan keputusan untuk meninggalkan perusahaan. `pd.read_csv(url)` membaca file CSV dari URL dan memuatnya ke dalam DataFrame data.

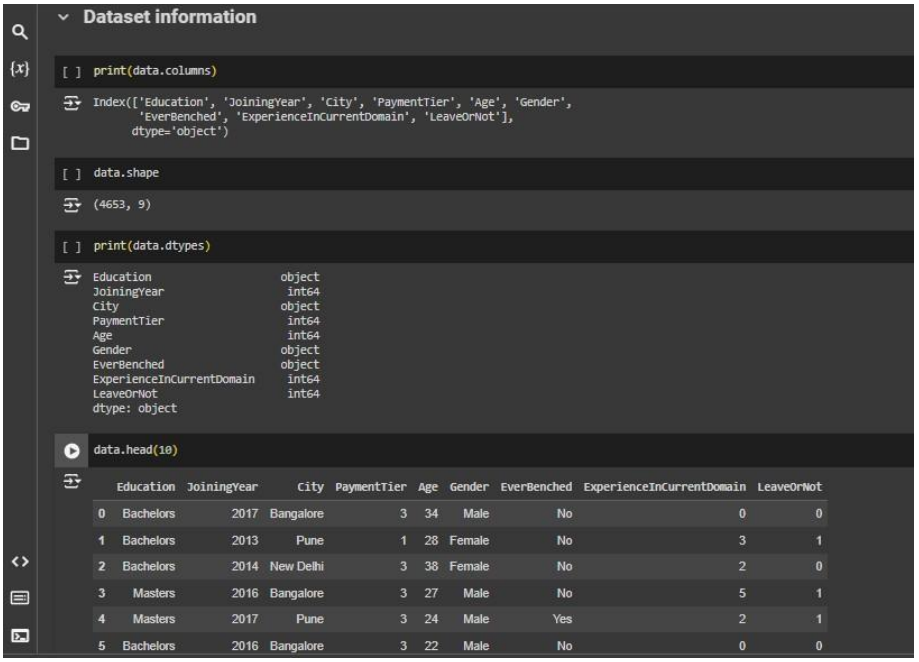


Gambar 4. 4 Load Data

4.1.4 Dataset Information

Pada tahap ini, dilakukan pemeriksaan informasi dasar mengenai dataset. Hal ini mencakup nama-nama kolom, jumlah baris dan kolom, tipe data untuk setiap kolom, serta 10 baris pertama data untuk mendapatkan gambaran awal tentang dataset.

1. `data.columns` menampilkan nama kolom dalam dataset.
2. `data.shape` memberikan jumlah baris dan kolom.
3. `data.dtypes` menampilkan tipe data setiap kolom.
4. `data.head(10)` menampilkan 10 baris pertama dari dataset.

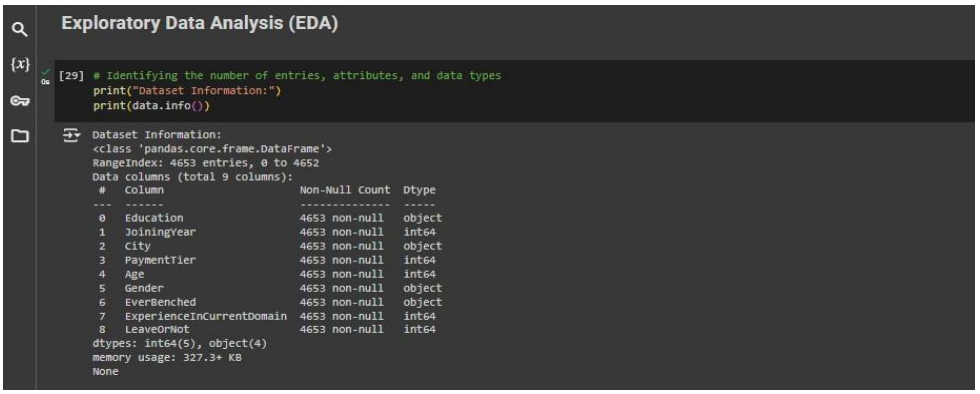


Gambar 4. 5 Dataset Information

4.1.5 Exploratory Data Analysis (EDA)

Pada tahap Exploratory Data Analysis (EDA), dilakukan analisis data awal untuk memahami distribusi data, mendeteksi anomali, dan mencari pola yang menarik. EDA membantu dalam menentukan strategi preprocessing dan fitur yang relevan untuk pemodelan.

1. `data.info()` memberikan informasi ringkas tentang dataset. Pemeriksaan jumlah entri, atribut, dan tipe data untuk memahami struktur dataset secara keseluruhan.



Gambar 4.6 EDA Dataset Information

2. `data.isnull().sum()` mengecek jumlah nilai yang hilang pada setiap kolom.

Identifikasi kolom yang memiliki nilai yang hilang untuk menentukan apakah diperlukan penanganan khusus, seperti imputasi atau penghapusan baris yang hilang.

```
[30] # Checking for missing values
     print("\nMissing Values:")
     print(data.isnull().sum())

Missing Values:
Education                   0
JoiningYear                 0
City                        0
PaymentTier                 0
Age                         0
Gender                      0
EverBenched                 0
ExperienceInCurrentDomain   0
LeaveOrNot                  0
dtype: int64
```

Gambar 4.7 Checking Missing Values

3. `data.describe()` memberikan statistik deskriptif untuk atribut numerik. Menghitung statistik deskriptif untuk atribut numerik untuk mendapatkan gambaran mengenai distribusi, rata-rata, standar deviasi, nilai minimum dan maksimum.

```
[31] # Descriptive statistics for numerical attributes
     print("\nDescriptive Statistics:")
     print(data.describe())

Descriptive Statistics:
       JoiningYear  PaymentTier          Age  ExperienceInCurrentDomain  \
count  4653.000000  4653.000000  4653.000000                4653.000000
mean   2015.062970     2.698259    29.393295                   2.905652
std       1.863377     0.561435     4.826087                   1.558240
min    2012.000000     1.000000    22.000000                   0.000000
25%    2013.000000     3.000000    26.000000                   2.000000
50%    2015.000000     3.000000    28.000000                   3.000000
75%    2017.000000     3.000000    32.000000                   4.000000
max    2018.000000     3.000000    41.000000                   7.000000

       LeaveOrNot
count  4653.000000
mean      0.343864
std       0.475047
min       0.000000
25%       0.000000
50%       0.000000
75%       1.000000
max       1.000000
```

Gambar 4.8 Descriptive Statistics

Menggunakan library Sweetviz dalam tahap eksplorasi awal dataset untuk membuat laporan eksplorasi data yang interaktif dan visual, membantu dalam memahami distribusi data dan hubungan antar atribut dengan lebih baik.

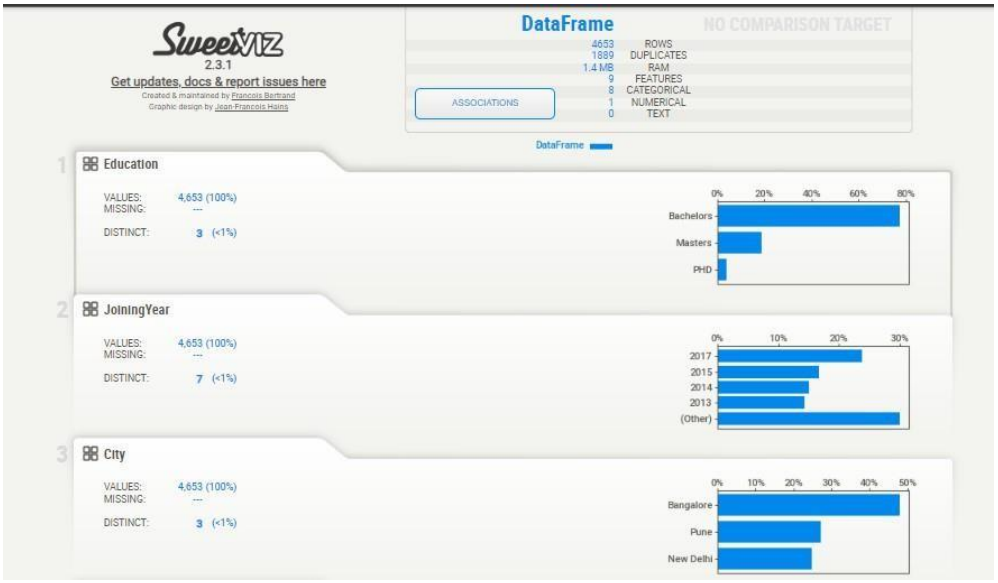
4. `sv.analyze(data)` menghasilkan laporan Sweetviz yang komprehensif.
5. `report.show_notebook()` menampilkan laporan di notebook.

```
[ ]  # Visualizing data distributions with Sweetviz

     report = sv.analyze(data) # Generate Sweetviz report

     report.show_notebook() # Show the report
```

Gambar 4. 9 Visualization Code Using Sweetviz

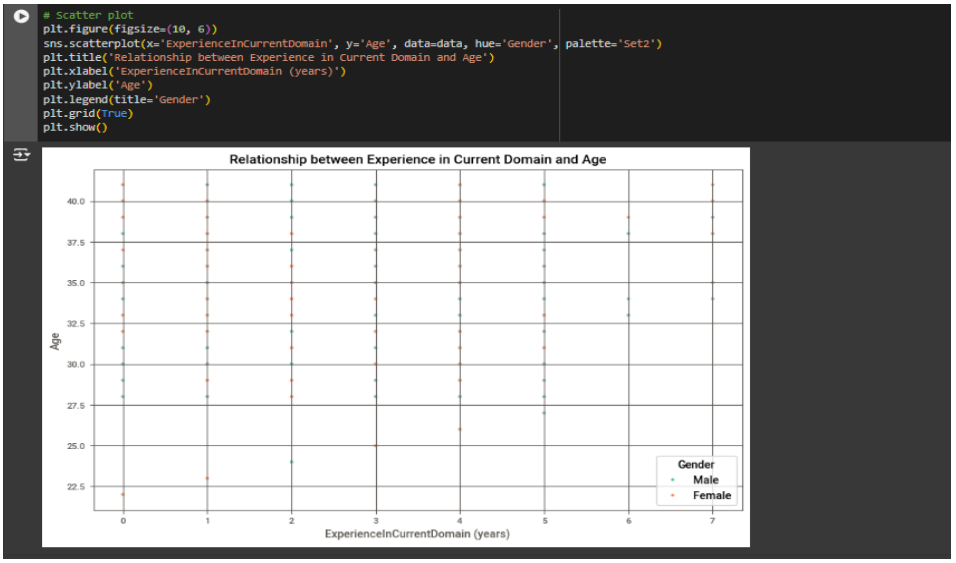


Gambar 4. 10 Visualization of attributes 1-3

Gambar 4. 11 Visualization of attributes 4-6

Gambar 4.12 Visualization of attributes 7-9

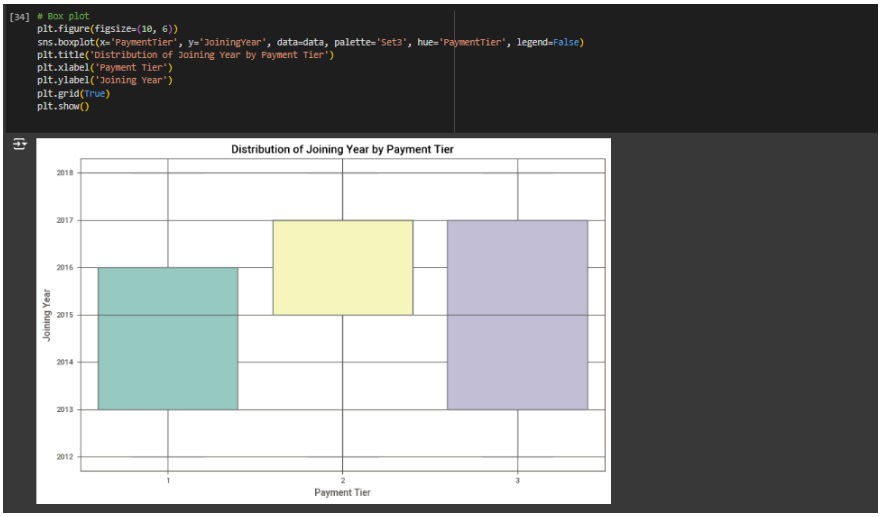
Scatter plot digunakan untuk memvisualisasikan hubungan antara dua variabel numerik. Dalam konteks ini, pengalaman dalam domain saat ini dan usia karyawan merupakan dua variabel yang mungkin saling berhubungan. Dengan melihat scatter plot, kita dapat mencari pola atau trend apakah ada kecenderungan bahwa karyawan yang lebih tua memiliki pengalaman lebih lama dalam domain tertentu, atau sebaliknya. Selain itu, penambahan dimensi gender sebagai hue memungkinkan kita untuk melihat apakah pola ini berbeda berdasarkan identitas gender.



Gambar 4. 13 Scatter Plot

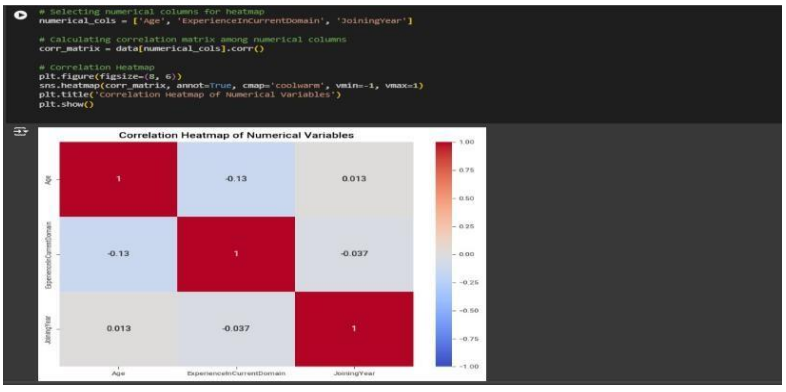
Box plot ideal untuk membandingkan distribusi variabel numerik (tahun

bergabung) dalam beberapa kategori (tier pembayaran). Dengan menggunakan box plot, kita bisa melihat perbedaan distribusi tahun bergabung antara berbagai tier pembayaran. Misalnya, apakah karyawan dengan tier pembayaran lebih tinggi cenderung bergabung dengan perusahaan pada tahun-tahun tertentu atau apakah distribusinya lebih merata.



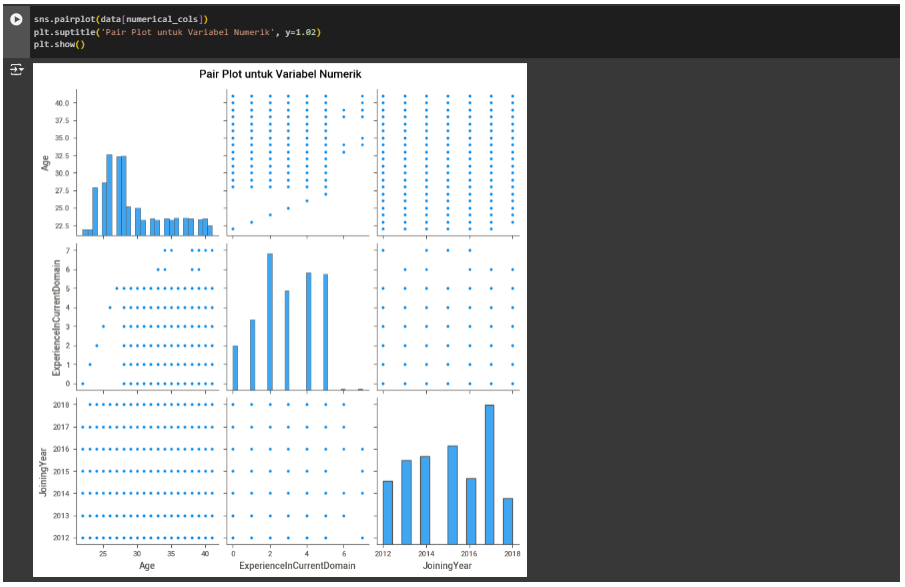
Gambar 4. 14 Box Plot

Heatmap korelasi digunakan untuk mengeksplorasi seberapa erat hubungan antara variabel numerik dalam dataset. Dalam konteks dataset karyawan, kita dapat melihat apakah ada korelasi yang signifikan antara variabel seperti usia, pengalaman dalam domain, dan tahun bergabung. Informasi ini penting untuk memahami bagaimana faktor-faktor ini berinteraksi dan mempengaruhi satu sama lain.



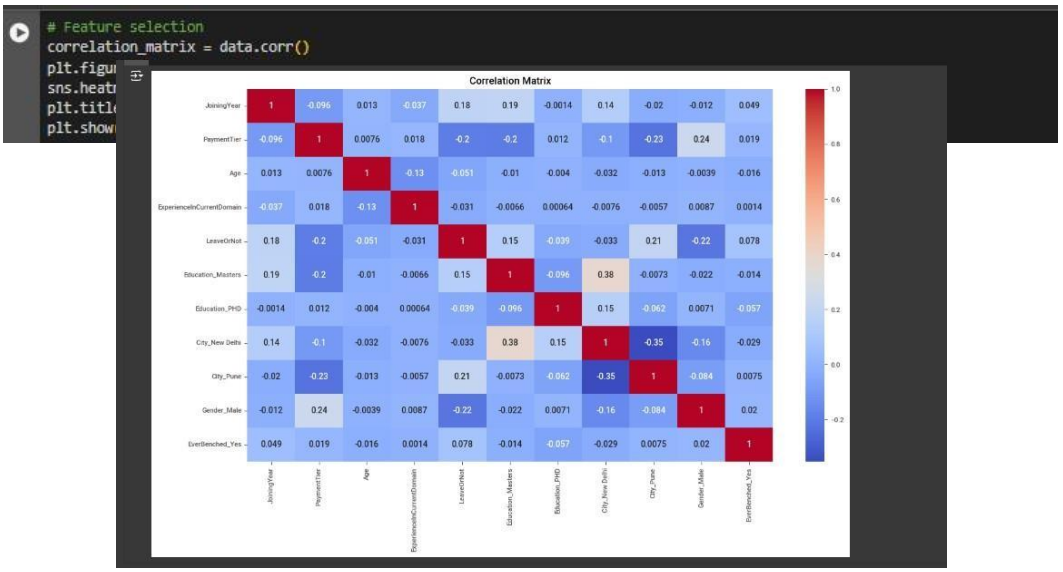
Gambar 4. 15 Heatmap

Pair plot berguna untuk mendapatkan gambaran yang komprehensif tentang hubungan antara semua pasangan variabel numerik dalam dataset. Dengan satu visualisasi, kita dapat melihat distribusi masing-masing variabel diagonal, sementara off-diagonal menunjukkan scatter plot dari pasangan variabel. Ini membantu dalam menemukan pola atau tren yang kompleks antara berbagai atribut numerik dalam dataset karyawan.



Gambar 4. 16 Pair Plot

Membuat matriks korelasi untuk memahami hubungan antara atribut. Korelasi ini menghasilkan nilai antara -1 dan 1, di mana 1 menunjukkan korelasi positif sempurna, -1 menunjukkan korelasi negatif sempurna, dan 0 menunjukkan tidak ada korela



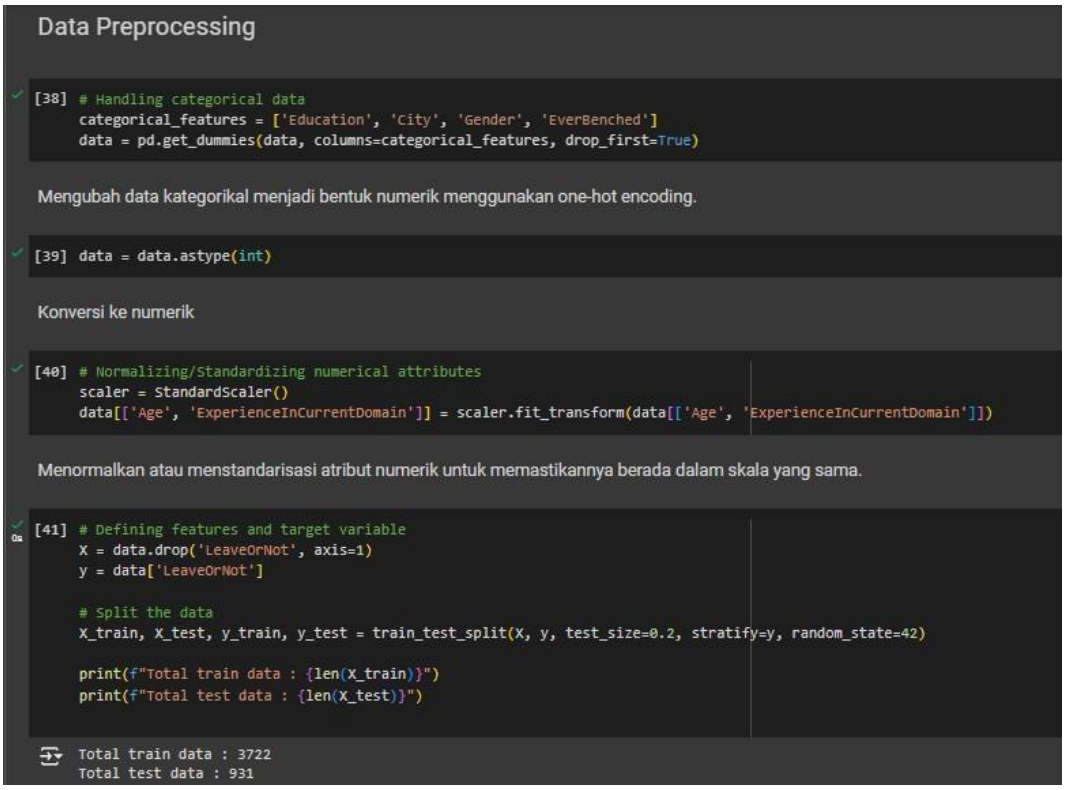
Gambar 4. 17 Correlation Matrix

4.1.6 Data Preprocessing

Dalam tahap ini melibatkan serangkaian langkah untuk mempersiapkan data sebelum digunakan dalam pemodelan. Langkah-langkah ini mencakup penanganan data kategorikal, normalisasi atribut numerik, serta pembagian data menjadi data pelatihan dan pengujian.

1. Menangani Data Kategorikal: Data kategorikal diubah menjadi variabel dummy menggunakan pd.get_dummies. Hal ini penting untuk memastikan bahwa model machine learning dapat memproses data kategorikal dengan benar. Dengan menggunakan pd.get_dummies, kita menghindari masalah multikolinearitas dengan menghapus salah satu kategori dari setiap variabel kategorikal (drop_first=True).
2. Normalisasi/Standarisasi Atribut Numerik: Atribut numerik seperti Age dan ExperienceInCurrentDomain dinormalisasi menggunakan StandardScaler untuk memastikan bahwa semua fitur berada dalam skala yang sama, yang penting untuk banyak algoritma machine learning.
3. Mendefinisikan Fitur dan Variabel Target: Memisahkan fitur (X) dan variabel target (y). Fitur adalah semua kolom kecuali LeaveOrNot, yang merupakan variabel target.
4. Membagi Data: Data dibagi menjadi data pelatihan (80%) dan data pengujian

(20%) menggunakan train_test_split dari scikit-learn. Pembagian ini dilakukan secara stratifikasi untuk memastikan distribusi kelas yang seimbang dalam set pelatihan dan pengujian.



Gambar 4. 18 Data Preprocessing

4.2 Modeling Machine Learning

Pada tahap ini, berbagai model machine learning dilatih dan dievaluasi untuk memprediksi apakah seorang karyawan akan meninggalkan perusahaan atau tidak (LeaveOrNot).

4.2.1 Feature Importance dengan Random Forest

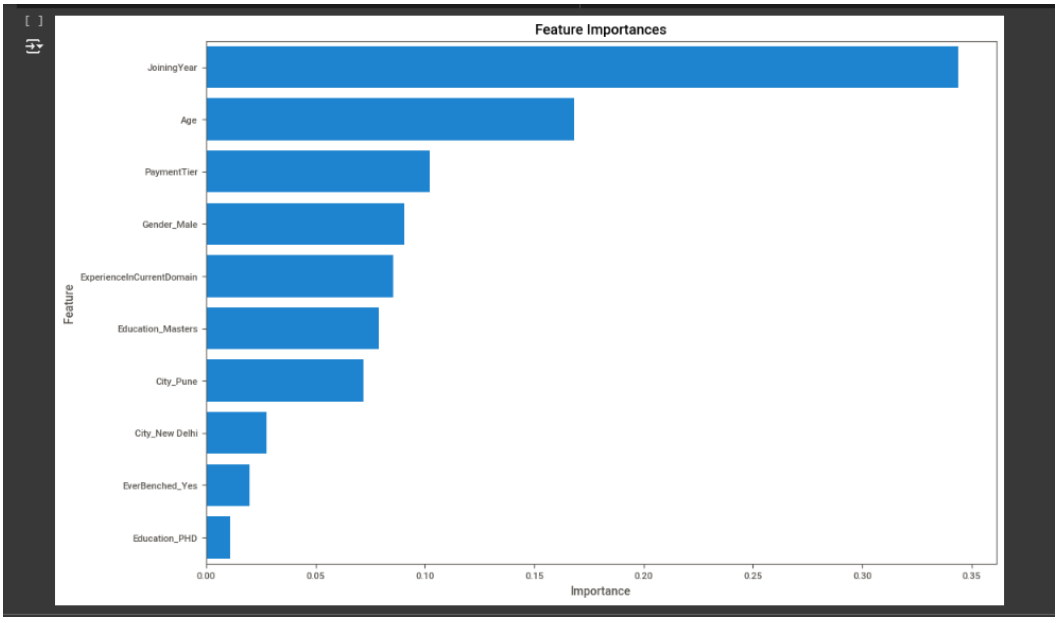
Model Random Forest dilatih untuk menentukan fitur mana yang paling penting dalam memprediksi variabel target. Random Forest adalah model ensemble yang menggunakan beberapa pohon keputusan untuk meningkatkan akurasi dan mengurangi overfitting.

Gambar 4. 19 Features Importance

1. Memilih Fitur Penting: Fitur yang paling penting dipilih berdasarkan nilai kepentingan fitur yang dihitung oleh model Random Forest. Hanya fitur-fitur yang paling penting yang digunakan untuk pelatihan model lebih lanjut.
2. Menampilkan Fitur Penting dari kode sebelumnya.

```
[ ]  # Display the feature importances visually
     plt.figure(figsize=(12, 8))
     sns.barplot(x=feature_importances['Importance'], y=feature_importances['Feature'])
     plt.title('Feature Importances')
     plt.show()

     print("\nFeature Importances:")
     print(feature_importances)
```



Gambar 4. 20 Visualization Features Importance

3. `feature_importances.head(7)` yaitu, mengambil 7 baris pertama dari feature_importances, yang berurutan dari yang paling penting ke yang kurang penting.
4. Setelah menentukan fitur-fitur teratas yang penting, langkah selanjutnya adalah mengambil hanya fitur-fitur ini dari data latih (X_train) dan data uji (X_test). Di sini, X_train_selected dan X_test_selected adalah data latih dan data uji yang telah direduksi, hanya mengandung fitur-fitur yang

terpilih (top_features). Ini bertujuan untuk membangun dan menguji model hanya dengan fitur-fitur yang dianggap penting, sehingga mempercepat proses pelatihan dan meningkatkan interpretasi model.

```
[ ]  # Select the top 7 important features
     top_features = feature_importances.head(7)['Feature']

[ ]  # Redefine X using only the top important features/feature selection
     X_train_selected = X_train[top_features]
     X_test_selected = X_test[top_features]
```

Gambar 4. 21 Selecting top 7 Important Features & redefine x

4.2.2 Random Forest Classification

1. Mengimpor, Melatih dan Memprediksi Model

   Kode ini menginisialisasi model Random Forest dengan parameter random_state=42 untuk memastikan hasil yang konsisten setiap kali model dijalankan. Kemudian Model Random Forest dilatih menggunakan data latih (X_train_selected untuk fitur dan y_train untuk label) dan setelah model dilatih, digunakan untuk memprediksi label dari data uji (X_test_selected).

```
[ ]  # Train and evaluate Random Forest model
     rf = RandomForestClassifier(random_state=42)
     rf.fit(X_train_selected, y_train)
     rf_pred = rf.predict(X_test_selected)
```

2. Model dievaluasi menggunakan beberapa metrik evaluasi:

   Accuracy: Proporsi prediksi yang benar dari keseluruhan prediksi.

   Precision: Proporsi prediksi positif yang benar dari semua prediksi positif.

   Recall: Proporsi kasus positif yang terdeteksi oleh model.

   F1 Score: Harmonic mean dari precision dan recall, memberikan gambaran keseimbangan antara keduanya.

```
# Evaluate model
rf_accuracy = accuracy_score(y_test, rf_pred)
rf_precision = precision_score(y_test, rf_pred)
rf_recall = recall_score(y_test, rf_pred)
rf_f1 = f1_score(y_test, rf_pred)
```

```
print(f"Random Forest Accuracy: {rf_accuracy:.2f}")
print(f"Random Forest Precision: {rf_precision:.2f}")
print(f"Random Forest Recall: {rf_recall:.2f}")
print(f"Random Forest F1 Score: {rf_f1:.2f}")

print("\nRandom Forest Classification Report:")
print(classification_report(y_test, rf_pred))
```

```
Random Forest Accuracy: 0.82
Random Forest Precision: 0.80
Random Forest Recall: 0.65
Random Forest F1 Score: 0.72

Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.91      0.87       611
           1       0.80      0.65      0.72       320

    accuracy                           0.82       931
   macro avg       0.82      0.78      0.79       931
weighted avg       0.82      0.82      0.82       931
```

3. Menampilkan hasil dari Model yang telah dievaluasi.

Gambar 4. 22 Random Forest Classification Report

Model Random Forest yang dilatih menunjukkan performa yang baik dengan accuracy sebesar 82%. Meskipun precision juga cukup tinggi (80%), recall untuk kelas positif (1) lebih rendah (65%), yang mengindikasikan model mungkin kurang mampu mendeteksi semua kasus positif. Namun, F1 score menunjukkan keseimbangan yang cukup baik antara precision dan recall.

4.2.3 Decision Tree Classification

Decision Tree Classifier

```
[ ]  # Train and evaluate Decision Tree model
     dt = DecisionTreeClassifier(random_state=42)
     dt.fit(X_train_selected, y_train)
     dt_pred = dt.predict(X_test_selected)

     # Evaluate Decision Tree model
     dt_accuracy = accuracy_score(y_test, dt_pred)
     dt_precision = precision_score(y_test, dt_pred)
     dt_recall = recall_score(y_test, dt_pred)
     dt_f1 = f1_score(y_test, dt_pred)

     print(f"Decision Tree Accuracy: {dt_accuracy:.2f}")
     print(f"Decision Tree Precision: {dt_precision:.2f}")
     print(f"Decision Tree Recall: {dt_recall:.2f}")
     print(f"Decision Tree F1 Score: {dt_f1:.2f}")

     print("\nDecision Tree Classification Report:")
     print(classification_report(y_test, dt_pred))
```

```
Decision Tree Accuracy: 0.80
Decision Tree Precision: 0.77
Decision Tree Recall: 0.62
Decision Tree F1 Score: 0.69

Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.90      0.86       611
           1       0.77      0.62      0.69       320

    accuracy                           0.80       931
   macro avg       0.79      0.76      0.77       931
weighted avg       0.80      0.80      0.80       931
```

Gambar 4. 22 Decision Tree Classification Report

Model Decision Tree yang dilatih menunjukkan performa yang cukup baik dengan accuracy sebesar 80%. Meskipun precision juga cukup tinggi (77%), recall untuk kelas positif (1) lebih rendah (62%), yang mengindikasikan model mungkin kurang mampu mendeteksi semua kasus positif. Namun, F1 score menunjukkan keseimbangan yang cukup baik antara precision dan recall.

4.2.4 XGBoost Classification

XGBoost Classification

```
[ ] #Train and evaluate XGBoost model
    xgb_model = xgb.XGBClassifier(random_state=42)
    xgb_model.fit(X_train_selected, y_train)
    xgb_pred = xgb_model.predict(X_test_selected)

    # Evaluate XGBoost model
    xgb_accuracy = accuracy_score(y_test, xgb_pred)
    xgb_precision = precision_score(y_test, xgb_pred)
    xgb_recall = recall_score(y_test, xgb_pred)
    xgb_f1 = f1_score(y_test, xgb_pred)

    print(f"XGBoost Accuracy: {xgb_accuracy:.2f}")
    print(f"XGBoost Precision: {xgb_precision:.2f}")
    print(f"XGBoost Recall: {xgb_recall:.2f}")
    print(f"XGBoost F1 Score: {xgb_f1:.2f}")

    print("\nXGBoost Classification Report:")
    print(classification_report(y_test, xgb_pred))
```

```
XGBoost Accuracy: 0.83
XGBoost Precision: 0.83
XGBoost Recall: 0.65
XGBoost F1 Score: 0.73

XGBoost Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.93      0.88       611
           1       0.83      0.65      0.73       320

    accuracy                           0.83       931
   macro avg       0.83      0.79      0.80       931
weighted avg       0.83      0.83      0.83       931
```

Gambar 4. 24 XGBoost Classification Report

Model XGBoost yang dilatih menunjukkan performa yang sangat baik dengan accuracy sebesar 83%. Precision juga tinggi (83%), namun recall untuk kelas positif (1) lebih rendah (65%), yang mengindikasikan model mungkin kurang mampu mendeteksi semua kasus positif. Namun, F1 score menunjukkan keseimbangan yang cukup baik antara precision dan recall.

Laporan klasifikasi memberikan gambaran detail tentang performa model pada setiap kelas dan keseluruhan dataset. Model XGBoost menunjukkan performa yang sedikit lebih baik dibandingkan dengan model Decision Tree dan Random Forest dalam hal akurasi dan precision.

4.3 Modeling Deep Learning

Pada tahap ini, model deep learning yakni ANN (Artificial Neural Networks) digunakan untuk pemodelan dan analisis 7 data penting untuk memprediksi apakah seorang karyawan akan meninggalkan perusahaan atau tidak (LeaveOrNot).

4.3.1 Encode Target

Encode Target merupakan encode target dengan format hot encode untuk neural network, untuk melengkapi bagian redefine X pada top features sebelum masuk ke ann. Encode menyesuaikan jaringan syaraf-menghindari asumsi yang salah, nantinya hot encode membantu dalam perhitungan fungsi loss (’categorical_crossentropy’), untuk ’to-categorical’ juga digunakan untuk mengubah label kelas target ke format one hot encode yang cocok digunakan di jaringan syaraf buatan.

```
[147] #------ANN CLASSIFICATION------
      # Encode TARGET variable for neural network
      y_train = to_categorical(y_train)
      y_test = to_categorical(y_test)
```

Gambar 4. 25 Encode Target DL

4.3.2 Define Model

Model Sequential yang kami buat terdiri dari beberapa layer. Dropout layer ditambahkan untuk mencegah overfitting dengan mengabaikan (drop) sejumlah unit acak selama pelatihan (untuk melihat overitting ada pada gambar overvitting).

1. Layer pertama
    1. Menambahkan dense layer pertama dengan 12 neuron.
    2. Menentukan jumlah fitur input yang diterima oleh layer pertama dari data pelatihan (7 data penting/feature).
    3. Mengunakan fungsi aktivasi ReLU untuk layer pertama.
    4. Menyisipkan layer drop dengan rasio 20% secara acak selama pelatihan

2. Layer kedua

   Menambahkan dense layer kedua dengan 8 neuron dan menguunakan fungsi aktivasi ReLU untuk layer kedua.

3. Layer ketiga

   Sebagai layer output dengan jumlah neuron sesuai dengan jumlah kelas target LeaveOrNot yaitu 2 neuron, menggunakan aktivation softmax yang cocok untuk masalah klasifikasi multikelas dan neuron lebih dari 1, yang nantinya menghasilkan distribusi probabilitas dari kelas yang berbeda.

```
[148] # Define the model with Dropout layers for press overvitting
      model = Sequential()
      # Adjust input_dim to selected features
      model.add(Dense(12, input_dim=X_train_selected.shape[1], activation='relu')) # layer pertama 12 bulatan/neuron
      model.add(Dropout(0.2))  # Adding dropout for regularization
      model.add(Dense(8, activation='relu')) # layer kedua 8 bulatan/neuron
      model.add(Dropout(0.2))  # Adding dropout for regularization
      model.add(Dense(y_train.shape[1], activation='softmax'))  # Output neurons equal to number of classes/layer ketiga 2 bulatan/neuron(2 label LeaveOrNot)
```

Gambar 4. 26 Define Model DL

4.3.3 Compile the keras model

Compile digunakan untuk Menyusun model keras yang telah didefinisikan sebelumnya agar siap dilatih(fit). Menggunkana loss untuk masalah dimana target variabel telah di encode ke bentuk one-hot encode, optimizer adam untuk meningkatkan hasil pelatihan, metric accuracy untuk mengevaluasi peforma model-membandingkan prediksi.

```
[149] # Compile the keras model
      model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy']) # lo
```

Gambar 4.27 Compile Model

4.3.4 Fit model dataset

Fit model diguakan untuk melatih (fit) model keras yang terlah kami definisikan dan dicompile menggunakan data pelatihan sebelumnya (‘X_train_selected’ dan ‘y_train’). Selama pelatihan model akan memperbarui

bobot untuk meminimalkan loss dan memantau overvitting (memvalidasi peforma setiap epoch menggunakan data tes).

```
[150] # Fit the model on the dataset
      model.fit(X_train_selected, y_train, epochs=50, batch_size=10, verbose=1, validation_data=(X_test_selected, y_test)) # validasi menggunakan data tes

Epoch 1/50
373/373 [==============================] - 2s 3ms/step - loss: 200.9516 - accuracy: 0.4997 - val_loss: 3.1366 - val_accuracy: 0.6563
Epoch 2/50
373/373 [==============================] - 1s 2ms/step - loss: 41.5462 - accuracy: 0.5357 - val_loss: 0.6387 - val_accuracy: 0.6563
Epoch 3/50
373/373 [==============================] - 1s 2ms/step - loss: 11.8316 - accuracy: 0.5454 - val_loss: 0.6896 - val_accuracy: 0.6026
Epoch 4/50
373/373 [==============================] - 1s 2ms/step - loss: 2.3486 - accuracy: 0.5712 - val_loss: 0.6620 - val_accuracy: 0.6563
Epoch 5/50
373/373 [==============================] - 1s 2ms/step - loss: 1.1581 - accuracy: 0.6182 - val_loss: 0.6447 - val_accuracy: 0.6563
Epoch 6/50
373/373 [==============================] - 1s 2ms/step - loss: 1.0060 - accuracy: 0.6263 - val_loss: 0.6481 - val_accuracy: 0.6563
Epoch 7/50
373/373 [==============================] - 1s 3ms/step - loss: 0.8371 - accuracy: 0.6338 - val_loss: 0.6436 - val_accuracy: 0.6563
Epoch 8/50
373/373 [==============================] - 1s 4ms/step - loss: 0.7576 - accuracy: 0.6292 - val_loss: 0.6438 - val_accuracy: 0.6563
Epoch 9/50
373/373 [==============================] - 1s 4ms/step - loss: 0.6994 - accuracy: 0.6402 - val_loss: 0.6470 - val_accuracy: 0.6563
Epoch 10/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6658 - accuracy: 0.6462 - val_loss: 0.6434 - val_accuracy: 0.6563
Epoch 11/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6614 - accuracy: 0.6472 - val_loss: 0.6503 - val_accuracy: 0.6563
Epoch 12/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6588 - accuracy: 0.6472 - val_loss: 0.6435 - val_accuracy: 0.6563
Epoch 13/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6590 - accuracy: 0.6440 - val_loss: 0.6449 - val_accuracy: 0.6563
Epoch 14/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6504 - accuracy: 0.6523 - val_loss: 0.6445 - val_accuracy: 0.6563
Epoch 15/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6538 - accuracy: 0.6464 - val_loss: 0.6455 - val_accuracy: 0.6563
Epoch 16/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6496 - accuracy: 0.6515 - val_loss: 0.6440 - val_accuracy: 0.6563
Epoch 17/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6488 - accuracy: 0.6507 - val_loss: 0.6448 - val_accuracy: 0.6563
Epoch 18/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6464 - accuracy: 0.6556 - val_loss: 0.6436 - val_accuracy: 0.6563
Epoch 19/50
373/373 [==============================] - 1s 2ms/step - loss: 0.6476 - accuracy: 0.6523 - val_loss: 0.6435 - val_accuracy: 0.6563
Epoch 20/50
```

Gambar 4. 28 Fit Model

4.3.5 Evaluate model

Digunakan untuk mengevaluasi(seberapa baik prediksi) peforma model yang telah dilatih pada data tes (’X_test_selected’ dan ‘y_test’) seperti menghitung loss, mengembalikan nilai loss(diabaikan) dan akurasi (disimpan), kemudia menampilkannya dalam bentuk sudah diformat dan dikonversi.

```
# Evaluate the model
_, accuracy = model.evaluate(X_test_selected, y_test)
print('Accuracy: %.2f' % (accuracy*100))

30/30 [==============================] - 0s 2ms/step - loss: 0.6435 - accuracy: 0.6563
Accuracy: 65.63
```

Gambar 4.29 Model Evaluatio

4.3.6 Predictions

Digunakan untuk memprediksi hasil pada data uji (X_test_selected) dengan model memproses 30 batch (karena memiliki sekitar 300 sampel dalam set uji dan ukuran batch default 10, yang menghasilkan sekitar 30 batch).

1. ‘model.predict(X_test_selected)’
    1. Untuk membuat prediksi pada data tes (X_test_selected).
    2. Menghasilkan array probabilitas tiap kelas di data sample.
    3. y_pred adalah hasil prediksi, berupa array dua dimensi dengan probabilitas untuk setiap kelas.
2. ’np.argmax(y_pred, axis=1)’
    1. Untuk mengambil indeks dari nilai maksimum sepanjang sumbu tertentu.
    2. • axis=1 berarti mencari indeks maksimum sepanjang baris, yang berarti kita mengambil kelas dengan probabilitas tertinggi untuk setiap sampel.
    3. • y_pred_class adalah array satu dimensi yang berisi kelas yang diprediksi untuk setiap sampel di data tes.
3. ‘np.argmax(y_test, axis=1)’
    1. Mirip dengan y_pred_class, np.argmax(y_test, axis=1) digunakan untuk mengambil indeks dari nilai maksimum sepanjang baris dalam y_test.
    2. y_test adalah array dua dimensi yang telah di-encode secara one-hot (mengembalikan kelas asli data tes).
    3. y_test_class adalah array satu dimensi yang berisi kelas sebenarnya untuk setiap sampel di data tes.

4.3.7 Confusion matrix

Confusion matrix digunakan untuk mengevaluasi kinerja model clasifikasi ann dari jumlah prediksi benar, prediksi salah, kemudian headmap memvisualkan setiap nilai untuk melihat pola dan tingkatan kebingungan. Berikut merupakan beberapa fungsi-fungsi codenya:

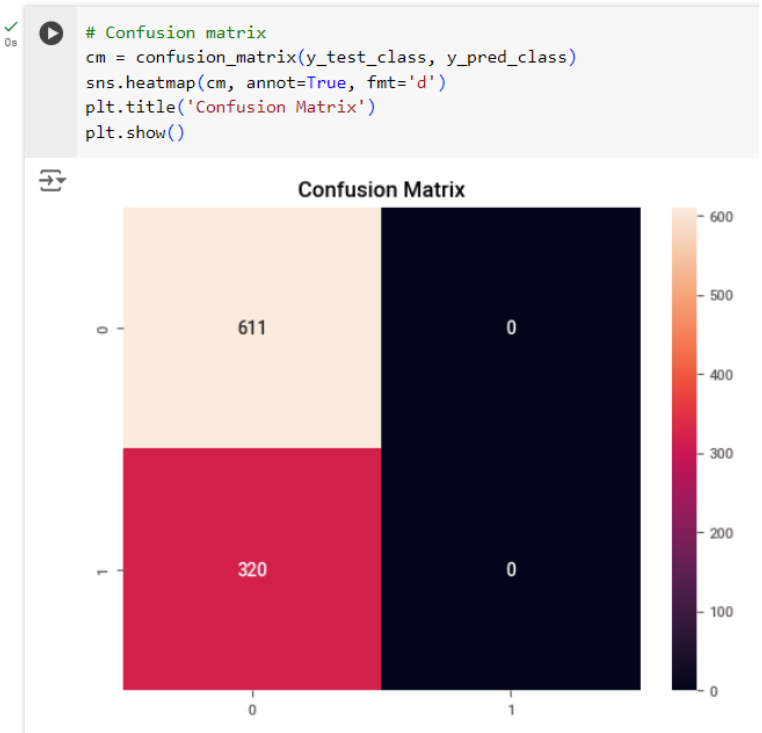
1. ‘confusion_matrix(y_test_class, y_pred_class)’

Fungsi ini untuk menghitung confusion matrix antara nilai aktual (y_test_class) dan nilai prediksi (y_pred_class).

2. ‘sns.heatmap’

   Fungsi dari library Seaborn untuk membuat heatmap.

   1. Cm adalah matrix yang dihasilkan dari confusion_matrix.
   2. ‘annot=True’ adalah parameter untuk menampilkan nilai di dalam sel heatmap.
   3. ’fmt='d'’ adalah parameter untuk format nilai di dalam sel (d sebagai integer).
   4. ‘plt.title('Confusion Matrix')’ untuk memberikan judul pada plot.
   5. ‘plt.show()’ untuk menampilkan plot heatmap confusion matrix.



Gambar 4. 30 Confusion Matrix

4.4 Optimasi Model

Optimasi model adalah proses penyesuaian hyperparameter suatu model untuk meningkatkan kinerja dan akurasi prediksi. Hyperparameter adalah

parameter yang nilainya ditentukan sebelum proses pelatihan dan tidak diubah selama pelatihan model. Contoh hyperparameter termasuk jumlah pohon dalam hutan untuk Random Forest, kedalaman maksimum pohon, dan lainnya. Tujuan dari optimasi model adalah menemukan kombinasi hyperparameter yang menghasilkan performa terbaik dari model pada data tertentu. Optimasi model penting karena:

a. Meningkatkan Akurasi: Membantu menemukan parameter yang memberikan hasil terbaik, meningkatkan akurasi prediksi.
b. Mengurangi Overfitting: Menyesuaikan parameter untuk menghindari overfitting, sehingga model lebih generalizable ke data baru.
c. Efisiensi: Membantu memilih parameter yang membuat model lebih efisien dalam hal komputasi dan waktu.

4.4.1 Optimasi Random Forest Model

Optimasi model Random Forest melibatkan pencarian kombinasi hyperparameter terbaik yang memberikan performa optimal. Salah satu teknik yang sering digunakan adalah Grid Search dengan Cross-Validation. Proses Optimasi dengan GridSearchCV

1. Menentukan Grid Hyperparameter: Mendefinisikan rentang nilai untuk setiap hyperparameter yang ingin diuji.
2. Inisialisasi GridSearchCV: Menggunakan GridSearchCV untuk menguji semua kombinasi hyperparameter yang ditentukan dalam grid.
3. Cross-Validation: Membagi data latih menjadi beberapa fold dan melatih model pada beberapa subset data untuk memastikan bahwa model tidak overfitting.
4. Evaluasi: Menggunakan metrik evaluasi (misalnya, akurasi) untuk memilih kombinasi hyperparameter terbaik.
5. Pelatihan Model Terbaik: Menggunakan hyperparameter terbaik untuk melatih model akhir pada seluruh data latih.
6. Prediksi dan Evaluasi: Mengevaluasi model terbaik pada data uji untuk memastikan performa yang baik.

```python
# Set hyperparameter grid
param_grid_rf = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Initialize GridSearchCV
grid_search_rf = GridSearchCV(estimator=rf, param_grid=param_grid_rf, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

# Perform grid search to find the best combination
grid_search_rf.fit(X_train, y_train)

# Print the best tuning results
print("Best Parameters for Random Forest:", grid_search_rf.best_params_)
print("Best Cross-Validation Accuracy for Random Forest:", grid_search_rf.best_score_)

# Evaluate the best model
rf_best = grid_search_rf.best_estimator_
rf_best.fit(X_train, y_train)
rf_pred = rf_best.predict(X_test)

# Evaluate the performance of the best model
print("\nRandom Forest Model Evaluation:")
print("Accuracy:", accuracy_score(y_test, rf_pred))
print("Precision:", precision_score(y_test, rf_pred))
print("Recall:", recall_score(y_test, rf_pred))
print("F1 Score:", f1_score(y_test, rf_pred))
print("\nClassification Report:")
print(classification_report(y_test, rf_pred))
```

Gambar 4. 31 Random Forest Model Optimization code

Hasil dari proses optimasi ini menunjukkan bahwa kombinasi hyperparameter terbaik untuk model Random Forest adalah:

max_depth: 20

min_samples_leaf: 4

min_samples_split: 10

n_estimators: 200

Model dengan kombinasi hyperparameter ini memiliki akurasi cross-validation terbaik sebesar 0.8547. Evaluasi akhir pada data uji menunjukkan akurasi sebesar 0.8561, precision sebesar 0.8875, recall sebesar 0.6656, dan F1 score sebesar 0.7607.

```
Fitting 5 folds for each of 81 candidates, totalling 405 fits
Best Parameters for Random Forest: {'max_depth': 20, 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
Best Cross-Validation Accuracy for Random Forest: 0.8546507180486398

Random Forest Model Evaluation:
Accuracy: 0.8560687432867884
Precision: 0.8875
Recall: 0.665625
F1 Score: 0.7607142857142856

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.96      0.90       611
           1       0.89      0.67      0.76       320

    accuracy                           0.86       931
   macro avg       0.87      0.81      0.83       931
weighted avg       0.86      0.86      0.85       931
```

Gambar 4. 32 Random Forest Model Evaluation

Optimasi model, khususnya Random Forest, adalah proses penting dalam machine learning untuk meningkatkan performa model. Dengan menggunakan GridSearchCV, kita dapat menemukan kombinasi hyperparameter yang menghasilkan model dengan kinerja terbaik, seperti yang ditunjukkan dalam contoh di atas. Optimasi ini membantu dalam mencapai akurasi yang lebih tinggi dan model yang lebih generalizable untuk data baru.

4.4.2 Optimasi Decision Tree Model

Optimasi model Decision Tree melibatkan penyesuaian hyperparameter untuk menemukan konfigurasi yang memberikan kinerja terbaik. Proses ini menggunakan GridSearchCV untuk mencari kombinasi hyperparameter terbaik berdasarkan evaluasi kinerja pada data latih dengan cross-validation. Laporan klasifikasi memberikan rincian berikut:

```
# Set hyperparameter grid
param_grid_dt = {
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Initialize GridSearchCV
grid_search_dt = GridSearchCV(estimator=dt, param_grid=param_grid_dt, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

# Perform grid search to find the best combination
grid_search_dt.fit(X_train, y_train)

# Print the best tuning results
print("Best Parameters for Decision Tree:", grid_search_dt.best_params_)
print("Best Cross-Validation Accuracy for Decision Tree:", grid_search_dt.best_score_)

# Evaluate the best model
dt_best = grid_search_dt.best_estimator_
dt_best.fit(X_train, y_train)
dt_pred = dt_best.predict(X_test)

# Evaluate the performance of the best model
print("\nDecision Tree Model Evaluation:")
print("Accuracy:", accuracy_score(y_test, dt_pred))
print("Precision:", precision_score(y_test, dt_pred))
print("Recall:", recall_score(y_test, dt_pred))
print("F1 Score:", f1_score(y_test, dt_pred))
print("\nClassification Report:")
print(classification_report(y_test, dt_pred))
```

Gambar 4. 33 Decision Tree Model Optimization

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Best Parameters for Decision Tree: {'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 2}
Best Cross-Validation Accuracy for Decision Tree: 0.8436364292415386

Decision Tree Model Evaluation:
Accuracy: 0.8259935553168636
Precision: 0.8558558558558559
Recall: 0.59375
F1 Score: 0.7011070110701106

Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.95      0.88       611
           1       0.86      0.59      0.70       320

    accuracy                           0.83       931
   macro avg       0.84      0.77      0.79       931
weighted avg       0.83      0.83      0.82       931
```

Gambar 4. 34 Decision Tree Model Evaluation

Dengan menggunakan GridSearchCV, kita berhasil menemukan kombinasi hyperparameter terbaik untuk model Decision Tree. Hasil evaluasi menunjukkan bahwa model yang dioptimalkan memiliki akurasi sebesar 82.60%. Precision untuk kelas positif (1) cukup tinggi (85.59%), tetapi recall lebih rendah (59.38%), yang mengindikasikan bahwa model masih mungkin kehilangan beberapa kasus positif. Namun, F1 score menunjukkan keseimbangan yang baik antara precision dan recall.

Optimasi model ini membantu meningkatkan performa model secara keseluruhan, yang terlihat dari peningkatan akurasi dan metrik evaluasi lainnya. Dengan menggunakan GridSearchCV, kita dapat memastikan bahwa model yang dihasilkan adalah yang terbaik untuk data yang diberikan.

4.4.3 Optimasi XGBoost Model

XGBoost (Extreme Gradient Boosting) adalah algoritma pembelajaran mesin yang kuat untuk tugas regresi dan klasifikasi. Ini adalah peningkatan dari teknik boosting dan terkenal karena efisiensi, kecepatan, dan kinerja prediksinya yang tinggi. Optimasi model XGBoost melibatkan penyesuaian hyperparameter untuk menemukan konfigurasi yang memberikan kinerja terbaik. Beberapa hyperparameter utama dalam XGBoost yang bisa dioptimalkan adalah:

1. learning_rate: Mengontrol kontribusi setiap pohon terhadap hasil akhir.
2. max_depth: Kedalaman maksimum pohon.
3. min_child_weight: Jumlah minimum berat sum anak yang diperlukan dalam sebuah node.

4. subsample: Proporsi sampel pelatihan yang digunakan untuk membangun masing-masing pohon.
5. colsample_bytree: Proporsi fitur yang dipilih secara acak untuk membangun masing-masing pohon.
6. gamma: Minimal pengurangan loss yang diperlukan untuk melakukan split pada node.

```python
# Set hyperparameter grid
param_grid_xgb = {
    'learning_rate': [0.05, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'min_child_weight': [1, 3, 5],
    'subsample': [0.8, 0.9, 1.0],
    'colsample_bytree': [0.8, 0.9, 1.0],
    'gamma': [0, 0.1, 0.2]
}
# Initialize GridSearchCV
grid_search_xgb = GridSearchCV(estimator=xgb_model, param_grid=param_grid_xgb, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

# Perform grid search to find the best combination
grid_search_xgb.fit(X_train, y_train)

# Print the best tuning results
print("Best Parameters for XGBoost:", grid_search_xgb.best_params_)
print("Best Cross-Validation Accuracy for XGBoost:", grid_search_xgb.best_score_)

# Evaluate the best model
xgb_best = grid_search_xgb.best_estimator_
xgb_best.fit(X_train, y_train)
xgb_pred = xgb_best.predict(X_test)

# Evaluate the performance of the best model
print("\nXGBoost Model Evaluation:")
print("Accuracy:", accuracy_score(y_test, xgb_pred))
print("Precision:", precision_score(y_test, xgb_pred))
print("Recall:", recall_score(y_test, xgb_pred))
print("F1 Score:", f1_score(y_test, xgb_pred))
print("\nClassification Report:")
print(classification_report(y_test, xgb_pred))
```

Gambar 4. 35 XGBoost Optimization Code

```
Fitting 5 folds for each of 729 candidates, totalling 3645 fits
Best Parameters for XGBoost: {'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.05, 'max_depth': 5, 'min_child_weight': 3, 'subsample': 1.0}
Best Cross-Validation Accuracy for XGBoost: 0.8562628996175219

XGBoost Model Evaluation:
Accuracy: 0.8517722878625135
Precision: 0.8729508196721312
Recall: 0.665625
F1 Score: 0.7553191489361701

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.95      0.89       611
           1       0.87      0.67      0.76       320

    accuracy                           0.85       931
   macro avg       0.86      0.81      0.82       931
weighted avg       0.85      0.85      0.85       931
```

Gambar 4.36 XGBoost Model Evaluation

Dengan menggunakan GridSearchCV, kita berhasil menemukan kombinasi hyperparameter terbaik untuk model XGBoost. Hasil evaluasi menunjukkan bahwa model yang dioptimalkan memiliki akurasi sebesar 85.18%. Precision untuk kelas

positif (1) cukup tinggi (87.30%), tetapi recall lebih rendah (66.56%), yang mengindikasikan bahwa model masih mungkin kehilangan beberapa kasus positif. Namun, F1 score menunjukkan keseimbangan yang baik antara precision dan recall.

Optimasi model ini membantu meningkatkan performa model secara keseluruhan, yang terlihat dari peningkatan akurasi dan metrik evaluasi lainnya. Dengan menggunakan GridSearchCV, kita dapat memastikan bahwa model yang dihasilkan adalah yang terbaik untuk data yang diberikan.

4.5 Penjelasan Hasil

4.5.1 Model Machine Learning

Dalam machine learning ini, kita telah membangun dan mengoptimalkan tiga model klasifikasi berbeda: Random Forest, Decision Tree, dan XGBoost. Masing-masing model telah dievaluasi berdasarkan berbagai metrik seperti akurasi, precision, recall, dan F1 score. Berikut adalah analisis hasil dari ketiga model tersebut dan perbandingannya.

Model	Accuracy	Precision	Recall	F1 Score	Best Parameters
Random Forest	0.8561	0.8875	0.6656	0.7607	{'max_depth': 20, 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200}
Decision Tree	0.8260	0.8559	0.5938	0.7011	{'max_depth': 10, 'min_samples_leaf': 2,

					'min_samples_split': 2}
XGBoost	0.8518	0.8730	0.6656	0.7553	{'colsample_bytree': 1.0, 'gamma': 0, 'learning_rate': 0.05, 'max_depth': 5, 'min_child_weight': 3, 'subsample': 1.0}

Tabel 4. 1 Analisis Hasil Model Machine Learning

Analisis Perbandingan Model

1. Random Forest

   Akurasi: 85.61%

   Precision: 88.75%

   Recall: 66.56%

   F1 Score: 76.07%

   Kelebihan:

   1. Tingkat akurasi yang paling tinggi dibandingkan model lain.
   2. Precision yang tinggi menunjukkan model ini sangat baik dalam mengidentifikasi kelas positif.

   Kekurangan:

   1) Recall yang lebih rendah menunjukkan bahwa model ini mungkin kehilangan beberapa instance positif.
   2) Model cenderung lebih kompleks dan memerlukan lebih banyak sumber daya komputasi.

2. Decision Tree

   Akurasi: 82.60%

   Precision: 85.59%

   Recall: 59.38%

   F1 Score: 70.11%

   Kelebihan:

1. Lebih mudah untuk diinterpretasi dan diimplementasikan.
2. Memiliki precision yang cukup baik.

Kekurangan:

1. Recall yang paling rendah menunjukkan model ini cenderung gagal menangkap instance positif.
2. Akurasi lebih rendah dibandingkan dengan Random Forest dan XGBoost.

3. XGBoost

Akurasi: 85.18%

Precision: 87.30%

Recall: 66.56%

F1 Score: 75.53%

Kelebihan:

1. Kombinasi antara akurasi dan precision yang tinggi.
2. Sangat baik dalam menangani dataset yang besar dan kompleks.

Kekurangan:

1. Memerlukan tuning hyperparameter yang lebih ekstensif dan lebih lama untuk dilatih.
2. Recall yang sama dengan Random Forest menunjukkan masih ada ruang untuk perbaikan dalam menangkap instance positif.

4. Kesimpulan
    1. Akurasi: Random Forest memiliki akurasi tertinggi (85.61%), diikuti oleh XGBoost (85.18%), dan Decision Tree (82.60%).
    2. Precision: Random Forest memiliki precision tertinggi (88.75%), menunjukkan kemampuan yang sangat baik dalam memprediksi kelas positif dengan benar.
    3. Recall: Random Forest dan XGBoost memiliki recall yang sama (66.56%), menunjukkan bahwa kedua model ini memiliki tingkat kesalahan yang sama dalam menangkap instance positif.
    4. F1 Score: Random Forest memiliki F1 score tertinggi (76.07%), menunjukkan keseimbangan terbaik antara precision dan recall.

5. Rekomendasi
    1. Random Forest: Direkomendasikan untuk digunakan jika tujuannya adalah untuk mendapatkan akurasi dan precision yang tinggi, meskipun harus mengorbankan interpretabilitas dan waktu komputasi.
    2. Decision Tree: Cocok untuk keperluan interpretasi yang mudah dan implementasi cepat, meskipun dengan performa yang lebih rendah.
    3. XGBoost: Pilihan yang baik untuk dataset yang besar dan kompleks dengan akurasi yang hampir setara dengan Random Forest, namun membutuhkan tuning hyperparameter yang lebih ekstensif.

4.5.2 Model Deep Learning

Melakukan modeling deep learning menggunakan model ANN secara umum menunjukan bahwa model mampu memberikan presentase memprediksi benar sebesar 65% dari 300 sample data uji dengan menggunakan 7 data penting dan LeaveOrNot sebagai target. Confusion matrix menjelaskan bahwa:

1. terdapat 320 karyawan secara actual menetap namun diprediksi pergi.
2. terdapat 611 karyawan secara actual pergi namun diprediksi pergi.
3. terdapat 0 karyawan secara aktual menetap namun diprediksi menetap.
4. terdapat 0 karyawan secara actual pergi namun secara prediksi menetap.

Jadi dapat disimpulkan bahwa 7 data penting yang digunakan mampu mempengaruhi pengambilan keputusan karyawan dalam mengambil keputusan untuk menetap atau pergi, namun dapat dipastikan dengan 65% kebenaran bahwa 611 karyawan memilih pergi.

BAB V
PENUTUP

4.1 Kesimpulan

Mengelola dan mempertahankan karyawan merupakan tantangan besar bagi perusahaan, dan penelitian ini menunjukkan bahwa analisis data menggunakan pendekatan machine learning dan deep learning sangat bermanfaat dalam mengatasi tantangan ini. Dengan memanfaatkan dataset yang terstruktur dan menerapkan berbagai teknik analisis, penelitian ini berhasil mengidentifikasi faktor-faktor kunci yang mempengaruhi keputusan karyawan untuk meninggalkan perusahaan. Wawasan yang dihasilkan dari analisis ini dapat digunakan oleh perusahaan untuk mengembangkan strategi manajemen karyawan yang lebih efektif, meningkatkan manajemen sumber daya manusia, serta meningkatkan kepuasan dan kinerja karyawan secara keseluruhan.

Pemilihan algoritma machine learning dan deep learning yang tepat menjadi faktor penting dalam memastikan akurasi prediksi yang baik. Penelitian ini membuktikan bahwa setiap algoritma memiliki kelebihan dan kelemahan masing-masing yang perlu dipertimbangkan dengan cermat dalam konteks pengaplikasiannya di industri. Model Random Forest terbukti memberikan performa terbaik dalam memprediksi turnover karyawan dibandingkan dengan algoritma lain seperti Decision Tree, XGBoost, dan Artificial Neural Network (ANN).

Secara keseluruhan, penelitian ini menunjukkan potensi besar dari penerapan teknik-teknik analisis data modern dalam manajemen SDM. Dengan menggunakan hasil analisis untuk membuat keputusan yang berbasis data, perusahaan dapat mengambil tindakan preventif yang lebih efektif untuk mengurangi tingkat turnover, meningkatkan retensi karyawan, dan menciptakan lingkungan kerja yang lebih kondusif.

4.2 Saran

1. Pengembangan Model Lebih Lanjut: Disarankan untuk melakukan pengembangan lebih lanjut pada model yang telah dibangun dengan menambahkan lebih banyak data dan fitur yang relevan, serta melakukan tuning hyperparameter yang lebih mendalam untuk meningkatkan akurasi prediksi.
2. Penelitian Lanjutan: Penelitian lebih lanjut disarankan untuk mengeksplorasi penggunaan algoritma lain dan teknik ensemble learning untuk meningkatkan performa prediksi. Selain itu, analisis longitudinal untuk memahami tren dan pola jangka panjang terkait turnover karyawan juga dapat memberikan wawasan yang berharga.

DAFTAR PUSTAKA

Ramadani, D., Bukhari, F., & Julianto, M. T. (2023). Metode Random Forest dan XGBoost: Studi Kasus Prediksi Arah Penutupan Harga Saham. Institut Pertanian Bogor.

WIRE, Giluwi. PREDIKSI KEMISKINAN DI PROVINSI PAPUA MENGGUNAKAN ALGORITMA NEURAL NETWORK. Artikel Sistem Informasi Universitas Muhammadiyah Gorontalo, 2023, 1-10.

RAHANRA, Musa Hendri Janto, et al. Analisa Kelulusan Mahasiswa Teknik Informatika Tepat Waktu Menggunakan Algoritma Artificial Neural Network (ANN). JURNAL FATEKSA: Jurnal Teknologi dan Rekayasa, 2022, 7.1: 1-11.

PRADANA, David, et al. Klasifikasi Penyakit Jantung Menggunakan Metode Artificial Neural Network. Indonesian Journal of Data and Science, 2022, 3.2: 55-60.

Universitas Gadjah Mada. (2021). Penerapan Algoritma XGBoost dan Random Forest untuk Estimasi Porositas Efektif Data Log Sumur.