

**IMPLEMENTASI ALGORITMA *RANDOM FOREST* PADA PREDIKSI  
PENURUNAN JUMLAH KARYAWAN**

**SKRIPSI**

Oleh:  
**EKA MULYANINGRUM**  
**NIM. 200605110043**



**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM  
MALANG  
2026**

**IMPLEMENTASI ALGORITMA *RANDOM FOREST* PADA PREDIKSI  
PENURUNAN JUMLAH KARYAWAN**

**SKRIPSI**

Diajukan kepada:  
Universitas Islam Negeri Maulana Malik Ibrahim Malang  
Untuk memenuhi Salah Satu Persyaratan dalam  
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:  
**EKA MULYANINGRUM**  
**NIM. 200605110043**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM  
MALANG  
2026**

## HALAMAN PERSETUJUAN

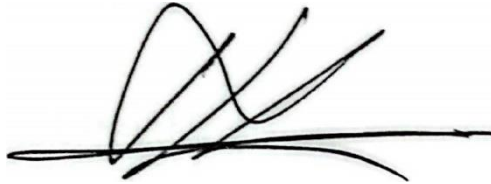
### IMPLEMENTASI ALGORITMA *RANDOM FOREST* PADA PREDIKSI PENURUNAN JUMLAH KARYAWAN

#### SKRIPSI

Oleh:  
**EKA MULYANINGRUM**  
**NIM. 200605110043**

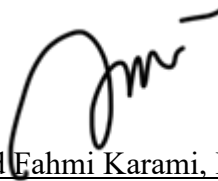
Telah Diperiksa dan Disetujui untuk Diuji:  
Tanggal: 24 April 2026

Pembimbing I,



Supriyono, M. Kom  
NIP. 19841010 201903 1 012

Pembimbing II,



Ahmad Fahmi Karami, M. Kom  
NIP. 19870909 202012 1 001

Mengetahui,  
Ketua Program Studi Teknik Informatika  
Fakultas Sains dan Teknologi  
Universitas Islam Negeri Maulana Malik Ibrahim Malang



Supriyono, M. Kom  
NIP. 19841010 201903 1 012

## HALAMAN PENGESAHAN

### IMPLEMENTASI ALGORITMA *RANDOM FOREST* PADA PREDIKSI PENURUNAN JUMLAH KARYAWAN

#### SKRIPSI

Oleh:  
**EKA MULYANINGRUM**  
**NIM. 200605110043**

Telah Dipertahankan di Depan Dewan Penguji Skripsi  
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan  
Untuk Memperoleh Gelar Sarjana Komputer ( S.Kom )  
Tanggal: 19 Juni 2026

#### Susunan Dewan Penguji

|                     |                                                                       |
|---------------------|-----------------------------------------------------------------------|
| Ketua Penguji       | : <u>Prof. Dr. Muhammad Faisal, M.T</u><br>NIP. 19740510 200501 1 007 |
| Anggota Penguji I   | : <u>Ashri Shabrina Afrah, M.T</u><br>NIP. 19900430 202012 2 003      |
| Anggota Penguji II  | : <u>Supriyono, M.Kom</u><br>NIP. 19841010 201903 1 012               |
| Anggota Penguji III | : <u>Ahmad Fahmi Karami, M.Kom</u><br>NIP. 19870909 202012 1 001      |

(  
(  
(  
(

Mengetahui dan Mengesahkan,  
Ketua Program Studi Teknik Informatika  
Fakultas Sains dan Teknologi  
Universitas Islam Negeri Maulana Malik Ibrahim Malang



Supriyono, M. Kom  
NIP. 19841010 201903 1 012

## PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : Eka Mulyaningrum

NIM : 200605110043

Fakultas / Prodi : Sains dan Teknologi / Teknik Informatika

Judul Skripsi : Implementasi Algoritma *Random Forest* Pada Penurunan  
Jumlah Karyawan

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 19 Juni 2026

Yang membuat pernyataan,

A handwritten signature in black ink is written over a yellow revenue stamp. The stamp features the Garuda Pancasila emblem, the text '1000', 'METERAI TEMPEL', and the serial number '0A738AOX048594891'.

Eka Mulyaningrum

NIM. 200605110043

## MOTTO

*“Tidakkah kamu penasaran dengan episode selanjutnya dari hidupmu?, kadang episode yang diharapkan memang sering kali sulit ditebak. Terus berjalan, meski pelan, tidak akan membuat kamu menjadi orang yang gagal”*

*“Takdirmu tidak akan tertukar atau terambil oleh orang lain, dan yang menjadi milikmu akan selalu menemukan jalannya sendiri untuk kembali kepadamu”*

(Ali bin Abi Thalib)

## HALAMAN PERSEMBAHAN

Dengan penuh rasa syukur, skripsi ini penulis persembahkan kepada kedua orang tua tercinta, ibuk dan bapak yang senantiasa menjadi tempat pulang, sumber kekuatan, serta alasan terbesar bagi penulis untuk terus berjuang. Terima kasih atas doa yang tidak pernah terputus, kasih sayang yang tiada batas, dukungan yang tulus, serta keyakinan dan kepercayaan yang selalu diberikan kepada penulis dalam setiap langkah menggapai cita-cita.

Skripsi ini juga penulis persembahkan kepada seluruh pihak yang telah hadir dan memberikan kebaikan dalam perjalanan penelitian ini, termasuk bapak supri, bapak fahmi, prof faisal, ibu ashri, bapak asrory, ibu yudi, juga adek, dan teman-teman semuanya, serta orang-orang yang memberikan bantuannya, baik melalui dukungan moral, material, maupun doa yang tulus. Semoga setiap bantuan, ketulusan, dan kebaikan yang telah diberikan mendapat balasan terbaik dari Allah *Subhānahu wa Ta'ālā*. Aamiin.

## KATA PENGANTAR

Segala puji hanya milik Allah *Subhānahu wa Ta‘ālā* atas segala nikmat, rahmat dan kasih sayang-Nya, sehingga penulis diberikan kemudahan dalam menyelesaikan skripsi yang berjudul “Implementasi Algoritma Random Forest Pada Penurunan Jumlah Karyawan”. Shalawat dan salam senantiasa terlimpah kepada Nabi Muhammad *Ṣallallāhu ‘Alaihi Wasallam*, beserta keluarga, sahabat, dan seluruh umatnya hingga akhir zaman. Semoga kita semua termasuk golongan yang memperoleh syafaat beliau di hari kiamat kelak, Aamiin. Dengan penuh rasa syukur, penulis menyampaikan terima kasih yang sebesar-besarnya kepada seluruh pihak yang telah memberikan dukungan, doa, serta motivasi, sehingga skripsi ini dapat terselesaikan dengan baik. Ucapan terima kasih ini penulis sampaikan kepada:

1. Prof. Dr. Hj. Ilfi Nurdiana, M.Si., CAHRM., CRMP, selaku Rektor Universitas Islam Negeri Malang.
2. Dr. Agus Mulyono, M.Kes, selaku Dekan Fakultas Saintek Universitas Islam Negeri Malang.
3. Supriyono, M. Kom., selaku Ketua program Studi Teknik Informatika Universitas Islam Negeri Malang dan juga selaku pembimbing I yang telah memberikan banyak bantuan, arahan, dan tidak lelah memberikan semangat dan dukungan kepada penulis, sehingga bisa menuntaskan skripsi ini.

4. Ahmad Fahmi Karami, M.Kom., selaku pembimbing II, yang memberikan banyak bantuan, dan arahan dalam menuntaskan skripsi ini, serta memberikan banyak ilmu baru bagi penulis.
5. Prof. Dr. Muhammad Faisal, M.T., selaku dosen ketua penguji dan Ashri Shabrina Afrah, M.T., selaku dosen penguji I yang telah meluangkan waktu untuk menguji, memberikan kritik, saran, dan masukan yang konstruktif dengan penuh ketulusan, sehingga penulis dapat menyempurnakan dan menyelesaikan skripsi ini dengan baik.
6. Nia Faricha S, Si., selaku admin Program Studi Teknik Informatika yang selalu sabar memberikan informasi, membantu, dan memberikan arahan selama perkuliahan dan proses penulisan skripsi ini.
7. Segenap dosen, laboran, jajaran staff, dan mahasiswa Program Studi Teknik Informatika yang telah memberikan banyak dukungan dan bimbingan selama pengerjaan skripsi ini.
8. Ibuk, bapak dan adek, yang sangat penulis cintai dan banggakan, ibu yudi sekeluarga, bapak asrory, serta saudara-saudara hingga teman-teman yang selalu memberikan dukungan dan motivasi untuk terus berusaha, serta untaian doa yang tidak pernah putus agar penulis dapat menuntaskan skripsi ini dengan baik dan lancar.

Malang, 19 Juni 2026

Penulis

## DAFTAR ISI

|                                                        |             |
|--------------------------------------------------------|-------------|
| <b>HALAMAN PERSETUJUAN .....</b>                       | <b>iii</b>  |
| <b>HALAMAN PENGESAHAN.....</b>                         | <b>iv</b>   |
| <b>PERNYATAAN KEASLIAN TULISAN .....</b>               | <b>v</b>    |
| <b>MOTTO .....</b>                                     | <b>vi</b>   |
| <b>HALAMAN PERSEMBAHAN .....</b>                       | <b>vii</b>  |
| <b>KATA PENGANTAR.....</b>                             | <b>viii</b> |
| <b>DAFTAR ISI.....</b>                                 | <b>x</b>    |
| <b>DAFTAR GAMBAR.....</b>                              | <b>xii</b>  |
| <b>DAFTAR TABEL .....</b>                              | <b>xiii</b> |
| <b>ABSTRAK .....</b>                                   | <b>xv</b>   |
| <b>ABSTRACT .....</b>                                  | <b>xvi</b>  |
| <b>البحث مستخلص.....</b>                               | <b>xvii</b> |
| <b>BAB I PENDAHULUAN.....</b>                          | <b>1</b>    |
| 1.1 Latar Belakang .....                               | 1           |
| 1.2 Pernyataan Masalah .....                           | 6           |
| 1.3 Batasan Masalah.....                               | 6           |
| 1.4 Tujuan Penelitian .....                            | 7           |
| 1.5 Manfaat Penelitian .....                           | 7           |
| <b>BAB II STUDI PUSTAKA .....</b>                      | <b>8</b>    |
| 2.1 Penurunan Jumlah Karyawan.....                     | 8           |
| 2.2 Penelitian Terkait .....                           | 9           |
| 2.3 Seleksi Fitur dengan Metode <i>Filtering</i> ..... | 16          |
| 2.3.1 <i>Pearson Correlation</i> .....                 | 16          |
| 2.3.2 <i>Analysis of Variance</i> (ANOVA).....         | 17          |
| 2.4 <i>Balancing</i> Data.....                         | 19          |
| 2.5 Algoritma Random Forest.....                       | 21          |
| 2.6 Evaluasi Model.....                                | 25          |
| <b>BAB III DESAIN DAN IMPLEMENTASI .....</b>           | <b>28</b>   |
| 3.1 Tahapan Penelitian .....                           | 28          |
| 3.2 Pengumpulan Data .....                             | 28          |
| 3.3 Desain Sistem.....                                 | 31          |
| 3.4 Preprocessing .....                                | 32          |
| 3.4.1 <i>Exploratory Data Analysis</i> (EDA).....      | 32          |
| 3.4.2 Transformasi Data .....                          | 37          |
| 3.4.3 Analisis Korelasi ( <i>Pearson</i> ) .....       | 38          |
| 3.4.4 Normalisasi Data .....                           | 40          |
| 3.4.5 Seleksi Data (F-Test).....                       | 41          |
| 3.4.6 <i>Balancing</i> Data .....                      | 44          |
| 3.4.7 Splitting Data.....                              | 47          |
| 3.5 Algoritma Random Forest.....                       | 48          |
| 3.6 Evaluasi Model.....                                | 53          |
| 3.6.1 <i>Confusion Matrix</i> .....                    | 53          |
| 3.6.2 Validasi Silang.....                             | 55          |

|                                                                       |            |
|-----------------------------------------------------------------------|------------|
| 3.7 Skenario Pengujian.....                                           | 56         |
| <b>BAB IV HASIL DAN PEMBAHASAN .....</b>                              | <b>59</b>  |
| 4.1 Hasil Pengujian .....                                             | 59         |
| 4.2 <i>Pearson Correlation</i> .....                                  | 60         |
| 4.3 ANOVA .....                                                       | 63         |
| 4.4 SMOTE .....                                                       | 65         |
| 4.5 Random Forest .....                                               | 69         |
| 4.6 Splitting Data .....                                              | 73         |
| 4.7 Hasil Uji Coba.....                                               | 74         |
| 4.7.1 Hasil Uji Skenario 1 .....                                      | 74         |
| 4.7.2 Hasil Uji Skenario 2 .....                                      | 76         |
| 4.7.3 Hasil Uji Skenario 3 .....                                      | 77         |
| 4.7.4 Hasil Uji Skenario 4 .....                                      | 79         |
| 4.7.5 Hasil Uji Skenario 5 .....                                      | 80         |
| 4.7.6 Hasil Uji Skenario 6 .....                                      | 82         |
| 4.7.7 Hasil Uji Skenario 7 .....                                      | 83         |
| 4.7.8 Hasil Uji Skenario 8 .....                                      | 85         |
| 4.7.9 Hasil Uji Skenario 9 .....                                      | 86         |
| 4.7.10 Hasil Uji Skenario 10 .....                                    | 88         |
| 4.7.11 Hasil Uji Skenario 11 .....                                    | 89         |
| 4.7.12 Hasil Uji Skenario 12 .....                                    | 91         |
| 4.8 Analisis Hasil Uji Coba.....                                      | 92         |
| 4.8.1 Analisis Evaluasi Matrik Akurasi.....                           | 92         |
| 4.8.2 Analisis Evaluasi Matrik Presisi.....                           | 94         |
| 4.8.3 Analisis Evaluasi Matrik <i>Recall</i> .....                    | 95         |
| 4.8.4 Analisis Evaluasi Matrik F1-Score.....                          | 97         |
| 4.9 Pembahasan.....                                                   | 98         |
| 4.9.1 Pengaruh Variasi Hyperparameter Random Forest.....              | 98         |
| 4.9.2 Perbandingan Train-Test Split dan K-Fold Cross Validation ..... | 99         |
| 4.9.3 Analisis Skenario Tanpa Balancing Data (Skenario 1–6) .....     | 99         |
| 4.9.4 Pengaruh Balancing Data Menggunakan SMOTE.....                  | 100        |
| 4.9.5 Pengaruh Seleksi Fitur Setelah Penerapan SMOTE.....             | 102        |
| 4.10 Integrasi Islam.....                                             | 102        |
| <b>BAB V KESIMPULAN DAN SARAN .....</b>                               | <b>106</b> |
| 5.1 Kesimpulan .....                                                  | 106        |
| 5.2 Saran.....                                                        | 108        |
| <b>DAFTAR PUSTAKA</b>                                                 |            |

## DAFTAR GAMBAR

|                                                                          |    |
|--------------------------------------------------------------------------|----|
| Gambar 2.1 Desain model Random Forest .....                              | 22 |
| Gambar 2.2 Cara kerja validasi silang .....                              | 24 |
| Gambar 3.1 Tahapan penelitian.....                                       | 28 |
| Gambar 3.2 Desain sistem pada tahapan pembangunan model prediksi .....   | 31 |
| Gambar 3.3 Diagram batang dalam mendeteksi outliers .....                | 34 |
| Gambar 3.4 Diagram pada distribusi fitur MaritalStatus .....             | 35 |
| Gambar 3.5 Diagram pada distribusi fitur JobRole .....                   | 36 |
| Gambar 3.6 Diagram pada distribusi fitur Age .....                       | 36 |
| Gambar 3.7 Hasil perancangan pohon keputusan .....                       | 52 |
| Gambar 3.8 Implementasi K-Fold CV .....                                  | 56 |
| Gambar 4.1 Kode untuk melihat korelasi antar fitur.....                  | 61 |
| Gambar 4.2 Kode untuk mengidentifikasi hubungan dengan fitur target..... | 63 |
| Gambar 4.3 Hasil Uji Fitur Pada ANOVA .....                              | 64 |
| Gambar 4.4 Identifikasi data mayoritas dan minoritas .....               | 66 |
| Gambar 4.5 Membuat data sintetik .....                                   | 67 |
| Gambar 4.6 Distribusi Data .....                                         | 68 |
| Gambar 4.7 Gini Impurity.....                                            | 69 |
| Gambar 4.8 Threshold.....                                                | 70 |
| Gambar 4.9 Decision Tree .....                                           | 71 |
| Gambar 4.10 Random Forest .....                                          | 72 |
| Gambar 4.11 Penggunaan Fungsi Random Forest.....                         | 73 |
| Gambar 4.12 Rata-rata akurasi pada evaluasi train-test-split.....        | 93 |
| Gambar 4.13 Rata-rata akurasi pada evaluasi k-fold .....                 | 93 |
| Gambar 4.14 Rata-rata presisi pada evaluasi train-test-split.....        | 94 |
| Gambar 4.15 Rata-rata presisi pada evaluasi k-fold .....                 | 94 |
| Gambar 4.16 Rata-rata recall pada evaluasi train-test-split .....        | 96 |
| Gambar 4.17 Rata-rata recall pada evaluasi k-fold.....                   | 96 |
| Gambar 4.18 Rata-rata f1-Score pada evaluasi train-test-split .....      | 97 |
| Gambar 4.19 Rata-rata f1-Score pada evaluasi k-fold .....                | 97 |

## DAFTAR TABEL

|                                                             |    |
|-------------------------------------------------------------|----|
| Tabel 2.1 Penelitian Terkait .....                          | 14 |
| Tabel 3.1 Deskripsi fitur .....                             | 28 |
| Tabel 3.2 Dataset Employee Attrition.....                   | 30 |
| Tabel 3.3 Dataset Employee Attrition.....                   | 34 |
| Tabel 3.4 Dataset Employee Attrition.....                   | 34 |
| Tabel 3.5 Transformasi encoding.....                        | 37 |
| Tabel 3.6 Transformasi one-hot encoding .....               | 38 |
| Tabel 3.7 Perhitungan korelasi fitur.....                   | 39 |
| Tabel 3.8 Implementasi dataset setelah dinormalisasi .....  | 41 |
| Tabel 3.9 Implementasi perhitungan SS Between Groups.....   | 42 |
| Tabel 3.10 Implementasi perhitungan SS total .....          | 43 |
| Tabel 3.11 Rancangan bootstrap sampling 1 .....             | 49 |
| Tabel 3.12 Tabel kontingensi dari fitur Age .....           | 49 |
| Tabel 3.13 Implementasi perhitungan gini impurity.....      | 51 |
| Tabel 3.14 Confusion matrix .....                           | 53 |
| Tabel 3.15 Implementasi confusion matrix.....               | 54 |
| Tabel 3.16 Skenario model .....                             | 57 |
| Tabel 3.17 Tuning Hyperparameter .....                      | 58 |
| Tabel 4.1 Fitur Setelah Transformasi.....                   | 59 |
| Tabel 4.2 Hasil Normalisasi .....                           | 60 |
| Tabel 4.3 Hasil Korelasi Fitur.....                         | 62 |
| Tabel 4.4 Fitur dengan signifikansi diatas 0,05 .....       | 64 |
| Tabel 4.5 Hasil data sintetik dari SMOTE .....              | 68 |
| Tabel 4.6 Splitting data .....                              | 73 |
| Tabel 4.7 Optimasi parameter .....                          | 74 |
| Tabel 4.8 Matrix evaluasi train-test-split skenario 1.....  | 74 |
| Tabel 4.9 Matrix evaluasi k-fold skenario 1 .....           | 75 |
| Tabel 4.10 Matrix evaluasi train-test-split skenario 2..... | 76 |
| Tabel 4.11 Matrix evaluasi k-fold skenario 2 .....          | 77 |
| Tabel 4.12 Matrix evaluasi train-test-split skenario 3..... | 78 |
| Tabel 4.13 Matrix evaluasi k-fold skenario 3 .....          | 78 |
| Tabel 4.14 Matrix evaluasi train-test-split skenario 4..... | 79 |
| Tabel 4.15 Matrix evaluasi k-fold skenario 4 .....          | 80 |
| Tabel 4.16 Matrix evaluasi train-test-split skenario 5..... | 81 |
| Tabel 4.17 Matrix evaluasi k-fold skenario 5 .....          | 81 |
| Tabel 4.18 Matrix evaluasi train-test-split skenario 6..... | 82 |
| Tabel 4.19 Matrix evaluasi k-fold skenario 6 .....          | 83 |
| Tabel 4.20 Matrix evaluasi train-test-split skenario 7..... | 84 |
| Tabel 4.21 Matrix evaluasi k-fold skenario 7 .....          | 84 |
| Tabel 4.22 Matrix evaluasi train-test-split skenario 8..... | 85 |

|                                                              |    |
|--------------------------------------------------------------|----|
| Tabel 4.23 Matrix evaluasi k-fold skenario 8 .....           | 86 |
| Tabel 4.24 Matrix evaluasi train-test-split skenario 9.....  | 87 |
| Tabel 4.25 Matrix evaluasi k-fold skenario 9 .....           | 87 |
| Tabel 4.26 Matrix evaluasi train-test-split skenario 10..... | 88 |
| Tabel 4.27 Matrix evaluasi k-fold skenario 10 .....          | 89 |
| Tabel 4.28 Matrix evaluasi train-test-split skenario 11..... | 90 |
| Tabel 4.29 Matrix evaluasi k-fold skenario 11 .....          | 90 |
| Tabel 4.30 Matrix evaluasi train-test-split skenario 12..... | 91 |
| Tabel 4.31 Matrix evaluasi k-fold skenario 12 .....          | 92 |

## ABSTRAK

Mulyaningrum, Eka. 2026. **Implementasi Algoritma Random Forest Pada Penurunan Jumlah Karyawan**. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Supriyono, M. Kom (II) Ahmad Fahmi Karami, M.Kom.

**Kata kunci:** *Random Forest, SMOTE, Filtering Selection, Pengunduran Karyawan.*

Penurunan jumlah karyawan menjadi salah satu tantangan utama dalam manajemen proyek yang dapat berdampak pada keberhasilan proyek. Oleh karena itu, diperlukan sistem prediksi yang mampu mengidentifikasi potensi penurunan jumlah karyawan secara akurat dan efisien. Penelitian ini bertujuan untuk membangun model prediksi penurunan jumlah karyawan menggunakan metode *Random Forest* (RF) dengan pendekatan optimasi melalui penyeimbangan data menggunakan *Synthetic Minority Over-sampling Technique* (SMOTE) serta seleksi fitur menggunakan metode filtering, yaitu *Pearson Correlation* dan *Analysis of Variance* (ANOVA). Pengujian dilakukan pada beberapa pengaturan parameter, dan hasil menunjukkan bahwa RF efektif dalam memprediksi penurunan jumlah karyawan, terutama pada konfigurasi hyperparameter yang lebih tinggi seperti pada *max\_depth* 20 dan *n\_estimators* 150. Hasil pengujian terbaik diperoleh pada kombinasi SMOTE dengan seleksi fitur menggunakan *Pearson Correlation* dan ANOVA dengan penyesuaian menghasilkan nilai rata-rata akurasi sebesar 93,07%, presisi 97,095, *recall* 88,8%, dan F1-score 92,75%. Sebaliknya, model tanpa balancing data menunjukkan nilai F1-score yang rendah, mengindikasikan bias terhadap kelas mayoritas. Dengan demikian, kombinasi balancing data dan seleksi fitur terbukti meningkatkan performa model secara signifikan dalam memprediksi penurunan jumlah karyawan, serta dapat mendukung pengambilan keputusan yang lebih proaktif dalam manajemen sumber daya manusia.

## ABSTRACT

Mulyaningrum, Eka. 2026. **Implementation of the Random Forest Algorithm for Employee Attrition**. Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University, Malang. Supervisors: (I) Supriyono, M.Kom. (II) Ahmad Fahmi Karami, M.Kom.

Employee attrition has become one of the main challenges in project management that can significantly affect project success. Therefore, a predictive system is required to accurately and efficiently identify potential employee attrition. This study aims to develop an employee attrition prediction model using the Random Forest (RF) algorithm, enhanced through data balancing using the Synthetic Minority Oversampling Technique (SMOTE) and feature selection using filter-based methods, namely Pearson Correlation and Analysis of Variance (ANOVA). The experiments were conducted under several hyperparameter settings, and the results show that Random Forest is effective in predicting employee attrition, particularly with higher hyperparameter configurations such as max\_depth 20 and n\_estimators 150. The best-performing result was achieved using a combination of SMOTE and feature selection with Pearson Correlation and ANOVA, yielding an average accuracy of 93.07%, precision of 97.09%, recall of 88.8%, and F1-score of 92.75%. In contrast, the model without data balancing showed low F1-score values, indicating a bias toward the majority class. Thus, the combination of data balancing and feature selection was proven to significantly improve model performance in predicting employee attrition and can support more proactive decision-making in human resource management.

**Keywords:** *Random Forest, SMOTE, Filtering Selection, Employees Attrition.*

## البحث مستخلص

موليانينغروم، إيك. 2026. تطبيق خوارزمية Random Forest على معدل ترك الموظفين. أطروحة. برنامج دراسة هندسة المعلوماتية، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية، مالانغ. المشرفان: (1) سوبريونو ماجستير في علوم الحاسوب. (2) أحمد فهمي كرامي، ماجستير في علوم الحاسوب.

الكلمة المفتاحية : تناقص عدد الموظفين، Random Forest, SMOTE, Filtering Selection

أصبح معدل ترك الموظفين أحد التحديات الرئيسية في إدارة المشاريع، وهو ما يمكن أن يؤثر بشكل كبير على نجاح المشروع. لذلك، هناك حاجة إلى نظام تنبؤي لتحديد احتمالية استنزاف الموظفين بدقة وكفاءة. تهدف هذه الدراسة إلى تطوير نموذج للتنبؤ باستنزاف الموظفين باستخدام خوارزمية مع تعزيزها من خلال موازنة البيانات باستخدام تقنية Random Forest (RF). مع تعزيزها من خلال موازنة البيانات باستخدام تقنية Synthetic Minority Oversampling Technique (SMOTE) واختيار السمات باستخدام طرق قائمة على المرشحات، وهي Pearson Correlation و ANOVA Analysis Of Variance. أُجريت التجارب في ظل عدة إعدادات للمعلمات الفائقة، وأظهرت النتائج أن خوارزمية Random Forest فعالة في التنبؤ بتسرب الموظفين، لا سيما مع تكوينات المعلمات الفائقة الأعلى مثل max\_depth 20 و n\_estimator 150. وقد تم تحقيق أفضل نتيجة من حيث الأداء باستخدام مزيج من تقنية SMOTE واختيار السمات باستخدام Pearson Correlation و ANOVA، مما أدى إلى دقة متوسطة بلغت 93,07%، ودقة تحديد بلغت 97,09%، ومعدل استرجاع بلغ 88,8%، و F1-Score بلغ 92,75%. في المقابل، أظهر النموذج الذي لم يخضع لموازنة البيانات قيمة منخفضة لـ F1-Score، مما يشير إلى وجود انحياز نحو الفئة الأغلبية. وبالتالي، ثبت أن الجمع بين موازنة البيانات واختيار السمات يحسن أداء النموذج بشكل ملحوظ في التنبؤ بتسرب الموظفين، ويمكن أن يدعم اتخاذ قرارات أكثر استباقية في إدارة الموارد البشرية.

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Manajemen proyek merupakan strategi penting untuk mencapai keberhasilan organisasi, termasuk perusahaan. Salah satu aspek krusial dalam manajemen proyek adalah pengelolaan sumber daya manusia, khususnya karyawan yang menjadi pelaku utama dalam menjalankan seluruh kegiatan operasional dan mendukung implementasi kebijakan (Kitole & Utouh, 2024). Namun, karyawan juga rentan terhadap berbagai faktor yang dapat memengaruhi kinerja dan kelangsungan keberadaannya dalam proyek (Jia, 2024).

Saat ini, salah satu tantangan besar pada manajemen proyek adalah penurunan jumlah karyawan, yang dapat mengganggu kelancaran operasional dan berdampak pada keberhasilan proyek (Kitole & Utouh, 2024). Penurunan jumlah karyawan tersebut umumnya disebabkan oleh berbagai faktor, antara lain kurangnya penghargaan dari perusahaan, pengembangan karir, ketidakpuasan terhadap gaji, beban kerja, lingkungan kerja yang kurang dinamis, dan beberapa kebijakan dari perusahaan itu sendiri hingga kebijakan dari pemerintahan, serta tingkat *turnover* yang tinggi (Usha & Balaji, 2021). *Turnover* karyawan pada konteks ini merujuk pada pengisian kembali posisi yang ditinggalkan oleh karyawan atau bisa disebut juga pergantian karyawan (Churchill et al., 2023; Mwakiluma, 2024). Penyebab-penyebab tersebut juga dapat mengakibatkan penurunan kualitas kerja hingga *attrition*, dimana *attrition* adalah tenaga kerja yang keluar secara sukarela maupun tidak dan belum ada rencana pergantian oleh

perusahaan. *Attrition* yang tinggi menyebabkan banyak kerugian hingga kegagalan proyek (Raza et al., 2022). Berdasarkan laporan Project Management Institute, kehilangan beberapa tenaga kerja apalagi yang memiliki keahlian khusus dapat menyebabkan keterlambatan proyek hingga 15-20% dari jadwal (Venema & Gueniche, 2014).

Menurut Survei Angkatan Kerja Nasional, antara bulan Februari 2020 hingga bulan Agustus 2021 yang dilakukan oleh Badan Pusat Statistik (BPS) di Indonesia, tenaga kerja yang berhenti bekerja menunjukkan peningkatan, dari nilai 4.99% menjadi 6,49% (Direktorat Statistik Kependudukan dan Ketenagakerjaan, 2023; Subdirektorat Statistik Ketenagakerjaan, 2019). Peningkatan ini juga dipengaruhi oleh pandemi *covid-19* yang membatasi aktivitas sosial hingga ekonomi. Selain itu, kebijakan efisiensi yang mengharuskan beberapa perusahaan memutuskan mengurangi karyawannya untuk menekan biaya belanja dan mencegah kerugian (Alfaridzy et al., 2025; Budiman et al., 2025). Kementerian Ketenagakerjaan (Kemnaker) mencatat, hingga bulan April 2025 telah terjadi pemutusan hubungan kerja (PHK) sebanyak 24.036 pekerja, yang sebagian besar terjadi di bulan Februari hingga 17.796 pekerja. Angka ini menunjukkan sepertiga dari total PHK tahun lalu yang berjumlah 77.965 pekerja (Kementrian Tenaga Kerja Republik Indonesia, 2025).

Sehingga, secara tidak langsung *attrition* yang tinggi juga dapat menyebabkan *turnover* secara struktural. Dimana terjadi pergantian tenaga kerja besar-besaran akibat penurunan jumlah karyawan yang terlibat. Di Amerika mencatat usia 16-19 memiliki masa kerja dengan rata-rata kurang dari 12 bulan,

dan setiap tahun hampir terjadi pergantian karyawan hingga 50% (Bureau Labor Statistics U. S. Department Of Labor, 2024). Pergantian tersebut juga mendesak beberapa perusahaan melakukan perekrutan karyawan baru. Sedangkan perkiraan biaya penggantian seorang karyawan dapat mencapai 1 kali gaji tahunan penuh karyawan tersebut, termasuk biaya perekrutan, pelatihan, dan kehilangan produktivitas (Buzeti et al., 2016).

Dengan memprediksi penurunan jumlah karyawan dapat menjadi langkah penting dalam upaya manajemen proyek, sehingga dapat mengidentifikasi karyawan yang berpotensi meninggalkan perusahaan dengan karyawan yang tetap tinggal di perusahaan. Dengan memprediksi ancaman-ancaman yang ada dapat membantu perusahaan melakukan pencegahan sedini mungkin, serta mitigasi resiko tersebut secara proaktif. Namun, proses prediksi yang dilakukan secara manual akan memakan waktu lama, rentan terhadap kesalahan subjektif, dan kurang efisien ketika jumlah karyawan yang bisa mencapai ratusan atau ribuan, dengan banyak informasi yang perlu dianalisis. Oleh karena itu, diperlukan sistem prediksi yang mampu memprediksi penurunan karyawan ini.

Perencanaan yang baik adalah upaya dari manajemen proyek, terutama dalam pengelolaan sumber daya manusia. Hal ini, dijelaskan pula dalam Al-Quran, dimana pentingnya perencanaan, seperti dijelaskan pada surat Al Hasyr ayat 18 :

يَا أَيُّهَا الَّذِينَ آمَنُوا اتَّقُوا اللَّهَ وَلْتَنْظُرْ نَفْسٌ مَّا قَدَّمَتْ لِغَدٍ وَاتَّقُوا اللَّهَ إِنَّ اللَّهَ خَبِيرٌ بِمَا تَعْمَلُونَ ﴿١٨﴾

*“Hai orang-orang yang beriman, bertakwalah kepada Allah dan hendaklah setiap diri memperhatikan apa yang telah diperbuatnya untuk hari esok (akhirat); dan bertakwalah kepada Allah, sesungguhnya Allah Maha Mengetahui apa yang kamu kerjakan.” (Q.S. Al Hasyr ayat 18).*

Dari ayat tersebut dijelaskan bahwa perencanaan yang baik harus mempertimbangkan kondisi saat ini dan masa depan. Perencanaan penting dalam manajemen, karena manusia bisa menentukan masa depannya lewat usaha dan ikhtiar. Manusia tidak boleh pasrah pada keadaan, melainkan harus terus berusaha. Masa depan adalah hasil dari masa lalu, kondisi sekarang, dan usaha yang sedang dilakukan. Oleh karena itu, dasar perencanaan adalah kemampuan manusia memilih arah masa depan, dan mengupayakannya secara sadar, termasuk menentukan manajemen yang tepat agar rencana bisa terlaksana dengan baik.

Dalam hal ini, algoritma *Random Forest* (RF) dipilih sebagai metode untuk perencanaan, yakni memprediksi penurunan jumlah karyawan pada manajemen proyek. RF merupakan algoritma *ensemble learning* yang sangat efektif dalam menangani masalah prediksi dengan data yang kompleks dan besar, karena dibangun dengan banyak pohon keputusan, sehingga lebih tahan terhadap data yang *noise* dan data yang tidak bersih daripada pohon keputusan tunggal atau *Decision Tree* (DT). Serta cenderung mengurangi resiko *overfitting*, sehingga dapat memberikan model yang lebih *generalizable* dengan akurasi yang lebih baik. Komputasi dalam model RF ini juga cenderung lebih efisien jika dibandingkan dengan model pembobotan seperti *Support Vector Machine* (SVM) atau *Neural Networks*. Pada penelitian yang dilakukan oleh (Cherian et al., 2025) menunjukkan performa model *Random Forest* lebih unggul dengan akurasi 87.07% dibandingkan dengan model *Tree-Based* lainnya, seperti *Hoeffding Tree*, *Rep Tree*, J48, dan *Logistic Model Tree* (LMT).

Namun, meskipun *Random Forest* dapat meningkatkan akurasi prediksi, tantangan lain dalam penggunaannya adalah masalah keseimbangan kelas dalam dataset, yang dapat menyebabkan model cenderung mengabaikan kelas minoritas, seperti karyawan yang berpotensi keluar. Oleh karena itu, penggunaan teknik seperti *Synthetic Minority Over-sampling Technique* (SMOTE) dapat membantu mengatasi masalah ini dengan menyeimbangkan dataset agar model lebih sensitif terhadap kelas minoritas (Lugoma et al., 2025).

Tidak hanya itu, dalam membangun model prediksi, pemilihan fitur juga menjadi tantangan (Hamdard & Lodin, 2023). Oleh karena itu, penggunaan seleksi fitur dengan metode *filtering* seperti korelasi fitur, dan *Analysis of Variance* (ANOVA) dapat membantu dalam menentukan fitur-fitur yang paling signifikan dalam memprediksi penurunan jumlah karyawan. Dengan menggunakan teknik ini, model dapat lebih tepat dalam menangkap variabel yang berpengaruh terhadap keputusan karyawan untuk bertahan atau keluar dari perusahaan.

Penelitian ini bertujuan untuk memanfaatkan algoritma *Random Forest* yang dikombinasikan dengan teknik SMOTE, ANOVA, dan analisis korelasi fitur untuk mengoptimalkan prediksi penurunan jumlah karyawan dalam manajemen proyek. Melalui pendekatan ini, diharapkan dapat dicapai prediksi yang lebih akurat dan efisien.

## 1.2 Pernyataan Masalah

Berdasarkan latar belakang diatas, maka rumusan masalah dalam penelitian ini adalah:

1. Bagaimana penerapan Metode *Random Forest* (RF) dapat digunakan secara efektif dalam memprediksi penurunan jumlah karyawan ?
2. Bagaimana pengaruh seleksi fitur dan penyeimbangan data pada metode *Random Forest* dalam memprediksi penurunan jumlah karyawan?
3. Sejauh mana kemampuan *Random Forest* dapat memprediksi dengan baik, berdasarkan nilai akurasi, presisi, *recall*, dan *F1-Score*, serta apa saja pola-pola analisis yang dapat dikembangkan ?

## 1.3 Batasan Masalah

Batasan masalah yang ditetapkan dalam penelitian ini, diantaranya:

1. Penelitian ini terbatas pada dataset publik yang tersedia secara terbuka pada platform *Kaggle* dan terakhir di perbarui sekitar tahun 2020 (Sahoo, 2020).
2. Dalam menangani penyeimbangan data penelitian ini menggunakan teknik SMOTE, untuk mengenali lebih baik kelas minoritas. Serta dalam seleksi fitur, penelitian ini menggunakan metode *filtering*, seperti teknik *pearson correlation* untuk mengetahui hubungan antar masing masing fitur, dan ANOVA untuk mengetahui hubungan fitur dengan 2 kelas dari fitur target yang ada.
3. *Hyperparameter* yang dioptimalkan adalah “*n\_estimators*”, dan “*max\_depth*”.

#### 1.4 Tujuan Penelitian

1. Membangun model prediksi penurunan jumlah karyawan dengan metode *Random Forest* (RF).
2. Mengukur pengaruh analisis korelasi fitur, seleksi fitur dan penyeimbangan data pada metode *Random Forest* (RF) dalam memprediksi penurunan jumlah karyawan.
3. Mengukur seberapa baik performa metode *Random Forest* (RF) dalam memprediksi penurunan jumlah karyawan.

#### 1.5 Manfaat Penelitian

Pada penelitian ini diharapkan mampu memberikan manfaat dengan melibatkan penerapan metode *Random Forest* (RF) untuk memprediksi pola penurunan jumlah karyawan berdasarkan banyaknya jumlah karyawan yang keluar dalam kurun waktu tertentu, sehingga diharapkan juga dapat meningkatkan kemampuan perusahaan dalam mengidentifikasi risiko dari penurunan jumlah karyawan secara lebih akurat, dan memungkinkan pengambilan keputusan yang lebih proaktif dalam mengelola sumber daya manusia dalam konteks manajemen proyek. Dengan demikian, perusahaan dapat mengantisipasi potensi kerugian akibat kekosongan posisi. Serta kontribusi terhadap literatur manajemen proyek dan analisis data dimana kontenks ini adalah prediksi menggunakan *machine learning* (ML).

## **BAB II**

### **STUDI PUSTAKA**

#### **2.1 Penurunan Jumlah Karyawan**

Konsep utama manajemen proyek adalah untuk mencapai hasil yang diinginkan dengan memperhatikan semua aspek yang terkait, mulai dari perencanaan, pelaksanaan, hingga evaluasi (Shabdar, 2009). Pengelolaan sumber daya manusia (SDM) sangat penting dalam manajemen proyek, tidak hanya bertanggung jawab untuk menjalankan tugas dan mencapai target, tetapi mereka juga harus memiliki keterampilan dan motivasi yang tinggi untuk memastikan proyek berjalan dengan lancar. Akibatnya, keberhasilan proyek sangat bergantung pada bagaimana SDM atau karyawan dikelola dan dijaga dalam kondisi terbaik (Sirshar et al., 2019).

Penurunan jumlah karyawan adalah salah satu tantangan dalam pengelolaan manajemen proyek. Banyak faktor penyebabnya, baik internal maupun eksternal yang memengaruhi kepuasan karyawan dan loyalitas mereka terhadap perusahaan. Faktor internal seperti kebijakan perusahaan yang tidak mendukung kesejahteraan karyawan, gaji yang tidak memadai, atau kurangnya peluang pengembangan karir. Data dari Jobstreet tahun 2025 menunjukkan bahwa 38% perusahaan mengurangi jumlah pegawai, dengan alasan restrukturisasi organisasi, otomatisasi, dan sebagian besar juga dipicu alasan internal perusahaan (Hutahaean, 2025).

Kurangnya pelatihan dan rendahnya kesadaran terhadap risiko kerja, juga menjadi faktor internal yang berkontribusi terhadap penurunan jumlah karyawan. Kondisi ini tidak hanya menghambat efektivitas penerapan sistem manajemen

proyek, terutama dalam keselamatan dan kesehatan kerja, tetapi juga berpotensi meningkatkan tingkat kesalahan kerja, kecelakaan, serta menurunkan kepuasan dan loyalitas karyawan (Candrawati, 2025).

Sementara itu, faktor eksternal seperti ekonomi yang tidak stabil atau krisis global juga dapat menyebabkan penurunan jumlah karyawan. Data dari Jobstreet ditahun yang sama 77% perusahaan mengurangi jumlah pegawai dengan alasan mengurangi biaya operasional dan 48% karena perkiraan kondisi ekonomi yang kurang baik (Hutahaean, 2025). Oleh karena itu, perusahaan harus memiliki strategi pengelolaan SDM yang kuat untuk mengurangi dampak penurunan jumlah karyawan pada manajemen proyek.

## 2.2 Penelitian Terkait

Dalam penelitian yang dilakukan oleh (Tang et al., 2025) dengan judul *“Enhancing Employee Retention: Predicting Attrition Using Machine Learning Models”*, penelitian ini menggunakan 10 metode, diantaranya *Bagging Classifier*, *Decision Tree Classifier*, *Extra Tree Classifier*, *K-Neighbors Classifier*, *Linear Discriminant Analysis*, *Logistic Regression*, *Random Forest Classifier*, *Ridge Classifier*, *Stochastic Gradient Descent Classifier*, dan *Support Vector Classification* untuk memprediksi *attrition* karyawan. Dataset yang digunakan adalah *IBM Employee Attrition* yang terdiri dari 1.470 data dengan 31 fitur. Untuk mengatasi ketidakseimbangan data, penelitian ini menerapkan teknik SMOTE, serta evaluasi menggunakan metode 10-fold *cross-validation*. Berdasarkan hasil penelitian, *Random Forest* menunjukkan performa terbaik dengan akurasi tertinggi sebesar 94,4% setelah penerapan SMOTE. Selain itu, analisis *Receiver Operating*

*Characteristic* (ROC) juga menunjukkan bahwa *Random Forest* memiliki performa yang lebih baik yakni 0,99 dibandingkan *Logistic Regression* 0,86.

Dalam penelitian yang dilakukan oleh (Pandey et al., 2024) dengan judul “*Attrition Analytics: Unveiling the Best Model for Predicting Employee Retention*”, penelitian ini menggunakan beberapa algoritma machine learning yaitu *Decision Tree*, *Random Forest*, *Logistic Regression*, dan *Naïve Bayes* untuk memprediksi *attrition* karyawan. Dataset yang digunakan berasal dari situs publik *Kaggle*. Dalam penelitian ini juga dilakukan optimasi model menggunakan teknik *hyperparameter tuning* seperti *GridSearchCV* untuk meningkatkan performa model. Evaluasi dilakukan menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, serta *Area Under the Curve - Receiver Operating Characteristic* (AUC-ROC). Berdasarkan hasil yang diperoleh, *Logistic Regression* menunjukkan performa terbaik dengan akurasi sekitar 88,41% pada data *training* dan 86,59% pada data *testing*. Sementara itu, *Random Forest* dan *Logistic Regression* memiliki nilai AUC tertinggi sebesar 93% yang menunjukkan kemampuan klasifikasi yang sangat baik, sedangkan *Decision Tree* dan *Naïve Bayes* memiliki AUC sebesar 84%.

Dalam penelitian yang dilakukan (Karimi & Viliyani, 2024) dengan judul “*Employee Turnover Analysis Using Machine Learning Algorithms*” ini, menggunakan tiga algoritma *machine learning* berbasis *ensemble* yakni *AdaBoost*, *Support Vector Machine* (SVM), dan *Random Forest* untuk mengukur akurasi prediksi pergantian karyawan. Penelitian tersebut menggunakan dataset IBM yang terdiri dari 30 fitur, dari 35 fitur yang ada. Berdasarkan hasil yang diperoleh,

analisis *Receiver Operating Characteristic* (ROC) mengungkapkan bahwa *AdaBoost* memiliki area di bawah kurva *Area Under the Curve* (AUC) tertinggi, yang menandakan kemampuan yang lebih baik dalam membedakan dan mengklasifikasikan instansi dengan tingkat *true positive rate* (TPR) sebesar 84%. Sementara itu, SVM dan *Random Forest* memiliki AUC masing-masing sebesar 83% dan 80%.

Pada penelitian (Al-Alawi & Albuainain, 2024) berfokus pada analisis klasifikasi pada siklus hidup karyawan dari perekrutan hingga pension, dengan judul “*Machine Learning in Human Resource Analytics: Promotion Classification using Data Balancing Techniques*”. Penelitian ini menggunakan dataset publik dari situs *Kaggle* dengan total 14 fitur. Penelitian tersebut menerapkan model *machine learning* *Decission Tree*, *Extra Trees*, *Random Forest*, dan *Extreme Gradient Boosting* (XGBoost). Serta penerapan *balancing* data dengan teknik SMOTE, penerapan normalisasi data, serta seleksi fitur dengan teknik *Principal Component Analysis* (PCA). Dan hasilnya menunjukkan bahwa penggunaan *Synthetic Minority Over Sampling Technique* (SMOTE) dan *Random Forest Classifier* (RFC) menghasilkan tingkat akurasi 99%.

Penelitian yang dilakukan oleh (Kaya & Şimşek, 2023) berfokus pada penerapan penanganan ketidakseimbangan dataset dengan judul “*Prediction of Employee Turnover with Imbalance Dataset Using Machine Learning Methods*”. Penelitian tersebut menggunakan dataset yang berasal dari website IBM dengan total fitur yang digunakan berjumlah 1470 data dan 35 fitur, yang terdiri dari 29 fitur numerik dan 9 fitur kategorikal. Penelitian tersebut mengimplementasikan

beberapa model *machine learning* diantaranya, *Random Forest*, *Support Vector Machine* (SVM), *Decision Tree*, *Logistic Regression*, *Naïve Bayes*, dan *K-Nearest Neighbour* (KNN) dengan membandingkan teknik penanganan ketidakseimbangan data yakni, *Random Oversampling Samples* (ROS), *Synthetic Minority Oversampling Technique* (SMOTE), *Random Subsampling* (RUS), dan *Near Miss*. Dalam penelitian tersebut juga menerapkan evaluasi model dengan metode *K-Fold*, serta *GridSearchCV* dalam pencarian parameter terbaik. Model *Random Forest* menunjukkan performa tertinggi berdasarkan nilai *F1-Score* senilai 92%, ditambah dengan penerapan *balancing* data maka ditemukan parameter terbaik menggunakan teknik ROS dengan nilai *F1-Score* 96%.

Pada penelitian yang dilakukan oleh (Sharma & Sharma, 2023) yang berjudul “*Analyzing Employee's Attrition and Turnover at Organization Using Machine learning Technique*” berfokus pada analisis pergantian karyawan menggunakan teknik pembelajaran *machine learning*, khususnya *Random Forest* dan *Logistic Regression*. Dalam penelitian tersebut menggunakan dataset IBM yang terdiri dari 35 fitur dan 1471 baris data. Hasil penelitian tersebut menunjukkan bahwa *Random Forest* memiliki performa yang lebih baik dari pada *Logistic Regression*, dengan masing-masing akurasi 87% dan 85%.

Hasil penelitian oleh (Silpa et al., 2023) yang menerapkan teknik seleksi fitur pada *machine learning* untuk analisis retensi karyawan dengan menggunakan dataset IBM dengan 28 fitur. Judul penelitian tersebut adalah “*An Enriched Employee Retention Analysis System with a Combination Strategy of Feature Selection and Machine Learning Techniques*”, dengan mengimplementasikan

beberapa model yakni, *XGBoost*, *Linear Regression*, *Decision Tree*, *Support Vector Machine*, dan *Naive Bayes*. Sedangkan teknik analisis dan seleksi fitur masing-masing menggunakan *Exploratory Data Analyst* (EDA), *Minimum Redundancy Maximum Relevance* (MRMR), *Chi-2*, *Analysis of Variance* (ANOVA), dan *Kruskal*. Hasilnya menunjukkan bahwa pemilihan fitur meningkatkan akurasi klasifikasi sekaligus mengurangi waktu pelatihan.

Dalam penelitian yang dilakukan oleh (Usha & Balaji, 2021) dengan judul “*A Comparative Study on Machine Learning Algorithms for Employee Attrition Prediction*”, penelitian ini membandingkan beberapa algoritma machine learning yaitu *Naive Bayes*, *Decision Table*, *J48*, *Random Forest*, dan *K-Means* dalam memprediksi attrition karyawan. Dataset yang digunakan merupakan data real-time dari karyawan sektor IT yang dikumpulkan melalui kuesioner dengan berbagai variabel demografis dan tingkat kepuasan kerja yang kurang lebih sejumlah 25 fitur. Evaluasi model dilakukan menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, serta waktu komputasi. Hasil penelitian menunjukkan bahwa *Naive Bayes* memiliki akurasi tertinggi sebesar 85,98% serta waktu komputasi tercepat. Sedangkan akurasi yang dimiliki *J48* dan *Random Forest* senilai 85.05%. Namun, algoritma *J48* dan *Random Forest* menunjukkan performa yang lebih baik dalam hal *recall* (94,03%) dan *F1-score* (88,73%), yang mengindikasikan kemampuan yang lebih baik dalam menangani keseimbangan data, dibandingkan dengan *recall* dan *F1-score* *Naive Bayes* yang senilai 86,57% dan 88,5%. Selain itu, analisis korelasi menunjukkan bahwa faktor seperti kebanggaan terhadap perusahaan, *job security*, *promotion policy*, *work-life balance*, *recognition*, dan *growth opportunity*

berpengaruh signifikan terhadap keputusan karyawan untuk bertahan di perusahaan.

Tabel 2.1 Penelitian Terkait

| No. | Peneliti                      | Input                                                         | Metode                                                                                                                                                                     | Perbedaan                                                                                                                                                                                                                                                                                                                                 |
|-----|-------------------------------|---------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.  | (Tang et al., 2025)           | Dataset IBM, yang terdiri dari 31 fitur                       | 1). Model ML : 10 model <i>machine learning</i> .<br>2). Model <i>balancing</i> : SMOTE.<br>3). Teknik evaluasi model : <i>K-Fold</i> dengan jumlah <i>fold</i> 10         | Peneliti menggunakan dataset publik milik IBM dengan 35 fitur , hanya menerapkan metode <i>Random Forest</i> , dan menerapkan metode pada <i>praprocessing</i> yakni seleksi fitur menggunakan ANOVA dan <i>person correlation</i> . Pada evaluasi pengujian peneliti hanya menggunakan teknik <i>k-fold</i> dengan jumlah <i>fold</i> 5. |
| 2.  | (Pandey et al., 2024)         | Employee dataset publik dari situs <i>Kaggle</i>              | 1). Model ML : <i>Random Forest</i> , <i>Decision Tree</i> , <i>Logistic Regression</i> , dan <i>Naïve Bayes</i> .<br>3). Teknik evaluasi model : <i>GridSearchCV</i>      | Peneliti menggunakan dataset publik milik IBM dengan 35 fitur , hanya menerapkan metode <i>Random Forest</i> , dan menerapkan metode pada <i>praprocessing</i> yakni <i>balancing</i> data menggunakan SMOTE dan seleksi fitur menggunakan ANOVA dan <i>person correlation</i> .                                                          |
| 3.  | (Karimi & Viliyani, 2024)     | Dataset IBM, yang terdiri dari 30 fitur                       | <i>Adaboost</i> , <i>SVM</i> , <i>Random Forest</i>                                                                                                                        | Peneliti menggunakan seluruh fitur pada dataset yakni 35 fitur , hanya menerapkan metode <i>Random Forest</i> , dan menerapkan metode pada <i>praprocessing</i> yakni <i>balancing</i> data menggunakan SMOTE dan seleksi fitur menggunakan ANOVA dan <i>person correlation</i> .                                                         |
| 4.  | (Al-Alawi & Albuainain, 2024) | Dataset publik dari situs <i>Kaggle</i> dengan total 14 fitur | 1). Model ML : <i>Decission Tree</i> , <i>Extra Trees</i> , <i>Random Forest</i> , dan <i>XGBoost</i><br>2). Model <i>praprocessing</i> : SMOTE, PCA, dan normalisasi data | Dataset yang digunakan berbeda dengan peneliti gunakan, peneliti menggunakan dataset dari IBM dengan total 35 fitur 1470 data, dan hanya menerapkan metode <i>Random Forest</i> dan SMOTE, serta menerapkan analisis dan seleksi fitur pada <i>praprocessing</i> menggunakan teknik ANOVA dan <i>person correlation</i> .                 |
| 5.  | (Kaya & ŞİMŞEK, 2023)         | Dataset IBM yang terdiri dari 35 fitur dan 1470 baris data    | 1). Model ML : <i>Random Forest</i> , <i>SVM</i> , <i>Decision Tree</i> , <i>Logistic Regression</i> , <i>Naïve Bayes</i> , dan (KNN)                                      | Peneliti hanya menerapkan metode <i>Random Forest</i> dan SMOTE , dan menerapkan metode pada <i>praprocessing</i> untuk seleksi fitur menggunakan ANOVA dan <i>person correlation</i> .                                                                                                                                                   |

| No. | Peneliti                | Input                                                                             | Metode                                                                                                                                                                                                                                         | Perbedaan                                                                                                                                                                                                                                                                      |
|-----|-------------------------|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|     |                         |                                                                                   | 2). Model <i>balancing</i> : ROS, SMOTE, RUS, dan <i>Near Miss</i> .<br><br>3). Teknik evaluasi model : <i>K-Fold</i> , dan <i>GridSearchCV</i>                                                                                                |                                                                                                                                                                                                                                                                                |
| 6.  | (Sharma & Sharma, 2023) | Dataset IBM yang terdiri dari 35 fitur dan 1471 baris data                        | <i>Random Forest</i> , <i>Logistic Regression</i>                                                                                                                                                                                              | Peneliti hanya menerapkan metode <i>Random Forest</i> , menerapkan metode pada <i>praprocessing</i> untuk <i>balancing</i> data menggunakan SMOTE dan seleksi fitur menggunakan ANOVA dan <i>person correlation</i> .                                                          |
| 7.  | (Silpa et al., 2023)    | Dataset IBM, yang terdiri dari 28                                                 | 1). Model ML : <i>Decision Tree</i> , <i>Linear Regression</i> , <i>Naive Bayes</i> , <i>Support Vector Machine</i> , dan <i>XGBoost</i><br><br>2). Model seleksi fitur : MRMR, Chi-2, ANOVA, dan Kruskal<br><br>3). Model analisis data : EDA | Peneliti menggunakan seluruh fitur pada dataset yakni 35 fitur, menerapkan metode <i>Random Forest</i> , dan hanya menerapkan teknik seleksi fitur menggunakan ANOVA dan <i>person correlation</i> .                                                                           |
| 8   | (Usha & Balaji, 2021)   | Berdasarkan kuisioner di bidang IT sektor yang terdiri dari kurang lebih 25 fitur | 1) Model ML : <i>Naïve Bayes</i> , <i>Decision table</i> , <i>Random Forest</i> , J48 dan <i>K-Means</i><br><br>2) Model Analisis data : <i>Fitur correlation</i>                                                                              | Peneliti menggunakan dataset publik milik IBM dengan 35 fitur, dan hanya menerapkan algoritma <i>Random Forest</i> , SMOTE , serta menerapkan metode pada <i>praprocessing</i> untuk seleksi fitur menggunakan ANOVA. Tidak menguji kecepatan komputasi pada setiap pengujian. |

Tabel 2.1 menyajikan ringkasan perbedaan penelitian ini dengan penelitian sebelumnya yang menggunakan berbagai metode dalam prediksi , terutama terkait prediksi karyawan, selain itu juga menyajikan beberapa informasi terkait penelitian sebelumnya yang menerapkan model terutama yang mengaplikasikan algoritma *Random Forest*, penerapan *balancing* data menggunakan SMOTE, serta seleksi

fitur menggunakan metode *filtering* dalam memprediksi dan menganalisis pola-pola karyawan. Tabel ini mencakup nama peneliti, dataset, fitur-fitur, dan metode yang digunakan, serta perbedaannya dengan penelitian ini.

### 2.3 Seleksi Fitur dengan Metode *Filtering*

*Filter Methods* adalah jenis seleksi fitur yang berfokus pada analisis statistik. Metode seleksi jenis ini biasanya mengukur fitur terpenting berdasarkan skor suatu atribut serta hubungannya dengan atribut target (Cheng, 2024). Contoh seleksi fitur dengan metode *filtering* adalah *Analysis of Variance* (ANOVA), *Chi-Square*, *Pearson Correlation*, *Univariate Selection Thecnique*, dan lain sebagainya (Cheng, 2024). Penelitian ini berfokus pada seleksi fitur dengan menggunakan metode *Pearson Correlation* untuk mengukur hubungan antar fitur dan ANOVA untuk mengukur hubungan fitur bebas dengan fitur target.

#### 2.3.1 *Pearson Correlation*

*Pearson Correlation* dikembangkan oleh Karl Pearson sekitar akhir abad ke-19, yang berasal dari konsep yang dicetuskan Francis Galton pada tahun 1880-an, mengenai hubungan dalam studi biologi, khususnya pewarisan sifat (Stanton, 2001). *Pearson correlation* atau *Pearson Product-Moment Correlation* adalah analisis hubungan antara masing-masing setiap fitur, yang umumnya digunakan untuk mengukur hubungan linear, yakni dengan mengukur kekuatan dan arah hubungan dari dua variabel dengan menggunakan rumus koefisien korelasi *pearson* ( $r$ ) (Wilson, 2018). Interpretasi nilai pada korelasi:

1. Nilai  $r = 0.00-0.19$ : Sangat lemah

2. Nilai  $r = 0.20-0.39$ : Lemah
3. Nilai  $r = 0.40-0.59$ : Sedang
4. Nilai  $r = 0.60-0.79$ : Kuat
5. Nilai  $r = 0.80-1.00$ : Sangat kuat

Nilai  $r$  semakin mendekati 1 maka ada hubungan yang kuat antara dua variabel yang diuji dengan nilai positif sempurna (Schober et al., 2018). Sehingga fitur-fitur yang memiliki hubungan sangat kuat bahkan ekstrim sebaiknya dihapus, karena dapat menyebabkan *noise* yang tidak perlu dan mempengaruhi akurasi prediksi (Crawford, 2006). Perhitungan korelasi atau *Pearson Product-Moment Correlation* (Lee Rodgers & Nicewander, 1988), dirumuskan sebagai berikut:

$$r = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}} \quad (2.1)$$

Keterangan:

- $\sum$  : Simbol yang menunjukkan total.
- $X_i$  : Nilai asli dari fitur  $X$ .
- $Y_i$  : Nilai asli dari fitur  $Y$ .
- $\bar{X}$  dan  $\bar{Y}$  : Nilai rata-rata dari fitur  $X$  dan fitur  $Y$ .

### 2.3.2 Analysis of Variance (ANOVA)

Analysis of Variance (ANOVA) dikembangkan oleh ahli statistik Inggris Ronald A. Fisher pada tahun 1920-an (Kaufmann & Schering, 2014). ANOVA merupakan teknik pengujian statistik untuk mengukur signifikansi setiap fitur terhadap fitur target berdasarkan perbedaan rata-rata antar kelompok (Shaker et al., 2025). Kelompok atau kelas yang dimaksud adalah label pada fitur target yakni fitur *Attrition*. Pada fitur *Attrition* terdapat dua label, label pertama dengan label ‘0’ yang diprediksi sebagai ‘Tidak Attrition’, sedangkan label kedua dengan label ‘1’ yang diprediksi sebagai ‘Ya Attrition’. Dalam *machine learning* ANOVA biasanya

diimplementasikan dalam bentuk pengujian ANOVA F-test yang berbasis ANOVA *One-Way* (Annisa et al., 2025). Oleh karena itu *One-Way* ANOVA dipilih untuk mengukur perbedaan rata-rata dari masing-masing kelas tersebut dengan variabel lainnya. Selanjutnya pemilihan fitur-fitur terpenting berdasarkan nilai F dan nilai *p\_value*. Hasil nilai *p\_value* akan dibandingkan dengan tingkat signifikansi, jika nilai *p\_value* dibawah 0,05 maka akan dipertahankan, sementara jika *p\_value* memiliki nilai diatas 0,05 akan dihilangkan. Langkah-langkah perhitungan ANOVA untuk mengukur nilai F (Pace, 2012):

1. Menghitung *Sum of Square Between Groups*, atau menghitung perbedaan rata-rata grup atau kelas pada fitur target. Berikut rumusnya:

$$SS_{between} = \sum_{j=1}^k n_j (\bar{X}_j - \bar{X})^2 \quad (2.2)$$

Keterangan:

- $k$  : Kelompok atau jumlah grub.
- $n_j$  : Jumlah data dalam grub.
- $\bar{X}_j$  : Rata-rata setiap kelompok atau grup.
- $\bar{X}$  : Rata-rata total seluruh data.

2. Menghitung total *Sum of Squares Within Groups*, atau menghitung variansi total dari masing-masing grup atau kelas pada fitur target untuk mengukur nilai perbedaan dalam setiap kelas. Berikut rumusnya:

$$SS_{within} = \sum_{j=1}^k \sum_{i=1}^{n_j} (X_{ij} - \bar{X}_j)^2 \quad (2.3)$$

Keterangan:

- $k$  : Kelompok atau jumlah grub.
- $n_j$  : Jumlah data dalam grub.
- $X_{ij}$  : Nilai data berdasarkan kelompok atau grup.
- $\bar{X}_j$  : Rata-rata total seluruh data dalam kelompok.

3. Menghitung *Mean Squares Between* dan *Mean Square Within*, atau menghitung rata rata *Sum of Square* dari *Between* dan *Within* dengan jumlah data, baik jumlah data pada tiap-tiap kelas ataupun pada jumlah keseluruhan data. Berikut rumusnya:

$$MS_{between} = \frac{SS_{between}}{k - 1} \quad (2.4)$$

$$MS_{within} = \frac{SS_{within}}{N - k} \quad (2.5)$$

Keterangan :

- a)  $k$  : Kelompok atau jumlah grub.
- b)  $N$  : Jumlah keseluruhan data dalam dataset.

4. Menghitung F-statistik, berikut rumus perhitungannya:

$$F - statistic = \frac{MS_{between}}{MS_{within}} \quad (2.6)$$

## 2.4 *Balancing Data*

Pengaruh *balancing* data terhadap distribusi data adalah dengan memberikan ukuran yang sama antara masing-masing kelas, sehingga tidak ada kelas yang lebih dominan. Ketidakseimbangan data sering menyebabkan masalah yang cukup serius pada pemodelan prediktif, sehingga model cenderung mempelajari data dengan kelas yang lebih banyak atau kelas mayoritas. Kurangnya referensi model dalam mengenali dan mempelajari kelas minoritas, membuat model cenderung sering memprediksi kelas mayoritas dan mengabaikan kelas minoritas. Hal ini menyebabkan evaluasi cenderung kurang valid dan bisa menurunkan performa model dalam mengenali data dengan baik (Goswami & Roy, 2021; Lukashevich & Klitsunova, 2025).

Oleh karena itu, dalam mengatasi masalah ketidakseimbangan data atau *imbalance* data dapat menggunakan teknik SMOTE. Metode SMOTE diperkenalkan oleh Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, dan W. Philip Kegelmeyer pada tahun 2002, yang terinspirasi dari tulisan Thanh M. Ha dan Horst Bunke tahun 1997 tentang penambahan data dengan operasi tertentu (Chawla et al., 2002). SMOTE adalah teknik menambahkan data sintetis baru pada kelas minoritas agar jumlahnya sama dengan kelas lainnya. Proses penambahan data sintetis baru ini disebut dengan *oversampling*, sehingga setiap kelas memiliki distribusi data yang seimbang atau sama rata (Chawla et al., 2002; Fernandez et al., 2018). Berikut langkah-langkah pembuatan data sintetis menggunakan SMOTE:

1. Mengelompokkan titik data asli berdasarkan kedekatan dengan titik data minoritas yang lain. Sehingga menentukan terlebih dahulu jumlah tetangga terdekat yang akan dicari yakni jumlah  $k$ . Mencari kedekatan dengan titik data minoritas yang lain pada umumnya dengan menggunakan teknik *K-Neighbors* (KNN) , yakni jarak masing-masing data akan dihitung dengan rumus *Euclidian Distance* (Pulungan et al., 2019). Berikut rumus *Euclidian Distance*:

$$d(X_0, X_1) = \sqrt{\sum_{i=1}^n (X_0 - X_1)^2} \quad (2.7)$$

Keterangan :

- $X_0$ : Data minoritas pertama
- $X_1$  : Data minoritas kedua
- $d$  : Jarak *Euclidian*

2. Selanjutnya data yang paling dekat dengan titik data asli akan dimanfaatkan untuk membuat data baru menggunakan interpolasi. Interpolasi adalah variable

acak dengan rentang nilai 0 hingga 1, yang berfungsi sebagai metode matematika dalam memperkirakan nilai diantara dua titik yang diketahui. Nilai 0 merepresentasikan kemiripan penuh data sintetis yang akan dibuat dengan data asli, sedangkan nilai 1 merepresentasikan kemiripan penuh data sintetis yang akan dibuat dengan data minoritas tetangga terdekat yang terpilih. Sehingga nilai interpolasi umumnya 0,5 agar nilainya seimbang, yakni mengandung nilai dari data asli dan data minoritas yang terpilih (Elreedy & Atiya, 2019). Berikut rumus interpolasi:

$$Z = X_0 + w (X - X_0) \quad (2.8)$$

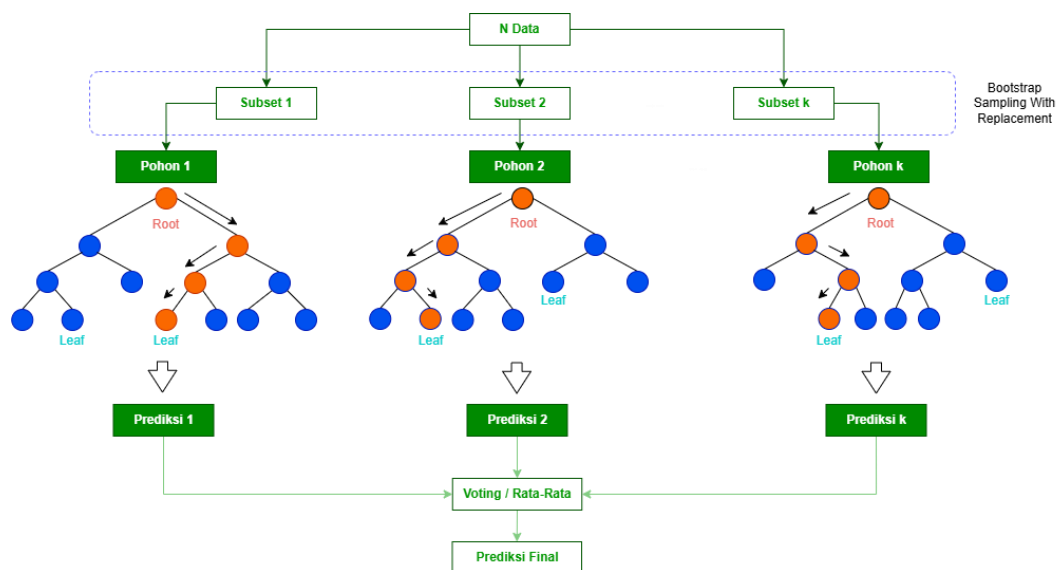
Keterangan :

- $Z$  : Data baru
- $X_0$  : Data kelas minoritas yang dijadikan titik data original
- $X$  : Data kelas minoritas tetangga terdekat yang dijadikan pembandingan
- $w$  : variabel acak dalam rentang  $[0,1]$

## 2.5 Algoritma Random Forest

Algoritma *Random Forest* (RF) diperkenalkan oleh Leo Breiman (Breiman, 2001). RF dikembangkan untuk mengatasi kelemahan yang terdapat pada metode *decision tree* jenis *classification and regression trees* (CART), terutama dalam hal *overfitting*. *Overfitting* adalah kondisi dimana model terlalu beradaptasi dengan data pelatihan, sehingga model kehilangan kemampuan untuk melakukan generalisasi pada data baru. RF disebut juga dengan model *ensemble learning* yakni gabungan model *machine learning* yang sama (Kavlakoglu, 2026; Wan & Yang, 2013). Artinya RF terdiri dari gabungan *decision tree*, sehingga RF dapat menghasilkan model yang lebih stabil dan handal dalam menghadapi variasi data.

Proses pembentukan RF melibatkan pembuatan sejumlah *subset* atau *subdataset* yang dihasilkan melalui pengambilan sampel data secara acak, dan setiap data yang diambil untuk pembuatan *subset* akan dikembalikan kembali ke dalam dataset asli, sehingga dimungkinkan setiap *subset* akan mengandung data yang sama, proses pengambilan sampel data dengan pengembalian tersebut disebut dengan *bootstrap sampling with replacement*. Selanjutnya hasil akhir prediksi RF akan ditentukan berdasarkan *voting* dari seluruh hasil prediksi pohon keputusan yang sudah dibangun atau disebut dengan *aggregating*. Sehingga secara keseluruhan teknik ini bisa disebut dengan *bagging* atau *bootstrap aggregating* (Sibirina & Gladkova, 2012).



Gambar 2.1 Desain model *Random Forest*

Sumber : <https://sainsdata.id/machine-learning/7728/random-forest-untuk-model-regresi-dengan-python-scikit-learn/>

Setiap pohon yang dibangun pada setiap subset, dibentuk dari berbagai node, baik dari *root node* atau titik keputusan akar yaitu keputusan paling penting, hingga *leaf node* atau titik keputusan daun yang umumnya menjadi akhir penentuan

keputusan (Salman et al., 2024). Proses pembentukan node tersebut dapat melalui beberapa metode seperti, *gini impurity* atau perhitungan ketidakmurnian dan *entropy* atau perhitungan ketidakberaturan. Dari perhitungan tersebut akan dilakukan proses evaluasi pemisahan (*split evaluation*) untuk menentukan titik pemisahan terbaik atau untuk menentukan *node split* selanjutnya, proses tersebut dapat dilakukan menggunakan berbagai kriteria, seperti *Gini Index*, *Information Gain*, *Gain Ratio*, dan *Misclassification Rate* (Tangirala, 2020). Pada penelitian ini, proses pembentukan node akan menggunakan perhitungan *gini impurity* dengan proses evaluasi pemisahan menggunakan *gini index* atau *gini split*.

Perhitungan *gini impurity* ini bertujuan untuk mengukur tingkat heterogenitas distribusi kelas dalam suatu node. Dimana semakin kecil nilai *gini index*, maka semakin tinggi tingkat kemurnian node tersebut, menunjukkan bahwa data dalam node semakin homogen atau semakin bersih (Hadi & Sirodj, 2023). Oleh karena itu fitur yang menghasilkan nilai *gini index* terendah setelah proses pemisahan dianggap sebagai fitur terbaik, karena mampu menghasilkan distribusi kelas yang paling homogen. Sehingga, fitur tersebut dipilih sebagai dasar pembentukan node atau cabang pemisahan untuk proses pembentukan keputusan berikutnya. Berikut adalah langkah-langkah pembentukan pohon keputusan dengan *gini impurity* (Tangirala, 2020):

1. Membuat subset dari data acak dengan mengembalikan kembali data yang terambil kedalam dataset asli atau *bootstrap with replacement*. Jumlah subset yang terbentuk menentukan jumlah pohon yang dibangun, umumnya ditentukan dengan parameter  $n\_estimators$ .

2. Pemilihan subset fitur secara acak, dimana masing-masing fitur tersebut akan dihitung tingkat ketidakmurniannya, melalui perhitungan *gini impurity*, berikut rumus *gini impurity*:

$$Gini(S_i) = 1 - \sum_{i=0}^{c-1} p_i^2 \quad (2.9)$$

Keterangan:

- $p_i$  : Nilai probabilitas
- $c$  : Jumlah kelas

3. Menghitung *gini split*, untuk menentukan fitur dan titik pemisahan (*split point*) terbaik yang menghasilkan tingkat kemurnian tertinggi (nilai *impurity* terendah). Fitur dengan nilai *gini split* terkecil akan dipilih sebagai *root node* atau node keputusan teratas, dengan rumus berikut:

$$Gini_{split} = \sum_{i=0}^{c-1} \left( \frac{n_i}{n} \right) Gini(s_i) \quad (2.10)$$

Keterangan:

- $c$  : Jumlah kelas
- $n_i$ : Total data dalam suatu kelas
- $n$  : Total data secara keseluruhan

4. Proses pemilihan fitur dan penentuan titik split terbaik dilakukan secara rekursif atau berulang pada setiap node hingga memenuhi kriteria penghentian, seperti batas kedalaman pohon (*max\_depth*).
5. Setelah seluruh pohon keputusan terbentuk, prediksi akhir model ditentukan melalui mekanisme agregasi, yaitu *majority voting* untuk kasus klasifikasi atau perhitungan rata-rata (*averaging*) untuk kasus regresi.

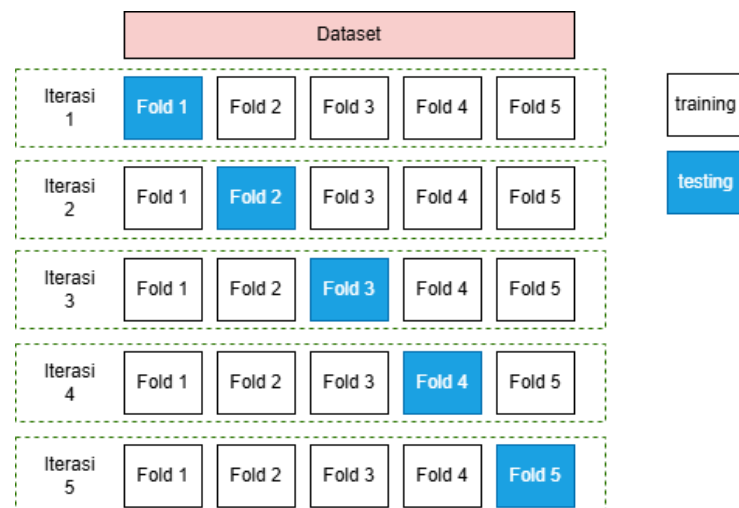
## 2.6 Evaluasi Model

Evaluasi kinerja model prediksi sangat penting untuk memastikan bahwa model yang dibangun dapat menghasilkan prediksi yang akurat dan efektif (Annisa et al., 2025). Dalam penelitian ini, beberapa metrik digunakan untuk mengukur kinerja model, diantaranya *accuracy*, *precision*, *recall*, dan *F1-score* (Erhani et al., 2024). Metrik *accuracy* memberikan gambaran umum tentang seberapa tepat model dalam memprediksi data, tetapi pada dataset yang tidak seimbang dimana jumlah karyawan yang keluar jauh lebih sedikit daripada yang bertahan, metrik ini kurang menggambarkan kinerja model secara keseluruhan (Miller et al., 2024). Oleh karena itu, metrik *precision* dan *recall* lebih relevan, di mana *precision* menunjukkan seberapa tepat model dalam memprediksi karyawan yang akan keluar, sementara *recall* mengukur seberapa baik model dalam menangkap semua karyawan yang berpotensi keluar. Untuk memberikan gambaran yang lebih seimbang antara keduanya, *F1-score* digunakan sebagai harmoni antara *precision* dan *recall* (Miller et al., 2024).

Dalam mengoptimalkan model yang dibangun, dataset akan dibagi menjadi dua bagian, yaitu data latih yang akan digunakan untuk melatih model dan data uji yang digunakan untuk menguji hasil prediksi. Pembagian ini bertujuan untuk memastikan model yang akan dibangun memiliki kemampuan generalisasi yang baik, yakni kemampuan untuk melakukan prediksi yang akurat pada data yang belum pernah dilihat sebelumnya (*testing*). Selain teknik pembagian *train-test-split* dengan pembagian presentase dari keseluruhan dataset, seperti pembagian dengan presentase 80:20, 70:30, 60:40, dimana 80%, 70% atau 60% dari dataset akan

digunakan sebagai data latih dan sisanya akan menjadi data uji. Teknik pembagian yang sering dilakukan lainnya adalah dengan menggunakan teknik validasi silang dengan menggunakan *k-fold*, di mana data dibagi menjadi beberapa subset, dan model dilatih serta diuji secara bergantian pada masing-masing subset untuk mendapatkan estimasi kinerja yang lebih robust (Qiu, 2024).

*K-Fold Cross Validation* merupakan metode evaluasi yang bertujuan mengukur kinerja model terhadap data yang tidak digunakan selama proses pelatihan. Teknik ini dilakukan dengan membagi dataset menjadi  $k$  bagian yang disebut *fold*. Pada setiap iterasi, dalam 1 *fold* akan digunakan sebagai data pengujian, sedangkan  $k-1$  *fold* lainnya digunakan sebagai data pelatihan (Tang et al., 2025). Prosedur ini diulang sebanyak  $k$  kali sehingga setiap bagian dataset memperoleh kesempatan yang sama untuk menjadi data uji. Gambar 2.2 menjelaskan proses evaluasi validasi silang *K-Fold Cross-Validation* dengan contoh *k-fold CV* = 5, berikut penjelasannya:



Gambar 2.2 Cara kerja validasi silang

*K-Fold Cross-Validation* mampu mengurangi bias dan variabilitas hasil evaluasi, serta menghasilkan estimasi performa model yang lebih representatif terhadap data yang belum pernah diamati sebelumnya. Selain itu, teknik ini membantu memastikan bahwa model memiliki kemampuan generalisasi yang baik, tidak hanya optimal pada satu subset data tertentu, tetapi konsisten pada berbagai subset data (Tang et al., 2025). Secara umum, tahapan pelaksanaan *K-Fold Cross-Validation* meliputi:

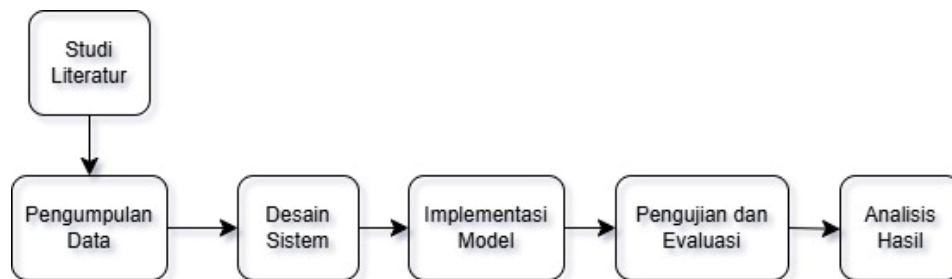
1. Membagi dataset menjadi  $k$  subset yang berukuran relatif sama.
2. Melatih model sebanyak  $k$  kali, dengan menggunakan  $k-1$  subset sebagai data pelatihan dan satu subset sebagai data pengujian pada setiap iterasi.
3. Menghitung nilai rata-rata dari metrik evaluasi yang diperoleh pada setiap iterasi sebagai representasi performa akhir model.
4. Hasil evaluasi dari *K-Fold Cross-Validation* umumnya digunakan untuk mengukur metrik kinerja seperti *accuracy* dan *F1-score*, serta untuk membandingkan performa antar model yang dikembangkan.

## BAB III

### DESAIN DAN IMPLEMENTASI

#### 3.1 Tahapan Penelitian

Tahapan penelitian merupakan alur utama yang menggambarkan proses, metode dan langkah - langkah yang akan digunakan agar penulisan terstruktur dengan baik. Tahapan penelitian diilustrasikan pada Gambar 3.1.



Gambar 3.1 Tahapan penelitian

#### 3.2 Pengumpulan Data

Data penelitian ini didapatkan dari dataset publik *IBM HR Analytics Employee Attrition & Performance*, yang berisi 1440 data dengan 35 fitur terkait kinerja karyawan, kepuasan kerja, dan tingkat keluar karyawan (*attrition*), yang diakses pada 30 September 2025. Fitur-fitur ini mencakup berbagai fitur atau variabel yang dapat mengevaluasi perilaku dan performa karyawan, seperti usia, jabatan, gaji, waktu bekerja, dan kepuasan kerja. Berikut penjelasan dari 35 fitur tersebut yang ditunjukkan pada Tabel 3.1.

Tabel 3.1 Deskripsi fitur

| Fitur            | Keterangan                                                                                              |
|------------------|---------------------------------------------------------------------------------------------------------|
| <i>Age</i>       | Usia Karyawan                                                                                           |
| <i>Attrition</i> | Kelas prediksi penurunan karyawan :<br>0 = Tidak Meninggalkan Perusahaan<br>1 = Meninggalkan Perusahaan |

| Fitur                            | Keterangan                                                                                                                 |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <i>Business Travel</i>           | Kategori frekuensi perjalanan bisnis karyawan <i>ex</i> : Non-Travel, Rarely, dan Frequently                               |
| <i>Daily Rate</i>                | Upah harian karyawan                                                                                                       |
| <i>Department</i>                | Departemen tempat karyawan bekerja <i>ex</i> : Research & Development , Sales, dan HR                                      |
| <i>Distance from Home</i>        | Jarak tempat tinggal karyawan dari kantor (dalam mil)                                                                      |
| <i>Education</i>                 | Tingkat pendidikan karyawan :<br>1 = 'Below College', 2 = 'College', 3 = 'Bachelor', 4 = 'Master', 5 = 'Doctor'            |
| <i>Education Field</i>           | Bidang pendidikan karyawan <i>ex</i> : Life Science, Medical, Marketing, Thecnical Degree anda Other                       |
| <i>Employee Count</i>            | Jumlah total karyawan di perusahaan (nilai konstan = 1 di dataset ini)                                                     |
| <i>Employee Number</i>           | Nomor identifikasi unik karyawan                                                                                           |
| <i>Environment Satisfaction</i>  | Tingkat kepuasan karyawan terhadap lingkungan kerja :<br>1 = 'Low', 2 = 'Medium', 3 = 'High', 4 = 'Very High'              |
| <i>Gender</i>                    | Jenis kelamin karyawan <i>ex</i> :<br>Male = Pria : 60%, Female = Wanita : 40%                                             |
| <i>Haourly Rate</i>              | Upah per jam karyawan                                                                                                      |
| <i>Job Involvement</i>           | Tingkat keterlibatan karyawan dalam pekerjaan :<br>1 = 'Low', 2 = 'Medium', 3 = 'High', 4 = 'Very High'                    |
| <i>Job Level</i>                 | Tingkatan jabatan karyawan:<br>1 = Level 1, 2 = Level 2, 3 = Level 3, 4 = Level 4, 5 = Level 5                             |
| <i>Job Role</i>                  | Peran atau jabatan karyawan <i>ex</i> :<br>Research Director, Sales Executive, Laboratory Thecnician and Other             |
| <i>Job Satisfaction</i>          | Kepuasan karyawan terhadap pekerjaannya:<br>1 = 'Low', 2 = 'Medium', 3 = 'High', 4 = 'Very High'                           |
| <i>Marital Status</i>            | Status pernikahan karyawan <i>ex</i> :<br>Single, Married, dan Divorced                                                    |
| <i>Mounthly Income</i>           | Pendapatan bulanan karyawan                                                                                                |
| <i>Mountly Rate</i>              | Tarif bulanan yang dikenakan untuk karyawan                                                                                |
| <i>Num Companies Worked</i>      | Jumlah perusahaan yang pernah menjadi tempat kerja karyawan                                                                |
| <i>Over 18</i>                   | Indikator apakah karyawan berusia di atas 18 tahun (konstan = Yes di dataset ini)                                          |
| <i>Over Time</i>                 | Apakah karyawan sering bekerja lembur (Yes atau No)                                                                        |
| <i>Percent Salary Hike</i>       | Persentase kenaikan gaji karyawan setiap tahun                                                                             |
| <i>Performance Rating</i>        | Peringkat kinerja karyawan:<br>1 = 'Low', 2 = 'Good', 3 = 'Excellent', 4 = 'Outstanding'                                   |
| <i>Relationship Satisfaction</i> | Kepuasan karyawan terhadap hubungan interpersonal di tempat kerja:<br>1 = 'Low', 2 = 'Medium', 3 = 'High', 4 = 'Very High' |
| <i>Standard Hours</i>            | Jam kerja standar (nilai konstan = 80 jam di dataset ini)                                                                  |
| <i>Stock Option Level</i>        | Tingkat opsi saham yang dimiliki karyawan                                                                                  |

| Fitur                             | Keterangan                                                                                     |
|-----------------------------------|------------------------------------------------------------------------------------------------|
| <i>Total Working Years</i>        | Jumlah total tahun bekerja karyawan                                                            |
| <i>Training Times Last Year</i>   | Jumlah pelatihan yang diikuti karyawan tahun lalu                                              |
| <i>Work-Life Balance</i>          | Tingkat keseimbangan kerja-hidup karyawan :<br>1 = 'Bad', 2 = 'Good', 3 = 'Better', 4 = 'Best' |
| <i>Years at Company</i>           | Jumlah tahun bekerja di perusahaan saat ini                                                    |
| <i>Years in Current Role</i>      | Jumlah tahun dalam peran atau jabatan saat ini                                                 |
| <i>Years Since Last Promotion</i> | Jumlah tahun sejak promosi terakhir                                                            |
| <i>Years with Current Manager</i> | Jumlah tahun bekerja dengan manajer saat ini                                                   |

Tabel 3.2 Dataset *Employee Attrition*

| Age | Attrition | BusinessTravel    | Daily Rate | Department             | Distance From Home | Education | Education Field | ... | Years With Curr Manager |
|-----|-----------|-------------------|------------|------------------------|--------------------|-----------|-----------------|-----|-------------------------|
| 41  | Yes       | Travel Rarely     | 1102       | Sales                  | 1                  | 2         | Life Sciences   | ... | 5                       |
| 49  | No        | Travel Frequently | 279        | Research & Development | 8                  | 1         | Life Sciences   | ... | 7                       |
| 37  | Yes       | Travel Rarely     | 1373       | Research & Development | 2                  | 2         | Other           | ... | 0                       |
| 33  | No        | Travel Frequently | 1392       | Research & Development | 3                  | 4         | Life Sciences   | ... | 0                       |
| 27  | No        | Travel Rarely     | 591        | Research & Development | 2                  | 1         | Medical         | ... | 2                       |
| ... | ...       | ...               | ...        | ...                    | ...                | ...       | ...             | ... | ...                     |
| 36  | No        | Travel Frequently | 884        | Research & Development | 23                 | 2         | Medical         | ... | 3                       |
| 39  | No        | Travel Rarely     | 613        | Research & Development | 6                  | 1         | Medical         | ... | 7                       |
| 27  | No        | Travel Rarely     | 155        | Research & Development | 4                  | 3         | Life Sciences   | ... | 3                       |
| 49  | No        | Travel Frequently | 1023       | Sales                  | 2                  | 3         | Medical         | ... | 8                       |
| 34  | No        | Travel Rarely     | 628        | Research & Development | 8                  | 3         | Medical         | ... | 2                       |

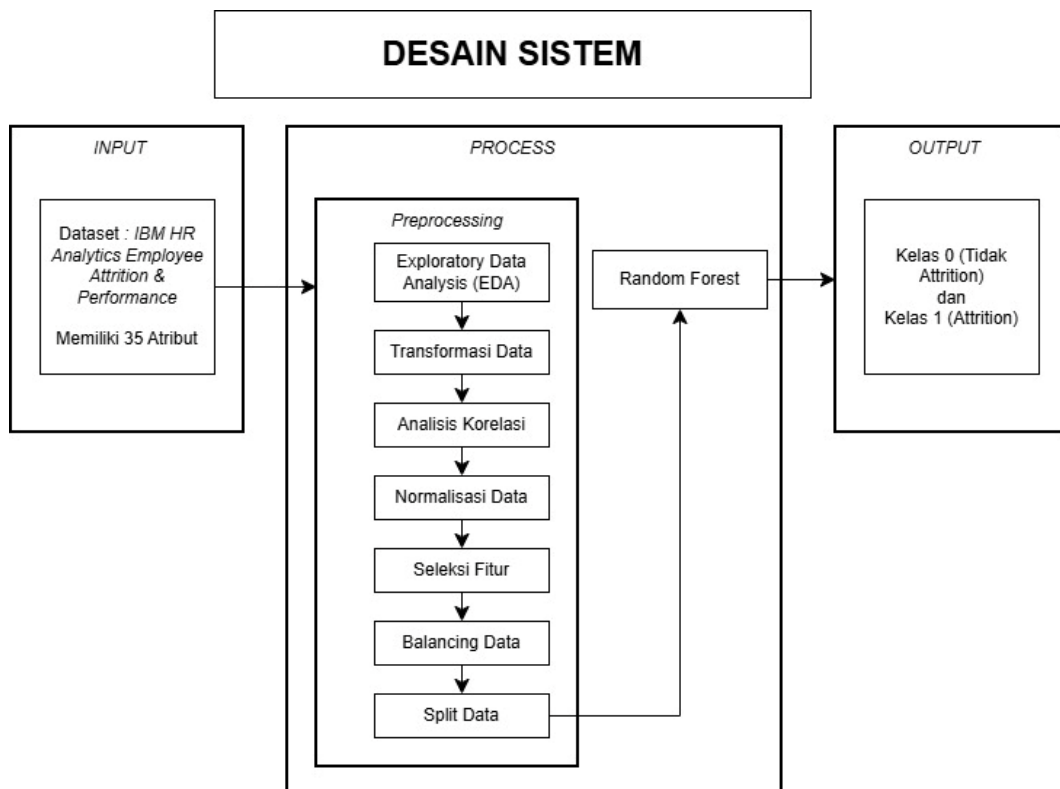
Dalam dataset diatas dari 35 fitur, terdapat 1 fitur target yakni Attrition.

Fitur Attrition memiliki 2 kelas prediksi, yakni kelas “No” sebagai karyawan diprediksi tidak keluar, dan kelas “Yes” yang menunjukkan karyawan diprediksi keluar. Fitur Attrition digunakan sebagai label output dalam pembelajaran model machine learning untuk memprediksi adanya penurunan jumlah karyawan. Dan 34

fitur lainnya bertujuan untuk membangun model serta sebagai wawasan yang berguna dalam pengembangan model machine learning yang terdiri dari 26 data numerikal dan 8 data kategorikal.

### 3.3 Desain Sistem

Desain sistem ini memperlihatkan alur tahapan dari sistem prediksi penurunan jumlah karyawan melalui prediksi *attrition* yang diilustrasikan pada Gambar 3.2.



Gambar 3.2 Desain sistem pada tahapan pembangunan model prediksi

Bahasa *python* dengan input dataset sejumlah 1470 akan digunakan pada penelitian ini. Setelah pengumpulan data, proses selanjutnya adalah *preprocessing* yang dimulai dengan EDA (*Exploratory Data Analysis*) yang berisi pengecekan data dan visualisasi, kemudian tahap transformasi data yakni merubah data yang

masih berbentuk kategorikal menjadi numerical menggunakan *encoding* dan *one-hot encoding*. Setelah itu data akan dianalisis untuk melihat korelasi antara setiap masing-masing fitur yang ada. Dilanjutkan dengan normalisasi untuk menyamakan skala dari semua fitur, dan melakukan analisis untuk mengetahui perbedaan signifikansi antara setiap fitur dengan kelas yang ada pada fitur target. Jika pemilihan fitur telah selesai, tahapan selanjutnya yakni penyeimbangan data menggunakan SMOTE dan pembagian data menjadi data latih serta data uji. Selanjutnya Pembangunan model prediksi dengan algoritma *random forest*, serta mengevaluasi performa model.

### 3.4 Preprocessing

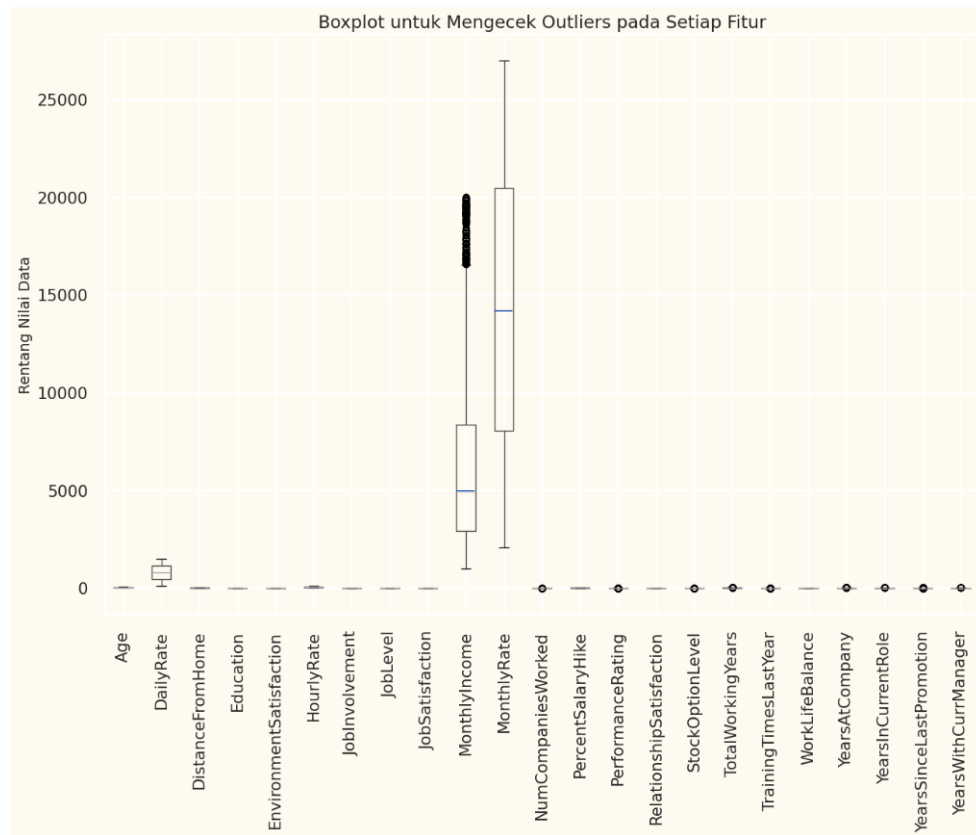
*Preprocessing* data pada penelitian ini bertujuan untuk memastikan bahwa data yang digunakan memiliki kualitas tinggi dan siap untuk digunakan sebagai model pembelajaran. Proses *Preprocessing* meliputi:

#### 3.4.1 Exploratory Data Analysis (EDA)

Analisis eksplorasi data atau EDA bertujuan untuk memahami persebaran dan distribusi pada fitur-fitur dalam dataset, serta hubungannya terhadap fitur target. Sehingga EDA memberikan informasi atau wawasan mengenai potensi masalah seperti ketidakseimbangan data yang dapat mempengaruhi akurasi model. Dalam proses EDA, dilakukan data *cleaning* yang meliputi identifikasi data ganda atau duplikat, identifikasi nilai yang hilang (*missing value*) pada fitur-fitur dalam dataset, identifikasi adanya *outlier* yang berpengaruh, serta visualisasi data dalam berbagai bentuk, seperti diagram batang, diagram lingkaran dan histogram, untuk

memberikan gambaran yang lebih jelas mengenai pola dalam dataset, berikut penjelasannya:

1. Pengecekan data duplikat: Tidak ditemukan adanya data ganda yang sama atau duplikat dalam dataset.
2. Pengecekan nilai yang hilang (*missing value*): Tidak ditemukan adanya *missing value* dalam dataset.
3. Penghapusan fitur yang tidak diperlukan, diantaranya ‘Over18’, ‘EmployeeCount’, ‘EmployeeNumber’, dan ‘StandardHours’, dimana 1 data kategorikal dan 3 lainnya data numerikal. Sehingga dari 35 fitur tersisa 31 fitur.
4. Pengecekan outlier: Walaupun terdapat outlier dalam dataset, namun dalam penelitian ini, mengidentifikasi bahwa outlier tersebut tidak memiliki pengaruh signifikan, karena nilai-nilai yang terdeteksi masih berada dalam kisaran wajar, seperti pada fitur ‘MounthlyIncome’ yang dinilai outlier yang diketahui melalui metode perhitungan *inquartile range* (IQR). Dianggap masih wajar karena pendapatan setiap individu tetap realistis dengan posisi jabatan yang mereka pegang. Berikut gambaran outlier pada data numerikal yang terdeteksi:



Gambar 3.3 Diagram batang dalam mendeteksi outliers

Tabel 3.3 Dataset *Employee Attrition*

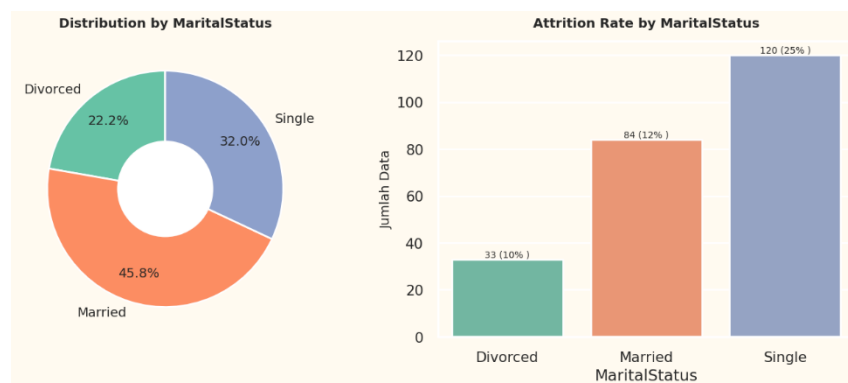
| Hourly Rate | Job Involvement | Job Level | Job Role          | Job Satisfaction | ... | Monthly Income | Monthly Rate |
|-------------|-----------------|-----------|-------------------|------------------|-----|----------------|--------------|
| 65          | 2               | 5         | Manager           | 3                | ... | 19999          | 5678         |
| 55          | 1               | 5         | Research Director | 3                | ... | 19973          | 20284        |
| 92          | 3               | 5         | Manager           | 1                | ... | 19943          | 18575        |
| 53          | 3               | 5         | Manager           | 2                | ... | 19926          | 17053        |
| 78          | 3               | 5         | Manager           | 1                | ... | 19859          | 21199        |

Tabel 3.4 Dataset *Employee Attrition*

| Hourly Rate | Job Involvement | Job Level | Job Role             | Job Satisfaction | ... | Monthly Income | Monthly Rate |
|-------------|-----------------|-----------|----------------------|------------------|-----|----------------|--------------|
| 32          | 3               | 1         | Research Scientist   | 3                | ... | 1009           | 26999        |
| 73          | 3               | 1         | Research Scientist   | 4                | ... | 1051           | 13493        |
| 74          | 3               | 1         | Sales Representative | 2                | ... | 1052           | 23384        |
| 89          | 3               | 1         | Sales Representative | 4                | ... | 1081           | 16019        |
| 45          | 4               | 1         | Sales Representative | 1                | ... | 1091           | 10642        |

Dalam tabel 3.3 dan tabel 3.4, pada kolom ke-4 fitur ‘JobRole’ dijelaskan bahwa terdapat perbedaan pendapatan bulanan seorang ‘Manager’ dengan ‘Sales Representative’. Jabatan Manager dianggap outlier karena tidak banyak orang dengan jabatan tersebut, sehingga ‘Manager’ dengan ‘JobLevel’ 5 akan memiliki ‘MounthlyIncome’ yang lebih banyak dibanding dengan Jabatan dengan ‘JobLevel’ 1. Maka IQR akan mendeteksi nilai pada fitur ‘MounthlyIncome’ seorang ‘Manager’ sebagai outlier karena melebihi batas atas dari penetapan hasil *quartile* keseluruhan data. Oleh karena itu, outlier tersebut tidak memerlukan intervensi lebih lanjut.

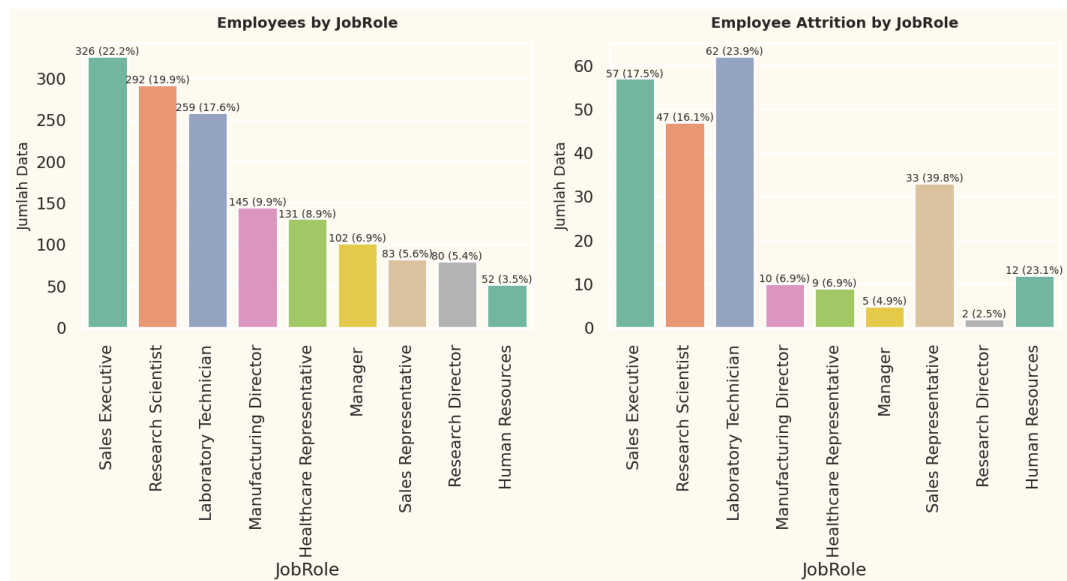
5. Gambar 3.4 merupakan visualisasi pada fitur MaritalSatatus menggunakan *pie cart* dan *bar cart* atau gabungan diagram lingkaran untuk menggambarkan proporsi persebaran data secara keseluruhan, dan diagram batang untuk menunjukkan perbandingan persebaran data pada fitur MaritalSatatus berdasarkan karyawan yang teridentifikasi ‘Attrition’.



Gambar 3.4 Diagram pada distribusi fitur *MaritalStatus*

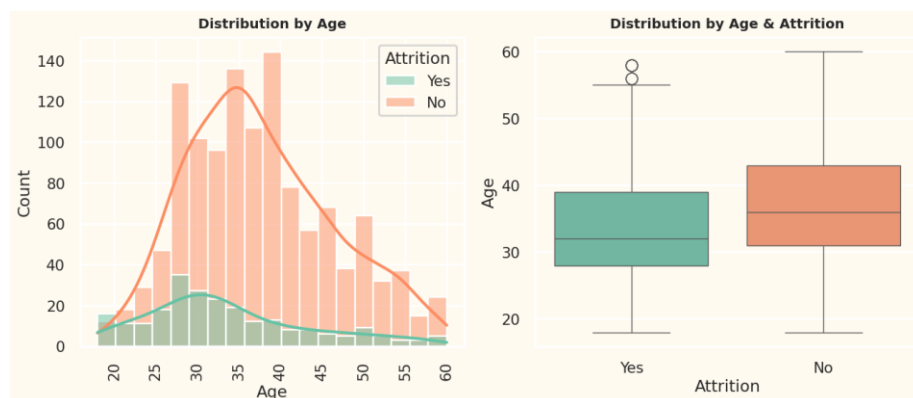
6. Gambar 3.5 merupakan visualisasi pada fitur JobRorel yang menggunakan *bar cart* atau diagram batang untuk menenunjukkan persebaran data secara keseluruhan, dan untuk menunjukkan perbandingan persebaran data pada fitur

JobRole berdasarkan karyawan yang teridentifikasi “Attrition” pada diagram batang sisi kiri. *Bar cart* dipilih karena jenis data dalam fitur JobRole yang banyak , serta untuk memperjelas sehingga diagram mudah terbaca dan mudah dipahami.



Gambar 3.5 Diagram pada distribusi fitur *JobRole*

7. Gambar 3.6 merupakan visualisasi pada fitur Age yang menggunakan diagram histogram untuk menunjukkan distribusi frekuensi fitur Age secara keseluruhan, dan di sisi kiri menunjukkan visualisasi untuk melihat outlier berdasarkan distribusi fitur Age pada data yang teridentifikasi “Attrition”.



Gambar 3.6 Diagram pada distribusi fitur *Age*

Berdasarkan analisis eksplorasi data, dari 35 fitur yang ada, tersisa 31 fitur untuk digunakan pada proses selanjutnya.

### 3.4.2 Transformasi Data

Transformasi data dilakukan untuk mengonversi data kategorikal menjadi data numerikal agar dapat diproses oleh algoritma pembelajaran mesin. Dalam penelitian ini, data kategorikal yang memiliki dua nilai akan diubah menjadi numerikal melalui proses label *encoding* seperti fitur 'Gender' dan 'Attrition'. Label encoding biasanya mengkonversi data kategorikal dalam bentuk angka unik, misalnya dalam fitur 'Gender', nilai 'Female' akan dikonversi menjadi '0' dan 'Male' akan dikonversi menjadi '1'. Begitupula fitur 'Attrition', dikonversi dari nilai 'Yes' menjadi '1' dan nilai 'No' menjadi '0'.

Tabel 3.5 Transformasi *encoding*

| Age | Attrition | Daily Rate | Distance From Home | Education | Environment Satisfaction | Gender | Hourly Rate | ... |
|-----|-----------|------------|--------------------|-----------|--------------------------|--------|-------------|-----|
| 41  | 1         | 1102       | 1                  | 2         | 2                        | 0      | 94          | ... |
| 49  | 0         | 279        | 8                  | 1         | 3                        | 1      | 61          | ... |
| 37  | 1         | 1373       | 2                  | 2         | 4                        | 1      | 92          | ... |
| 33  | 0         | 1392       | 3                  | 4         | 4                        | 0      | 56          | ... |
| 27  | 0         | 591        | 2                  | 1         | 1                        | 1      | 40          | ... |

Sedangkan data kategorikal dengan nilai lebih dari dua akan diubah menjadi numerical melalui proses *one-hot encoding* seperti fitur 'BusinessTravel', 'Department', 'EducationField', 'JobRole', 'MaritalStatus', dan 'OverTime'. Sementara dalam proses *one-hot encoding* akan mengkonversi setiap nilai dalam data kategorikal menjadi nilai biner. Pada tabel 3.6 memberikan gambaran data kategorikal setelah melalui proses *one-hot-encoding*, sebagai contoh pada fitur

‘Business Travel’ yang memiliki nilai data karegorikal berupa ‘Non-Travel’, ‘Rarely’, dan ‘Frequently’. Nilai data tersebut akan dipecah atau ditransformasikan menjadi fitur baru seperti, ‘BusinessTravel\_Non-Travel’, ‘BusinessTravel\_Travel\_Rarely’, dan ‘BusinessTravel\_Travel\_Frequently’. Sehingga apabila dalam data sebelumnya pada fitur ‘Business Travel’ berisi nilai ‘Rarely’, maka setelah *one-hot-encoding*, pada fitur baru ‘BusinessTravel\_Travel\_Rarely’ akan bernilai ‘1’, sedangkan fitur baru pecahan lainnya dari fitur ‘Business Travel’ akan bernilai ‘0’.

Tabel 3.6 Transformasi one-hot encoding

| BusinessTravel_Non-Travel | BusinessTravel_Travel_Frequently | BusinessTravel_Travel_Rarely | Department_Human Resources | Department_Research & Development | Department_Sales | ... |
|---------------------------|----------------------------------|------------------------------|----------------------------|-----------------------------------|------------------|-----|
| 0                         | 0                                | 1                            | 0                          | 0                                 | 1                | ... |
| 0                         | 1                                | 0                            | 0                          | 1                                 | 0                | ... |
| 0                         | 0                                | 1                            | 0                          | 1                                 | 0                | ... |
| 0                         | 1                                | 0                            | 0                          | 1                                 | 0                | ... |
| 0                         | 0                                | 1                            | 0                          | 1                                 | 0                | ... |

### 3.4.3 Analisis Korelasi (*Pearson*)

Untuk mengidentifikasi hubungan antar variable dilakukan analisis korelasi antara fitur-fitur dalam dataset. Tujuan dari korelasi ini adalah untuk mengukur kedekatan hubungan antara fitur-fitur tersebut. Fitur yang saling berkorelasi tinggi akan dipertimbangkan untuk dihapus demi mengurangi adanya informasi ganda, karena dalam analisis korelasi terdapat skala penilaian 0 hingga 1, hubungan antar fitur yang semakin mendekati nilai 1 akan dianggap sebagai data dengan informasi yang hampir sama, sedangkan hubungan antar fitur yang nilainya semakin mendekati nilai 0, maka dianggap fitur tersebut murni dan memiliki banyak

informasi yang diperlukan. Berikut langkah-langkah perhitungan koefisien korelasi fitur menggunakan *pearson*:

- a. Mencari nilai rata-rata pada masing-masing fitur, sesuai dengan persamaan 2.1.

Sebagai contoh perhitungan, yang akan digunakan sebagai variabel  $X_i$  adalah fitur 'Age', sementara variabel  $Y_i$  adalah fitur 'Attrition'.

- b. Pengurangan dari nilai data asli dengan nilai rata-rata masing-masing fitur.

Langkah berikutnya dijabarkan dalam tabel berikut:

Tabel 3.7 Perhitungan korelasi fitur

| No.   | ( $X_i$ ) | ( $Y_i$ ) | ( $\bar{X}$ ) | ( $\bar{Y}$ ) | ( $X_i - \bar{X}$ ) | ( $Y_i - \bar{Y}$ ) | ( $X_i - \bar{X}$ )( $Y_i - \bar{Y}$ ) | ( $X_i - \bar{X}$ ) <sup>2</sup> | ( $Y_i - \bar{Y}$ ) <sup>2</sup> |
|-------|-----------|-----------|---------------|---------------|---------------------|---------------------|----------------------------------------|----------------------------------|----------------------------------|
| 1     | 41        | 1         | 37,2          | 0,2           | 3,8                 | 0,8                 | 3,04                                   | 14,44                            | 0,64                             |
| 2     | 49        | 0         | 37,2          | 0,2           | 11,8                | -0,2                | -2,36                                  | 139,24                           | 0,04                             |
| 3     | 37        | 1         | 37,2          | 0,2           | -0,2                | 0,8                 | -0,16                                  | 0,04                             | 0,64                             |
| 4     | 33        | 0         | 37,2          | 0,2           | -4,2                | -0,2                | 0,84                                   | 17,64                            | 0,04                             |
| 5     | 27        | 0         | 37,2          | 0,2           | -10,2               | -0,2                | 2,04                                   | 104,04                           | 0,04                             |
| 6     | 36        | 0         | 37,2          | 0,2           | -1,2                | -0,2                | 0,24                                   | 1,44                             | 0,04                             |
| 7     | 39        | 0         | 37,2          | 0,2           | 1,8                 | -0,2                | -0,36                                  | 3,24                             | 0,04                             |
| 8     | 27        | 0         | 37,2          | 0,2           | -10,2               | -0,2                | 2,04                                   | 104,04                           | 0,04                             |
| 9     | 49        | 0         | 37,2          | 0,2           | 11,8                | -0,2                | -2,36                                  | 139,24                           | 0,04                             |
| 10    | 34        | 0         | 37,2          | 0,2           | -3,2                | -0,2                | 0,64                                   | 10,24                            | 0,04                             |
| Total |           |           |               |               |                     |                     | 3,6                                    | 533,6                            | 1,6                              |

- c. Langkah selanjutnya adalah, menghitung akar kuadrat dari, total nilai pada kolom 9 dikali dengan total nilai pada kolom 10.

$$\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2} = \sqrt{(533,6)(1,6)} = 29,2192$$

- d. Menghitung nilai korelasi berdasarkan rumus pada langkah a.

$$r = \frac{3,6}{29,2192}$$

$$= 0,123206778$$

- e. Lakukan langkah a hingga d kepada semua masing-masing fitur dengan seluruh fitur lainnya. Kemudian menghapus fitur dengan nilai korelasi atau nilai  $r$  berdasarkan interpretasi yang telah ditentukan.

#### 3.4.4 Normalisasi Data

Normalisasi data dilakukan untuk menyamakan skala dari semua fitur yang digunakan dalam model, sehingga tidak ada fitur yang mendominasi proses pelatihan. Proses ini memastikan bahwa setiap fitur memiliki kontribusi yang seimbang dalam model. Metode ini relatif lebih tahan terhadap *outlier* karena tidak membatasi nilai data pada rentang tertentu, seperti nilai minimum dan maksimum. *Z-score* melakukan standarisasi berdasarkan rata-rata dan simpangan baku, sehingga menghasilkan distribusi data dengan mean 0 dan standar deviasi 1. Pendekatan ini lebih fleksibel dan sesuai untuk algoritma berbasis perhitungan jarak, seperti *K-Nearest Neighbor* (KNN), karena memastikan setiap fitur memiliki kontribusi yang proporsional dalam proses pengukuran jarak (Pagan et al., 2023). Perhitungan *Z-score* didasari dari rumus perhitungan standar deviasi, kemudian dibagi dengan hasil dari nilai data dikurangi nilai rata-rata setiap fitur (Handayani & Ikrimach, 2020).

$$Z = \frac{X - \mu}{\sigma} \quad (3.1)$$

Keterangan :

- $z$  : Nilai Z-Score yang dihasilkan.
- $x$  : Nilai data dari setiap fitur.
- $\mu$  : Mean (rata-rata) dari setiap fitur.
- $\sigma$  : Standar deviasi dari fitur yang bersangkutan.

Tabel 3.8 Implementasi dataset setelah dinormalisasi

| Age      | Daily Rate | Distance From Home | Education | Environment Satisfaction | Gender   | ... | OverTime_Yes |
|----------|------------|--------------------|-----------|--------------------------|----------|-----|--------------|
| 0,44635  | 0,742527   | -1,01091           | -0,89169  | 0,44635                  | 2,280906 | ... | 1,591746     |
| 1,322365 | -1,29777   | -0,14715           | -1,86843  | 1,322365                 | -0,43842 | ... | -0,62824     |
| 0,008343 | 1,414363   | -0,88752           | -0,89169  | 0,008343                 | 2,280906 | ... | 1,591746     |
| -0,42966 | 1,461466   | -0,76412           | 1,061787  | -0,42966                 | -0,43842 | ... | 1,591746     |
| -1,08668 | -0,52429   | -0,88752           | -1,86843  | -1,08668                 | -0,43842 | ... | -0,62824     |
| ...      | ...        | ...                | ...       | ...                      | ...      | ... | ...          |
| -0,10116 | 0,202082   | 1,703764           | -0,89169  | 0,254625                 | 0,816497 | ... | -0,62824     |
| 0,227347 | -0,46975   | -0,39394           | -1,86843  | 1,169781                 | 0,816497 | ... | -0,62824     |
| -1,08668 | -1,60518   | -0,64073           | 0,085049  | -0,66053                 | 0,816497 | ... | 1,591746     |
| 1,322365 | 0,546677   | -0,88752           | 0,085049  | 1,169781                 | 0,816497 | ... | -0,62824     |
| -0,32016 | -0,43257   | -0,14715           | 0,085049  | -0,66053                 | 0,816497 | ... | -0,62824     |

### 3.4.5 Seleksi Data (F-Test)

Fitur seleksi melalui ANOVA melalui rangkaian proses F-Test dilakukan untuk mengidentifikasi fitur-fitur yang memiliki hubungan signifikan dengan variabel target. F-Test mengukur variabilitas antar kelompok data dan berfungsi untuk mengidentifikasi fitur yang memberikan kontribusi signifikan terhadap model. Fitur dengan nilai 'p' diatas 0,05 akan diseleksi untuk meningkatkan kinerja model karena dianggap tidak signifikan dalam mempengaruhi variable target , sehingga dengan menyeleksi variable tersebut untuk meningkatkan kinerja model dan mengurangi *overfitting*. Untuk mengetahui nilai *p\_value*, dalam seleksi ini, langkah pertama perlu dihitung nilai *F\_statistic* antara masing masing fitur dengan kelas target. Berikut langkah-langkah perhitungannya:

1. Menghitung *Sum of Square Between Groups*, atau menghitung perbedaan rata-rata grup atau kelas pada fitur target, dalam hal ini terdapat dua kelas yakni kelas '0' sebagai prediksi karyawan tidak keluar dan kelas '1' sebagai prediksi

karyawan keluar, dengan rata-rata total keseluruhan data. Perhitungan menggunakan persamaan 2.2.

Tabel 3.9 Implementasi perhitungan *SS Between Groups*

| No.                                     | Age<br>( $X_i$ ) | Attrition | Attrition                                 |            |
|-----------------------------------------|------------------|-----------|-------------------------------------------|------------|
|                                         |                  |           | 1                                         | 0          |
| 1                                       | 0,44635          | 1         | 0,44635                                   | 1,322365   |
| 2                                       | 1,322365         | 0         | 0,008343                                  | -0,42966   |
| 3                                       | 0,008343         | 1         |                                           | -1,08668   |
| 4                                       | -0,42966         | 0         |                                           | -0,10116   |
| 5                                       | -1,08668         | 0         |                                           | 0,227347   |
| 6                                       | -0,10116         | 0         |                                           | -1,08668   |
| 7                                       | 0,227347         | 0         |                                           | 1,322365   |
| 8                                       | -1,08668         | 0         |                                           | -0,32016   |
| 9                                       | 1,322365         | 0         | <b>Rata-Rata (<math>\bar{X}_j</math>)</b> |            |
| 10                                      | -0,32016         | 0         | 0,22735                                   | -0,0190329 |
| <b>Jumlah data</b>                      | 10               |           | 2                                         | 8          |
| <b>Rata-rata (<math>\bar{X}</math>)</b> | 0,030243         |           |                                           |            |

Pada tabel 3.9 menjelaskan langkah-langkah perhitungan berdasarkan fitur Age dengan kelas target 'Attrition' yakni kelas '1' dan kelas '0' , kemudian merubah tabel tersebut menjadi tabel kontingensi pada kolom 4 dan 5, yakni mengelompokkan nilai data 'Age' sesuai dengan kelas target pada fitur Attrition. Selanjutnya mengitung rata-rata berdasarkan data yang sudah di kelompokkan berdasarkan kelasnya, serta rata-rata pada seluruh data pada fitur Age. Kemudian menghitung jumlah data pada masing-masing kelas dan pada seluruh data pada fitur Age. Dilanjutkan dengan rumus berikut:

$$(\bar{X}_1 - \bar{X})^2 = 0,2273465 - 0,030243 = 0,1971035^2 = 0,03884979$$

$$(\bar{X}_0 - \bar{X})^2 = (-0,019032875) - 0,030243 = -0,049275875^2 = 0,002428112$$

$$\sum_{j=1}^k n_j (\bar{X}_j - \bar{X})^2 = (2 \times 0,03884979) + (8 \times 0,002428112) = 0,097124474$$

2. Menghitung total *Sum of Squares Within Groups* atau  $SS_{within}$ , perhitungan menggunakan persamaan 2.3.

Tabel 3.10 Implementasi perhitungan SS total

| Age<br>( $X_i$ ) | Attrition | Attrition                 |            | $(X_{ij} - \bar{X}_j)$ |             | $(X_{ij} - \bar{X}_j)^2$ |             |
|------------------|-----------|---------------------------|------------|------------------------|-------------|--------------------------|-------------|
|                  |           | 1                         | 0          | 1                      | 0           | 1                        | 0           |
| 0,44635          | 1         | 0,44635                   | 1,322365   | 0,2190035              | 1,341397875 | 0,047962533              | 1,799348259 |
| 1,322365         | 0         | 0,008343                  | -0,42966   | 0,2190035              | -0,41062713 | 0,047962533              | 0,168614636 |
| 0,008343         | 1         |                           | -1,08668   |                        | -1,06764713 |                          | 1,139870384 |
| -0,42966         | 0         |                           | -0,10116   |                        | -0,08212713 |                          | 0,006744865 |
| -1,08668         | 0         |                           | 0,227347   |                        | 0,246379875 |                          | 0,060703043 |
| -0,10116         | 0         |                           | -1,08668   |                        | -1,06764713 |                          | 1,139870384 |
| 0,227347         | 0         |                           | 1,322365   |                        | 1,341397875 |                          | 1,799348259 |
| -1,08668         | 0         |                           | -0,32016   |                        | -0,30112713 |                          | 0,090677545 |
| 1,322365         | 0         | Rata-Rata ( $\bar{X}_j$ ) |            |                        |             | Total1                   | Total0      |
| -0,32016         | 0         | 0,22735                   | -0,0190329 |                        |             | 0,095925066              | 6,205177374 |

Pada tabel 3.10 menjelaskan langkah-langkah perhitungan berdasarkan fitur Age dengan kelas target 'Attrition' yakni kelas '1' dan kelas '0', kemudian merubah tabel tersebut menjadi tabel kontingensi pada kolom 3 dan 4, yakni mengelompokkan nilai data 'Age' sesuai dengan kelas target pada fitur Attrition. Selanjutnya menghitung rata-rata berdasarkan data yang sudah di kelompokkan berdasarkan kelasnya. Nilai data asli pada fitur Age akan dikurangi dengan nilai rata-rata perkelas yang telah dikelompokkan. Sehingga nilai dari total *Sum of Square* adalah:

$$SS_{within} = \text{Total1} + \text{Total0} = 0,095925066 + 6,205177374 = 6,30110244$$

3. Menghitung  $MS_{between}$  dan  $MS_{within}$ , atau menghitung rata rata *Sum of Square* dari *Between* dan *Within* dengan jumlah data, baik jumlah data pada masing-masing kelas ataupun pada seluruh data. Sehingga diketahui bahwa nilai  $k = 2$ , yakni kelas '0' dan kelas '1', sedangkan nilai  $N$  adalah 10. Berikut adalah contoh perhitungannya menggunakan persamaan 2.4 dan persamaan 2.5.

$$MS_{between} = \frac{0,09712}{1} = 0,09712$$

$$MS_{within} = \frac{6,30110244}{8} = 0,787637805$$

4. Menghitung F-statistik, berikut perhitungannya sesuai persamaan 2.6:

$$F - statistic = \frac{0,09712}{0,78763} = 0,12331$$

### 3.4.6 Balancing Data

Dataset yang digunakan pada penelitian ini memiliki label kelas yang tidak seimbang yakni label ‘1’ pada atribut ‘Attrition’ diprediksi sebagai sebagai “karyawan tidak keluar” yang berjumlah 243 dari dataset, sedangkan label ‘0’ diprediksi sebagai “karyawan keluar” berjumlah 1227 data dari seluruh dataset. Oleh karena itu, untuk mengatasi ketidakseimbangan data tersebut dilakukan *balancing* data menggunakan metode *oversampling* menggunakan SMOTE.

Metode SMOTE digunakan untuk menangani ketidakseimbangan kelas pada data dengan membuat data baru dari kelas minoritas. Berikut adalah contoh langkah-langkah perhitungan dengan metode SMOTE:

- a. Memilih secara acak sampel data dari data minoritas. Sebagai contoh perhitungan data diambil dari fitur *Age*, *DailyRate*, *DistanceFromHome*, *Education*, *EnvironmentSatisfaction*, dan *Gender*.

$$X_0 = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, \\ 0,254624925, 0,816496581]$$

- b. Menghitung jarak *euclidian* antara  $X_0$  dengan semua contoh data lain dalam kelas minoritas, dengan persamaan 2.7. Sebagai contoh, hitung jarak  $X_0$  dengan  $X_1$ :

$$X_0 = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, 0,254624925, 0,816496581]$$

$$X_1 = [0,446350404, 0,742526534, -1,010909343, -0,89168825, -0,660530674, -1,224744871]$$

Menghitung jarak dengan *euclidian*:

$$d(X_0, X_1) =$$

$$\sqrt{((-0,977173664) - 0,446350404)^2 + ((-1,734096594) - 0,742526534)^2 + (1,827157989 - (-1,010909343))^2 + (0,085049252 - (-0,89168825))^2 + (0,254624925 - (-0,660530674))^2 + (0,816496581 - (-1,224744871))^2}$$

$$\sqrt{2,02642077 + 6,133662119 + 8,054626183 + 0,954016148 + 0,837509772 + 4,166666667}$$

$$\sqrt{22,17290166} = 4,708811066$$

Sebagai contoh lainnya, hitung jarak  $X_0$  dengan  $X_2$ :

$$X_0 = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, 0,254624925, 0,816496581]$$

$$X_2 = [0,008342998, 1,414363238, -0,887515111, -0,89168825, 1,169780525, 0,816496581]$$

$$d(X_0, X_1) =$$

$$\sqrt{\frac{((-0,977173664) - 0,008342998)^2 + ((-1,734096594) - 1,414363238)^2 + (1,827157989 - (-0,887515111))^2 + (0,085049252 - (-0,89168825))^2 + (0,254624925 - 1,169780525)^2 + (0,816496581 - 0,816496581)^2}{0,971243091 + 9,912799318 + 7,369450043 + 0,954016148 + 0,837509772 + 0}}$$

$$\sqrt{20,04501837} = 4,477166333$$

- c. Menentukan jumlah KNN atau tetangga terdekat yang akan digunakan, yakni misal mencari tetangga terdekat sejumlah 3 atau 5, dan sebagainya. Sebagai contoh, perhitungan ini mencari tetangga terdekat sejumlah 1.

$$X_0 \text{ dengan } X_1 : 4,708811066$$

$$X_0 \text{ dengan } X_2 : 4,477166333$$

Maka nilai jarak tetangga terdekat  $X_0$  dengan  $X_2$  yang terpilih.

- d. Buat data sampel baru ( $Z$ ) dari  $X_0$  dengan  $X_2$  sesuai persamaan 2.8:

$$Z = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, 0,254624925, 0,816496581] + 0,5 ([0,008342998, 1,414363238, -0,887515111, -0,89168825, 1,169780525, 0,816496581] - [-0,977173664, -1,734096594, 1,827157989, 0,085049252, 0,254624925, 0,816496581])$$

$$Z = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, 0,254624925, 0,816496581] + 0,5 ([0,985516662, 3,148459833, -2,714673101, -0,976737502, 0,9151556, 0])$$

$$Z = [-0,977173664, -1,734096594, 1,827157989, 0,085049252, \\ 0,254624925, 0,816496581] + [0,492758331, 1,574229916, \\ -1,35733655, -0,488368751, 0,4575778, 0]$$

$$Z = [-0,484415333, -0,159866678, 0,469821439, -0,403319499, \\ 0,712202725, 0,816496581]$$

- e. Untuk mengatasi ketidakseimbangan kelas dalam dataset, pilih data minoritas secara acak, hitung jarak dari tetangga terdekat, dan buat lebih banyak sampel hingga jumlah data untuk tiap label kelas seimbang atau sama.

### 3.4.7 Splitting Data

Dataset dibagi menjadi data latih (*training data*) dan data uji (*testing data*). Data latih akan digunakan untuk membangun serta melatih model *machine learning*, khususnya *random forest*. Sedangkan data uji akan digunakan sebagai data pengujian untuk menilai performa model pada data baru yang belum pernah dipelajari oleh *machine learning* sebelumnya. Penelitian ini akan memebagi dataset menjadi data latih dan data uji dengan rasion 80:20, atau 80% data akan digunakan sebagai data latih, sedangkan 20% akan digunakan sebagai data uji.

Penggunaan *K-fold* dan *cross-validation* atau *split train-test* adalah dua teknik pembagian data juga akan digunakan. Tujuan dari pembagian ini adalah untuk mencegah overfitting dan menjamin generalisasi model yang baik dengan melatih model pada satu bagian data dan menguji kinerjanya pada bagian data lainnya.

### 3.5 Algoritma Random Forest

*Random Forest* dibangun dari gabungan beberapa pohon keputusan. Pembentukan beberapa pohon keputusan ini diawali dengan *bootstrap sampling with resampling*. Selanjutnya perhitungan *gini impurity* melalui perhitungan *gini index* dan proses pemisahan node dengan *gini split*, dimana nilai *gini* terkecil akan digunakan sebagai, *root node* untuk akar keputusan terpenting. Nilai *gini index* terkecil dipilih, karena semakin kecil nilai *gini*-nya, maka nilai fitur tersebut homogen atau semakin murni nilai fitur tersebut.

Namun, sebelum masuk kedalam perhitungan *gini* , perlu ditentukan terlebih dahulu *threshold* pada setiap fitur. *Threshold* ditentukan berdasarkan nilai tengah dari fitur sebagai batas ambang untuk membagi data kedalam *subset kiri* dan *subset kanan*. Berikut adalah tahapan perhitungan random forest beserta contoh perhitungannya:

1. Langkah pertama adalah melakukan *bootstrap sampling* dari dataset yang tersedia dengan *resampling* atau pengembalian. Pemilihan secara acak setiap fitur atas setiap pohon keputusan, dapat juga ditentukan dengan melihat nilai dari  $\sqrt{m}$  , atau akar pangkat dari seluruh fitur tanpa fitur target. Pada contoh perhitungan ini, akan menggunakan rumus  $\sqrt{m}$  dalam membentuk *bootstrap sampling*, namun dalam implementasinya *bootstrap sampling* dilakukan dengan melakukan pengambilan data secara acak pada baris data saja dan pemilihan fiturnya berdasarkan parameter *max\_depth*. Implementasi  $\sqrt{m}$  misalnya, total fitur tanpa fitur target adalah 34 fitur, maka  $\sqrt{34}$  adalah 5,83, sehingga fitur

acak yang terpilih di *bootstrap* pertama sejumlah 6 fitur acak. Berikut contoh implementasi 6 fitur acak tersebut:

Tabel 3.11 Rancangan bootstrap sampling 1

| Age      | DailyRate | Distance FromHome | Environment Satisfaction | Job Involvement | Job Satisfaction | Attrition |
|----------|-----------|-------------------|--------------------------|-----------------|------------------|-----------|
| 0,44635  | 0,742527  | -1,01091          | -0,66053                 | 0,379672        | 1,153254         | 1         |
| 1,431867 | -0,97301  | 2,320735          | 1,169781                 | -1,02617        | -1,56791         | 1         |
| 0,008343 | 1,414363  | -0,88752          | 1,169781                 | -1,02617        | 0,2462           | 1         |
| 1,431867 | 0,187208  | -1,01091          | -0,66053                 | 0,379672        | 0,2462           | 1         |
| 2,088878 | -1,22092  | -0,27054          | 1,169781                 | 0,379672        | 0,2462           | 1         |
| -0,10116 | 0,202082  | 1,703764          | 0,254625                 | 1,785511        | 1,153254         | 0         |
| 0,227347 | -0,46975  | -0,39394          | 1,169781                 | -1,02617        | -1,56791         | 0         |
| -1,08668 | -1,60518  | -0,64073          | -0,66053                 | 1,785511        | -0,66085         | 0         |
| 1,322365 | 0,546677  | -0,88752          | 1,169781                 | -1,02617        | -0,66085         | 0         |
| -0,32016 | -0,43257  | -0,14715          | -0,66053                 | 1,785511        | 0,2462           | 0         |

2. Langkah selanjutnya yaitu perhitungan *gini impurity* yang digunakan untuk membuat *root node* atau akar pohon pada setiap pohon keputusan dengan memilih fitur – fitur paling penting berdasarkan kriteria dari *gini impurity*. Untuk mempermudah perhitungan *gini impurity*, perlu merubah tabel diatas menjadi tabel kontingensi, sebagai berikut:

Tabel 3.12 Tabel kontingensi dari fitur *Age*

| Age           | Attrition |   | Total |
|---------------|-----------|---|-------|
|               | 1         | 0 |       |
| >0,336848552  | 4         | 1 | 5     |
| >=0,336848552 | 1         | 4 | 5     |
| Total         | 5         | 5 | 10    |

Sebelum menentukan nilai *gini index* pada setiap *subset*, maka sebelumnya perlu menentukan nilai *threshold* sebagai ambang batas yang dihasilkan dari nilai-nilai tengah fitur terkait, contohnya *threshold* fitur ‘Age’ pada tabel 3.12.

Berikut contoh perhitungan *gini* sesuai dengan persamaan 2.9:

Perhitungan *subset* kanan ( $> 0,336848552$ ):

$$Gini(S_{right}) = 1 - \left( \left( \frac{4}{5} \right)^2 + \left( \frac{1}{5} \right)^2 \right)$$

$$Gini(S_{right}) = 1 - \left( \frac{16}{25} + \frac{1}{25} \right)$$

$$Gini(S_{right}) = 1 - \left( \frac{17}{25} \right)$$

$$Gini(S_{right}) = 0,32$$

Perhitungan *subset* kiri ( $\leq 0,336848552$ ):

$$Gini(S_{left}) = 1 - \left( \left( \frac{1}{5} \right)^2 + \left( \frac{4}{5} \right)^2 \right)$$

$$Gini(S_{left}) = 1 - \left( \frac{1}{25} + \frac{16}{25} \right)$$

$$Gini(S_{left}) = 1 - \left( \frac{17}{25} \right)$$

$$Gini(S_{left}) = 0,32$$

3. Menentukan *gini index* selanjutnya adalah dengan menghitung *gini split*.

Sebagai contoh perhitungannya adalah sebagai berikut sesuai dengan persamaan 2.10:

$$Gini_{split} = \left( \frac{5}{10} \right) \times 0,32 + \left( \frac{5}{10} \right) \times 0,32$$

$$Gini_{split} = 0,32$$

4. Nilai *gini impurity* terkecil akan dijadikan sebagai *root node* dalam membentuk pohon keputusan. Berikut tabel penjelasan lengkap nilai gini 6 fitur:

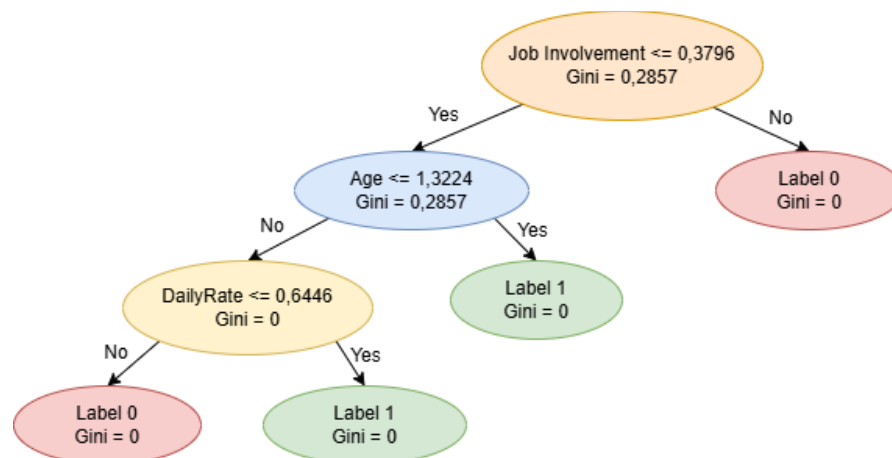
Tabel 3.13 Implementasi perhitungan *gini impurity*

| Fitur                    | Threshold | Attrition    |   |             |   | Total Subset Kanan | Total Subset Kiri | Total | Gini Impurity           |                        |            |
|--------------------------|-----------|--------------|---|-------------|---|--------------------|-------------------|-------|-------------------------|------------------------|------------|
|                          |           | Subset Kanan |   | Subset Kiri |   |                    |                   |       | Gini Index Subset Kanan | Gini Index Subset Kiri | Gini Split |
|                          |           | 1            | 0 | 1           | 0 |                    |                   |       |                         |                        |            |
| Age                      | 0,336849  | 4            | 1 | 1           | 4 | 5                  | 5                 | 10    | 0,32                    | 0,32                   | 0,32       |
| DailyRate                | -0,12268  | 3            | 2 | 3           | 2 | 5                  | 5                 | 10    | 0,48                    | 0,48                   | 0,48       |
| Distance FromHome        | -0,51733  | 2            | 3 | 3           | 2 | 5                  | 5                 | 10    | 0,48                    | 0,48                   | 0,48       |
| Environment Satisfaction | 0,712203  | 3            | 2 | 2           | 3 | 5                  | 5                 | 10    | 0,48                    | 0,48                   | 0,48       |
| JobInvolvement           | 0,379672  | 0            | 3 | 5           | 2 | 3                  | 7                 | 10    | 0                       | 0,408                  | 0,286      |
| JobSatisfaction          | 0,2462    | 1            | 1 | 4           | 4 | 2                  | 8                 | 10    | 0,5                     | 0,5                    | 0,5        |

5. Pada tabel 3.13, fitur ‘JobInvolvement’ terpilih sebagai root dalam *bootstrap sampling* 1 atau dalam subset data 1, karena memiliki nilai *gini impurity* terkecil sebesar 0,286. Selanjutnya, penentuan *rules* pada setiap *node* atau fitur terpilih ditentukan berdasarkan nilai ambang batas (threshold) yang telah ditetapkan sebelumnya, dengan mempertimbangkan nilai gini pada setiap gini subset. Misalnya pada fitur terpilih ‘JobInvolvement’, memiliki nilai gini subset kanan dengan nilai 0, yang menunjukkan bahwa nilai tersebut dianggap telah murni (homogen) dan sudah tidak ada lagi data dengan label yang berbeda. Sementara itu, subset kiri memiliki nilai gini sebesar 0,408, sehingga nilai gini subset kiri dipilih sebagai dasar pembentukan *rules* pohon keputusan karena masih mengandung variasi data yang perlu diproses lebih lanjut.
6. Karena nilai gini subset kiri yang terpilih sebagai dasar pembentukan *rules* pohon keputusan, maka nilai *threshold* yang digunakan untuk menghasilkan nilai gini subset tersebut akan digunakan sebagai kriteria untuk memisahkan dataset pada proses selanjutnya. Sehingga dataset akan dipisah berdasarkan fitur

‘JobInvolvement’ dengan nilai  $\leq 0,336848552$ . Proses ini menghasilkan 2 sub data yang akan dihitung kembali nilai *gini index* hingga *gini split*, untuk menentukan *node* dan *rules* selanjutnya.

7. Proses perhitungan tersebut berlangsung secara iteratif hingga mencapai batas *max\_depth* yang telah ditentukan, atau hingga kondisi penghentian terpenuhi, seperti ketika seluruh data pada suatu *node* telah homogen atau tidak terdapat lagi peningkatan yang signifikan dalam nilai *gini impurity*, sehingga pembentukan *node* dan *rules* pada pohon keputusan dapat dihentikan. Berikut contoh pohon keputusan yang dibangun:



Gambar 3.7 Hasil perancangan pohon keputusan

8. Tahap 1 hingga tahap 7 terus berulang sesuai jumlah pohon keputusan yang dibentuk, yang ditentukan oleh parameter *n\_estimators* atau *n\_trees* dalam model. Selanjutnya, hasil prediksi dari setiap pohon digabungkan melalui mekanisme *majority voting*, atau kelas yang paling sering muncul pada setiap pohon ditetapkan sebagai hasil akhir dalam model prediksi *Random Forest*.

### 3.6 Evaluasi Model

Evaluasi model pada penelitian ini menggunakan kombinasi metode *confusion matrix*, dan model validasi silang jenis *K-Fold Cross-Validation*, berikut penjelasannya:

#### 3.6.1 *Confusion Matrix*

*Confusion matrix* umumnya digunakan untuk evaluasi model klasifikasi pada dataset dan akan ditampilkan dalam matrik 2x2, dimana berisi nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Berikut contoh dari *confusion matrix*:

Tabel 3.14 Confusion matrix

| <div> <div>Nilai<br/>Prediksi</div> <div>Nilai<br/>Aktual</div> </div> | Positif | Negatif |
|------------------------------------------------------------------------|---------|---------|
| Positif                                                                | TP      | FP      |
| Negatif                                                                | FN      | TN      |

TP merupakan jumlah data yang sebenarnya berlabel tidak terkena attrition atau label '0' dan berhasil diprediksi dengan benar oleh model sebagai tidak terkena attrition. Sedangkan FP, jumlah data yang sebenarnya terkena attrition atau berlabel '1' namun salah diprediksi oleh model sebagai tidak terkena attrition atau menjadi diprediksi sebagai label '0'. Dan FN adalah jumlah data yang sebenarnya tidak terkena attrition atau berlabel '0' namun salah diprediksi oleh model sebagai terkena attrition atau berlabel '1'. Begitupula TN, jumlah data yang sebenarnya tidak terkena attrition atau berlabel '0' dan berhasil diprediksi dengan benar oleh model sebagai tidak terkena attrition. Nilai yang dikeluarkan oleh confusion matrix

akan digunakan sebagai acuan untuk perhitungan evaluasi model seperti akurasi, presisi, *recall* dan *F1-Score* (Rainio et al., 2024). Misalkan data *confution matrix* berikut ini:

Tabel 3.15 Implementasi confusion matrix

| <div> <div>Nilai<br/>Prediksi</div> <div>Nilai<br/>Aktual</div> </div> | Attrition(1) | No Attrition(0) |
|------------------------------------------------------------------------|--------------|-----------------|
| Attrition(1)                                                           | 62           | 22              |
| No Attrition(0)                                                        | 14           | 50              |

Berdasarkan tabel tersebut maka evaluasi matrix dapat dihitung sebagai berikut:

1. Akurasi adalah model evaluasi untuk mengukur performa model dari seberapa sering model bisa memprediksi dengan tepat. Berikut contoh perhitungannya:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

$$Accuracy = \frac{62 + 50}{62 + 22 + 14 + 50}$$

$$Accuracy = \frac{112}{148} = 0,75 = 75\%$$

2. Presisi digunakan untuk mengukur seberapa sering model dapat memprediksi kelas positif dengan benar, di antara seluruh prediksi positif yang model buat. Berikut contoh perhitungannya:

$$Precision = \frac{TP}{TP + FP}$$

$$Precision = \frac{62}{62 + 14}$$

$$Precision = \frac{62}{76} = 0,81 = 81\%$$

3. *Recall* digunakan untuk mengukur seberapa jauh model dapat mengenali dan mengklasifikasikan kelas positif dengan benar. Berikut contoh perhitungannya:

$$Recall = \frac{TP}{TP + FN}$$

$$Recall = \frac{62}{62 + 22}$$

$$Recall = \frac{62}{84} = 0,74 = 74\%$$

4. *F1-Score* digunakan untuk mengukur seberapa baik model dalam mengklasifikasikan kelas positif maupun negative dengan tepat. *F1-Score* juga digunakan untuk alat ukur keseimbangan data. Berikut rumus dan contoh perhitungan *F1-Score*:

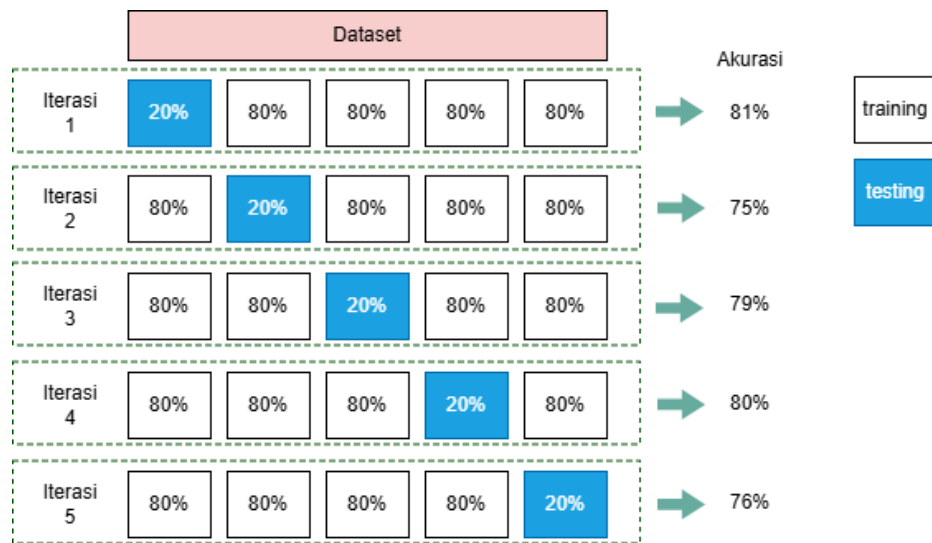
$$F1 - Score = 2 \times \frac{Recall \times Precisson}{Recall + Precisson}$$

$$F1 - Score = 2 \times \frac{0,74 \times 0,81}{0,74 + 0,81}$$

$$F1 - Score = 2 \times \frac{0,599}{1,55} = 0,77 \approx 77\%$$

### 3.6.2 Validasi Silang

Selain dalam mengevaluasi model dengan teknik pembagian train-test-split, model RF juga akan dievaluasi dengan menggunakan teknik pembagian validasi silang jenis *K-Fold*. Gambar 2.3 menjelaskan contoh implementasi *K-Fold Cross-Validation* dengan nilai *k* atau lipatan berjumlah 5, dan ratio permbagian data latih dan data uji adalah 80:20.



Gambar 3.8 Implementasi K-Fold CV

Pada Gambar 2.3 dijelaskan data latih akan menempati posisi lipatan data yang lain bergantian secara merata, sehingga proses perhitungan evaluasi matrik dengan validasi silang sebagai berikut:

$$Akurasi = \frac{81\% + 75\% + 79\% + 80\% + 76\%}{5}$$

$$Akurasi = 78,2\%$$

### 3.7 Skenario Pengujian

Skenario pengujian yang akan dilakukan dalam penelitian ini adalah dengan mengkombinasikan seleksi fitur dengan *balancing* data, dengan seluruh model akan menggunakan data latih sebanyak 80% dan data uji sebanyak 20%, dengan menggunakan *K-Fold* sebanyak 5 *fold*.

Tabel 3.16 Skenario model

| Skenario    | SMOTE | <i>Pearson<br/>Corelation</i> | ANOVA | ANOVA<br>Penyesuaian |
|-------------|-------|-------------------------------|-------|----------------------|
| Skenario 1  | -     | -                             | -     | -                    |
| Skenario 2  | -     | √                             | -     | -                    |
| Skenario 3  | -     | -                             | √     | -                    |
| Skenario 4  | -     | -                             | -     | √                    |
| Skenario 5  | -     | √                             | √     | -                    |
| Skenario 6  | -     | √                             | -     | √                    |
| Skenario 7  | √     | -                             | -     | -                    |
| Skenario 8  | √     | √                             | -     | -                    |
| Skenario 9  | √     | -                             | √     | -                    |
| Skenario 10 | √     | -                             | -     | √                    |
| Skenario 11 | √     | √                             | √     | -                    |
| Skenario 12 | √     | √                             | -     | √                    |

Berikut penjelasan skenario pengujian penelitian ini, sesuai dengan Tabel 3.17:

1. Model *baseline*: model ini terdapat pada skenario 1, yakni model yang tidak berisi pengujian dengan seleksi fitur apapun ataupun dengan *balancing* data (SMOTE).
2. Model seleksi fitur: model ini terdapat pada skenario 2,3,4,5, dan skenario 6, yakni model yang menggunakan pengujian seleksi fitur tanpa *balancing* data, diantaranya:
  - Seleksi fitur menggunakan korelasi
  - Seleksi fitur menggunakan ANOVA
  - Kombinasi seleksi fitur menggunakan korelasi dan ANOVA
- 3 Model *balancing* data: proses pemodelan ini dilakukan dengan pengujian menggunakan SMOTE tanpa seleksi fitur yang terdapat pada model skenario 7.

- 4 Model *balancing* data: proses pemodelan ini dilakukan dengan pengujian menggunakan SMOTE tanpa seleksi fitur yang terdapat pada model skenario 7.
- 5 Model kombinasi: mengkombinasikan pengujian menggunakan seleksi fitur dengan *balancing* data. Model kombinasi ini terdapat pada pengujian skenario 8,9,10,11 dan skenario 12.
- 6 Model juga akan diuji untuk optimasi parameter menggunakan *K-Fold Cross-Validation* , yang dijabarkan dalam tabel berikut:

Tabel 3.17 Tuning Hyperparameter

| <i>Tuning Hyperparameter</i> |    |    |     |     |
|------------------------------|----|----|-----|-----|
| <i>max_dept</i>              | 5  | 10 | 15  | 20  |
| <i>n_estimator</i>           | 50 | 75 | 100 | 150 |

## BAB IV

### HASIL DAN PEMBAHASAN

#### 4.1 Hasil Pengujian

Pada bab ini menyajikan hasil pengujian dari seluruh tahapan pemrosesan data dan pemodelan, meliputi normalisasi data, seleksi fitur berbasis metode *filtering*, yang diantaranya menggunakan *pearson correlation* dan ANOVA, serta hasil pengujian *balancing* data menggunakan SMOTE dan *splitting* data, serta implementasi *Random Forest* berdasarkan skenario pengujian pada BAB III.

Setelah proses transformasi, dari 35 fitur yang ada, berkembang menjadi 51 fitur. Kemudian seluruh data di normalisasi agar memiliki nilai dengan rentang yang sama menggunakan teknik standardisasi berbasis distribusi rata-rata dan simpangan baku atau *z-score normalization*, berikut 51 fitur dan hasil normalisasi:

Tabel 4.1 Fitur Setelah Transformasi

| Fitur                    |                           |                                   |                                |
|--------------------------|---------------------------|-----------------------------------|--------------------------------|
| Age                      | NumCompaniesWorked        | BusinessTravel_Travel_Frequently  | JobRole_Laboratory Technician  |
| Attrition                | PercentSalaryHike         | BusinessTravel_Travel_Rarely      | JobRole_Manager                |
| DailyRate                | PerformanceRating         | Department_Human Resources        | JobRole_Manufacturing Director |
| Distance FromHome        | RelationshipSatisfaction  | Department_Research & Development | JobRole_Research Director      |
| Education                | StockOptionLevel          | Department_Sales                  | JobRole_Research Scientist     |
| Environment Satisfaction | TotalWorkingYears         | EducationField_Human Resources    | JobRole_Sales Executive        |
| Gender                   | TrainingTimesLastYear     | EducationField_Life Sciences      | JobRole_Sales Representative   |
| HourlyRate               | WorkLifeBalance           | EducationField_Marketing          | MaritalStatus_Divorced         |
| JobInvolvement           | YearsAtCompany            | EducationField_Medical            | MaritalStatus_Married          |
| JobLevel                 | YearsInCurrentRole        | EducationField_Other              | MaritalStatus_Single           |
| JobSatisfaction          | YearsSinceLastPromotion   | EducationField_Technical Degree   | OverTime_No                    |
| MonthlyIncome            | YearsWithCurrManager      | JobRole_Healthcare Representative | OverTime_Yes                   |
| MonthlyRate              | BusinessTravel_Non-Travel | JobRole_Human Resources           |                                |

Tabel 4.2 Hasil Normalisasi

| Age       | Daily Rate | Distance From Home | Education | Environment Satisfaction | Gender    | ... | OverTime_Yes |
|-----------|------------|--------------------|-----------|--------------------------|-----------|-----|--------------|
| 0,44635   | 0,742527   | -1,01091           | -0,89169  | 0,44635                  | 2,280906  | ... | 1,591746     |
| 1,322365  | -1,29777   | -0,14715           | -1,86843  | 1,322365                 | -0,43842  | ... | -0,62824     |
| 0,008343  | 1,414363   | -0,88752           | -0,89169  | 0,008343                 | 2,280906  | ... | 1,591746     |
| -0,42966  | 1,461466   | -0,76412           | 1,061787  | -0,42966                 | -0,43842  | ... | 1,591746     |
| -1,08668  | -0,52429   | -0,88752           | -1,86843  | -1,08668                 | -0,43842  | ... | -0,62824     |
| -0,539166 | 0,50205    | -0,88751           | -0,89169  | 1,16978                  | 0,816496  | ... | -0,62824     |
| 2,417383  | 1,29289    | -0,76412           | 0,08505   | 0,25462                  | -1,224744 | ... | -0,62824     |
| ...       | ...        | ...                | ...       | ...                      | ...       | ... | ...          |
| -0,64867  | -1,18374   | -0,517332          | 0,08505   | -0,660531                | 0,816497  | ... | 1,591746     |
| -1,196177 | 0,90367    | -0,51733           | 0,08505   | 1,169781                 | -1,224745 | ... | -0,62824     |
| -0,10116  | 0,202082   | 1,703764           | -0,89169  | 0,254625                 | 0,816497  | ... | -0,62824     |
| 0,227347  | -0,46975   | -0,39394           | -1,86843  | 1,169781                 | 0,816497  | ... | -0,62824     |
| -1,08668  | -1,60518   | -0,64073           | 0,085049  | -0,66053                 | 0,816497  | ... | 1,591746     |
| 1,322365  | 0,546677   | -0,88752           | 0,085049  | 1,169781                 | 0,816497  | ... | -0,62824     |
| -0,32016  | -0,43257   | -0,14715           | 0,085049  | -0,66053                 | 0,816497  | ... | -0,62824     |

## 4.2 Pearson Correlation

Dalam seleksi fitur berbasis *filtering* menggunakan *Pearson Correlation* fitur yang memiliki korelasi tinggi satu sama lain dengan fitur lainnya akan dipilih salah satu untuk dipertimbangkan masuk ke dalam model *machine learning*. Pemilihan tersebut dilakukan dari nilai koefisien korelasi *pearson* yang berada dalam rentang 0,60 hingga 1,00. Nilai tersebut diambil karena dianggap memiliki hubungan linier yang kuat atau rentang ambang batas ini dipilih karena nilai korelasi  $\geq 0,60$  umumnya mencerminkan asosiasi yang signifikan dan kuat, di mana semakin mendekati 1,00 semakin tinggi tingkat kesamaan informasi yang dibawa oleh kedua fitur tersebut (Schober et al., 2018).

```

# Calculate the correlation matrix
correlation_matrix = df.corr()

# Create a mask to identify the features with a correlation coefficient greater than or equal to 0.75
high_correlation_mask = correlation_matrix >= 0.60

# Identify and list the highly correlated features
highly_correlated_features = []

for feature in high_correlation_mask.columns:
    correlated_with = high_correlation_mask.index[high_correlation_mask[feature]].tolist()
    for correlated_feature in correlated_with:
        if feature != correlated_feature and (correlated_feature, feature) not in highly_correlated_features:
            highly_correlated_features.append((feature, correlated_feature))

# Print the highly correlated features
print("Highly correlated features:")
for feature1, feature2 in highly_correlated_features:
    print(f"{feature1} and {feature2}")

```

Gambar 4.1 Kode untuk melihat korelasi antar fitur

Gambar 4.1 digunakan untuk mengidentifikasi pasangan fitur yang memiliki korelasi tinggi berdasarkan matriks korelasi yang telah dihitung sebelumnya. Proses dimulai memanggil fungsi *corr* yang dimiliki *library* pandas, kemudian membuat *threshold* atau *masking*, yakni berupa batasan dengan nilai *boolean* melalui kode `high_correlation_mask = correlation_matrix >= 0.60`, di mana setiap pasangan fitur dengan nilai korelasi  $\geq 0,60$  akan bernilai *True* dan sisanya *False*. Selanjutnya, dibuat list kosong *highly\_correlated\_features* untuk menampung pasangan fitur yang memenuhi kriteria tersebut. Melalui perulangan pada setiap fitur, kode mengambil fitur lain yang memiliki nilai *True*, kemudian dilakukan pengecekan untuk memastikan bahwa fitur tidak dibandingkan dengan dirinya sendiri serta untuk menghindari duplikasi pasangan misalnya fitur A–B dan B–A. Jika memenuhi kondisi, pasangan fitur tersebut ditambahkan ke dalam list sebagai fitur dengan korelasi tinggi, sehingga menghindari *multicollinearity* dan redundansi yang dapat menurunkan stabilitas serta interpretabilitas model *machine learning*, salah satu fitur dari pasangan yang berkorelasi tinggi tersebut akan

dieliminasi atau dipilih salah satu untuk dipertahankan. Berikut fitur-fitur yang memiliki korelasi tinggi tersebut:

Tabel 4.3 Hasil Korelasi Fitur

| Fitur                                                         | Nilai       | Interpretasi Nilai |
|---------------------------------------------------------------|-------------|--------------------|
| Age and TotalWorkingYears                                     | $\geq 0.60$ | Kuat               |
| JobLevel and MonthlyIncome                                    |             |                    |
| JobLevel and TotalWorkingYears                                |             |                    |
| MonthlyIncome and TotalWorkingYears                           |             |                    |
| MonthlyIncome and JobRole_Manager                             |             |                    |
| PercentSalaryHike and PerformanceRating                       |             |                    |
| TotalWorkingYears and YearsAtCompany                          |             |                    |
| YearsAtCompany and YearsInCurrentRole                         |             |                    |
| YearsAtCompany and YearsSinceLastPromotion                    |             |                    |
| YearsAtCompany and YearsWithCurrManager                       |             |                    |
| YearsInCurrentRole and YearsWithCurrManager                   |             |                    |
| Department_Human Resources and EducationField_Human Resources |             |                    |
| Department_Human Resources and JobRole_Human Resources        |             |                    |
| Department_Sales and JobRole_Sales Executive                  | $\geq 0.80$ | Sangat Kuat        |
| JobLevel and MonthlyIncome                                    |             |                    |
| Department_Human Resources and JobRole_Human Resources        |             |                    |
| Department_Sales and JobRole_Sales Executive                  |             |                    |

Pada tabel 4.3 dijelaskan terkait bahwa fitur-fitur hasil tranformasi seperti hasil dari *encoding* dan *one-hot-encoding* yang memiliki korelasi kuat akan tetap dipertahankan, hal ini dilakukan atas pertimbangan bahwa fitur-fitur tersebut pada dasarnya berasal dari transformasi yang sama terhadap fitur asli, sehingga menyebabkan banyak memiliki informasi yang sama. Fitur-Fitur yang tereliminasi diantaranya, fitur ‘Age’, ‘JobLevel’, ‘PercentSalaryHike’, ‘YearsInCurrentRole’, ‘YearsSinceLastPromotion’, dan ‘YearsWithCurrManager’.

### 4.3 ANOVA

Tahap seleksi fitur selanjutnya yang berbasis *filtering* adalah dengan menerapkan *Analysis of Variance* (ANOVA), khususnya uji ANOVA satu arah yakni *one-way* ANOVA atau ANOVA F-test, untuk mengevaluasi signifikansi fitur terhadap kelas dalam variabel target.

```
import numpy as np
from scipy import stats

f_scores = {}
p_values = {}

# Mendeskripsikan kolom numerik
num_cols = new_datax.columns

for column in num_cols:
    # Memisahkan data berdasarkan kategori 'Attrition'
    group_0 = new_datax[target == 0][column] # Data untuk Attrition = 0
    group_1 = new_datax[target == 1][column] # Data untuk Attrition = 1

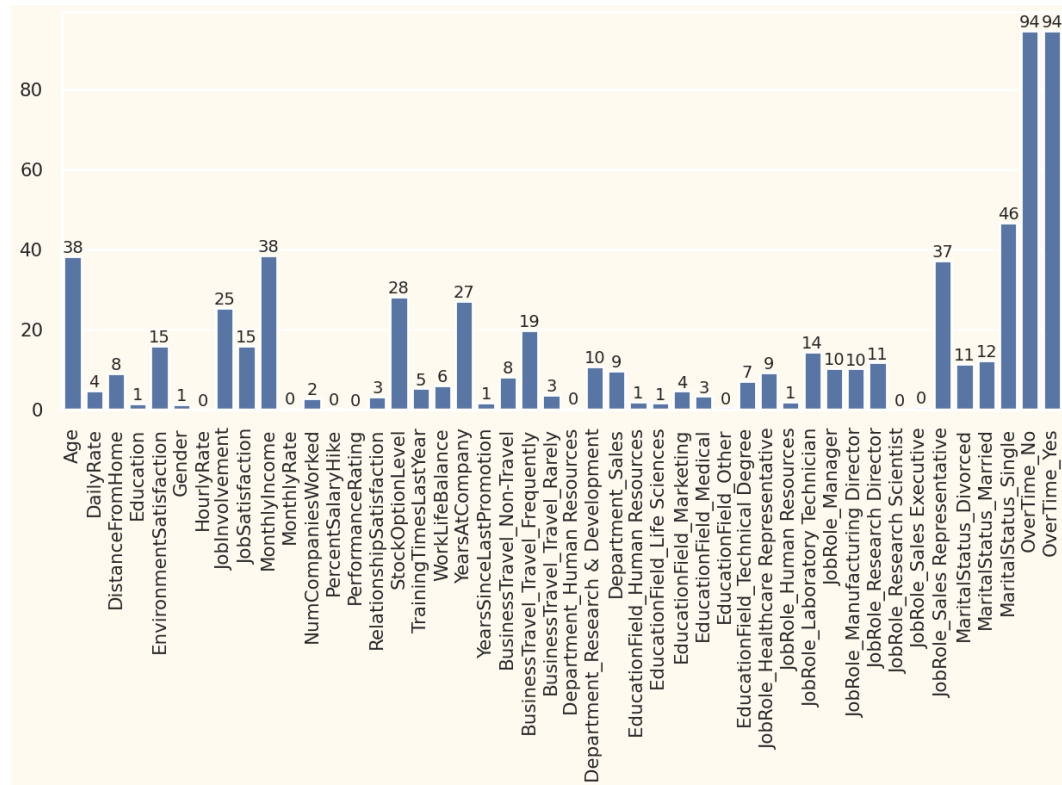
    # Melakukan ANOVA
    f_score, p_value = stats.f_oneway(group_0, group_1)

    # Menyimpan hasilnya
    f_scores[column] = f_score
    p_values[column] = p_value
```

Gambar 4.2 Kode untuk mengidentifikasi hubungan dengan fitur target

Gambar 4.2 mengidentifikasi apakah terdapat perbedaan yang signifikan antara dua kelompok atau grup dalam fitur target yakni grup 0 (No Attrition = 0) dan grup 1 (Attrition = 1). Dalam potongan kode tersebut identifikasi dilakukan dengan memanfaatkan *library* stats untuk memanggil fungsi *oneway*. Kemudian mengidentifikasi fitur-fitur independent dalam *new\_datax*. Selanjutnya, seluruh kolom numerik pada *new\_datax* diiterasi satu per satu, untuk dilakukan pemisahan data menjadi dua grup berdasarkan nilai target pada fitur target, yakni grup 0 atau grup 1. Setelah itu, fungsi *stats.f\_oneway()* digunakan untuk menghitung nilai *F*-

*score* dan *p-value*, yang masing-masing menunjukkan tingkat perbedaan antar grup dan signifikansinya secara statistik. Berikut hasil uji ANOVA:



Gambar 4.3 Hasil Uji Fitur Pada ANOVA

Berdasarkan hasil uji ANOVA, fitur-fitur yang menunjukkan pengaruh signifikan terhadap kelas dalam variable target, ditandai dengan nilai F yang tinggi dan  $p\text{-value} < 0,05$ . Sebaliknya, fitur yang menunjukkan nilai F yang rendah dan  $p\text{-value} > 0,05$  memiliki pengaruh yang kurang signifikan terhadap kelas dalam variable target, sehingga akan di eliminasi untuk mengurangi dimensi data dan potensi noise. Berikut adalah fitur dengan nilai nilai F yang rendah dan  $p\text{-value} > 0,05$ :

Tabel 4.4 Fitur dengan signifikasi diatas 0,05

| No | Features                     | T_Score  | P_value     |
|----|------------------------------|----------|-------------|
| 1  | BusinessTravel_Travel_Rarely | 3,61133  | 0,057582943 |
| 2  | EducationField_Medical       | 3,249813 | 0,071636615 |

| No | Features                       | T_Score  | P_value     |
|----|--------------------------------|----------|-------------|
| 3  | RelationshipSatisfaction       | 3,095576 | 0,07871363  |
| 4  | NumCompaniesWorked             | 2,782287 | 0,095525262 |
| 5  | EducationField_Human Resources | 1,954722 | 0,162290159 |
| 6  | JobRole_Human Resources        | 1,927858 | 0,165203657 |
| 7  | YearsSinceLastPromotion        | 1,602218 | 0,205789959 |
| 8  | EducationField_Life Sciences   | 1,571701 | 0,210159792 |
| 9  | Education                      | 1,446308 | 0,229315203 |
| 10 | Gender                         | 1,274587 | 0,259092364 |
| 11 | JobRole_Sales Executive        | 0,57425  | 0,448696551 |
| 12 | EducationField_Other           | 0,470382 | 0,492919751 |
| 13 | Department_Human Resources     | 0,416027 | 0,519027187 |
| 14 | MonthlyRate                    | 0,337916 | 0,561123598 |
| 15 | PercentSalaryHike              | 0,266728 | 0,605612824 |
| 16 | HourlyRate                     | 0,068796 | 0,793134769 |
| 17 | PerformanceRating              | 0,01225  | 0,911884042 |
| 18 | JobRole_Research Scientist     | 0,00019  | 0,989009929 |

Fitur-fitur tersebut akan dieliminasi dengan 2 skenario uji, yang pertama seluruh fitur yang  $>0,05$ , dan yang kedua adalah fitur-fitur asli dengan signifikansi  $>0,05$  dan mengabaikan fitur hasil transformasi meskipun memiliki signifikansi yang juga  $>0,05$ .

#### 4.4 SMOTE

Penerapan *Synthetic Minority Over-sampling Technique* (SMOTE) bertujuan untuk mengatasi masalah distribusi kelas yang tidak seimbang antara kelas “Yes” dan “No” dalam attrition.

```

import numpy as np
import pandas as pd

# Fungsi untuk menghitung Euclidean distance antara dua titik
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))

X_new = x_scalling

print("Before SMOTE:\n", pd.Series(y).value_counts(), "\n")

# Pisahkan mayoritas dan minoritas
class_counts = pd.Series(y).value_counts()
minority_class = class_counts.idxmin()
majority_class = class_counts.idxmax()

x_minority = X_new[y == minority_class]
n_minority, n_features = x_minority.shape
n_majority = class_counts[majority_class]

# Hitung berapa banyak sampel sintetis yang perlu dibuat
n_synthetic = n_majority - n_minority

```

Gambar 4.4 Identifikasi data mayoritas dan minoritas

Gambar 4.4 menjelaskan identifikasi data minoritas dan data mayoritas untuk keperluan dalam membuat data sintetis minoritas agar data yang dimiliki seimbang antara data pada kelompok attrition dengan kelompok tidak attrition. Pembuatan data sintetis pada SMOTE diawali dengan mencari tetangga terdekatnya dengan data minoritas lainnya, jumlah tetangga terdekat yang diinginkan bisa disesuaikan dan diinisialisasi dengan  $k$  pada gambar 4.5, kemudian perhitungan kedekatan ini menggunakan perhitungan jarak lurus dengan fungsi *euclidean\_distance*.

```

# 1. Mencari k tetangga terdekat secara manual
k = 5
synthetic_samples = []

for _ in range(n_synthetic):
    # Ambil random sample dari minority
    i = np.random.randint(0, n_minority)
    x_i = x_minority[i]

    # Hitung jarak Euclidean dari x_i ke semua titik di x_minority
    distances = []
    for x in x_minority:
        dist = euclidean_distance(x_i, x)
        distances.append(dist)

    # Temukan k tetangga terdekat
    neighbors_indices = np.argsort(distances)[1:k+1] # Skip diri sendiri (indeks pertama)

    # Pilih salah satu tetangga acak
    nn_index = np.random.choice(neighbors_indices)
    x_nn = x_minority[nn_index]

    # 2. Interpolasi untuk membuat data sintetis
    lam = np.random.rand()
    x_new = x_i + lam * (x_nn - x_i)

    # Simpan data sintetis
    synthetic_samples.append(x_new)

# 3. Konversi data sintetis ke dalam DataFrame
x_synthetic = pd.DataFrame(synthetic_samples, columns=[f'Feature{i+1}' for i in range(n_features)])

# 4. Membuat label untuk data sintetis
y_synthetic = pd.Series([minority_class] * len(x_synthetic))

# 5. Gabungkan data sintetis dengan data asli
X = np.vstack([X_new, x_synthetic])
Y = np.hstack([y, y_synthetic])

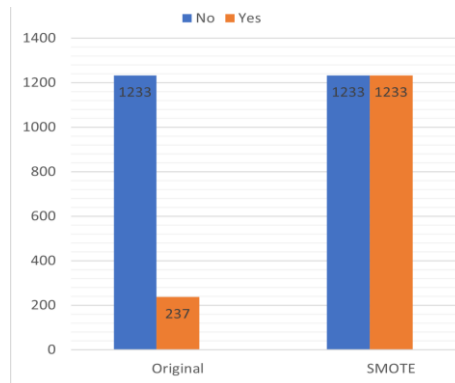
# 6. Menampilkan distribusi kelas setelah SMOTE
print("After SMOTE:\n", pd.Series(Y).value_counts())

```

Gambar 4.5 Membuat data sintetik

Pada Gambar 4.5 menjelaskan kode dalam mengambil data *minority* untuk dijadikan sebagai sample dari pembuatan data sintetik. Kemudian data sample tersebut dicarikan tetangga terdekat dengan fungsi *euclidean\_distance* sejumlah tetangga yang diinginkan yakni  $k=5$ . Dari 5 tetangga tersebut akan dipilih secara acak untuk diinterpolasikan atau dikombinasikan dengan sample data *minority* yang sudah terambil sebelumnya untuk membuat data sintetik baru, dengan kemiripan

0,5 dari masing-masing data. Berikut Gambaran persebaran distribusi data sebelum dan sesudah SMOTE:



Gambar 4.6 Distribusi Data

Pada tabel 4.6 memberikan contoh hasil dari data sintetis minoritas berdasarkan *oversampling* yang dilakukan menggunakan teknik SMOTE , sebagai berikut:

Tabel 4.5 Hasil data sintetik dari SMOTE

| Age      | DailyRate | Distance FromHome | Education | Environment Satisfaction | Gender   | ... | Atrition |
|----------|-----------|-------------------|-----------|--------------------------|----------|-----|----------|
| 0,651843 | -1,14551  | -0,40916          | -0,28154  | -0,88873                 | 0,816497 | ... | 1        |
| -0,95906 | -0,37395  | -1,01091          | 0,369616  | -0,44435                 | -1,22474 | ... | 1        |
| 0,290573 | -0,26027  | -0,42001          | -0,49961  | -0,11273                 | 0,816497 | ... | 1        |
| 0,376267 | -0,35469  | 0,979767          | 0,099446  | -1,12485                 | 0,816497 | ... | 1        |
| -0,7293  | -1,23931  | 1,626331          | -0,89169  | 0,254625                 | 0,816497 | ... | 1        |
| 0,059648 | -0,9537   | 0,080368          | 1,061787  | 1,169781                 | 0,816497 | ... | 1        |
| -0,53917 | 0,730101  | -0,29953          | -0,7999   | 1,169781                 | 0,816497 | ... | 1        |
| -1,55161 | -0,36905  | -0,45092          | -1,86843  | -0,39298                 | 0,816497 | ... | 1        |
| -0,60635 | -0,5503   | -0,25147          | -0,36557  | 0,747571                 | -0,12524 | ... | 1        |
| -1,21781 | -0,97748  | 0,02969           | -0,46864  | -0,53813                 | -0,06762 | ... | 1        |
| -1,62521 | 1,539182  | 0,953287          | -1,80238  | 1,169781                 | -1,08671 | ... | 1        |
| -1,07819 | 0,822419  | -0,86198          | 0,085049  | 0,617514                 | -0,60883 | ... | 1        |

## 4.5 Random Forest

*Random Forest* atau gabungan *decision tree* ini, dibangun dengan memilih fitur dan nilai threshold yang menghasilkan tingkat kemurnian data terbaik pada setiap proses pembentukan keputusan. Tingkat kemurnian tersebut diukur menggunakan metrik *Gini Impurity*, di mana nilai yang lebih rendah menunjukkan pemisahan kelas yang semakin optimal dalam setiap node pohon keputusan.

```
import numpy as np

# Fungsi untuk menghitung Gini Impurity
def gini_impurity(Y):
    class_counts = np.bincount(Y)
    probabilities = class_counts / len(Y)
    return 1 - np.sum(probabilities ** 2)

# Fungsi untuk membagi data berdasarkan threshold
def split_data(X, Y, feature_index, threshold):
    left_mask = X[:, feature_index] <= threshold
    right_mask = ~left_mask
    return X[left_mask], Y[left_mask], X[right_mask], Y[right_mask]
```

Gambar 4.7 Gini Impurity

Gambar 4.7 menjelaskan awal pembentukan RF yang dimulai dari pembuatan pohon keputusan. Setiap pembuatan pohon dimulai dengan pencarian nilai gini terendah atau paling murni dengan perhitungan gini pada masing-masing fitur. Oleh karena itu untuk langkah awal adalah dengan menginisialisasi fungsi gini, dan fungsi *threshold*. *Threshold* sendiri digunakan sebagai batas pemisah data dan juga digunakan sebagai batas pemisah pembentukan cabang kiri dan kanan pada pohon keputusan.

```

# Fungsi untuk memilih fitur dan threshold terbaik berdasarkan Gini Impurity
def best_split(X, Y):
    best_gini = float('inf')
    best_feature = None
    best_threshold = None
    n_features = X.shape[1]

    for feature_index in range(n_features):
        values = np.sort(np.unique(X[:, feature_index]))
        thresholds = (values[:-1] + values[1:]) / 2

        for threshold in thresholds:
            X_left, Y_left, X_right, Y_right = split_data(X, Y, feature_index, threshold)

            # Menghindari pembagian yang menghasilkan bagian kosong
            if len(Y_left) == 0 or len(Y_right) == 0:
                continue

            gini_left = gini_impurity(Y_left)
            gini_right = gini_impurity(Y_right)
            gini = (len(Y_left) / len(Y)) * gini_left + (len(Y_right) / len(Y)) * gini_right

            if gini < best_gini:
                best_gini = gini
                best_feature = feature_index
                best_threshold = threshold

    return best_feature, best_threshold

```

Gambar 4.8 Threshold

Fungsi *best\_split(X, Y)* pada Gambar 4.8 bertugas mencari kombinasi fitur dan *threshold* terbaik dengan cara mencoba semua kemungkinan pembagian atau nilai tengah dari setiap baris data, lalu memilih fitur dan cabang yang menghasilkan nilai gini paling kecil. Jika nilai gini 0 maka perhitungan dicabang tersebut berhenti dan hanya melanjutkan perhitungan di cabang sebelah lainnya.

```

# Fungsi untuk membuat decision tree (max_dept)
def build_tree(X, Y, depth=0, max_depth=None):
    if len(np.unique(Y)) == 1: # Semua data dalam satu kelas
        return {'label': np.unique(Y)[0]}

    if max_depth and depth >= max_depth: # Jika mencapai kedalaman maksimal
        return {'label': np.bincount(Y).argmax()}

    feature, threshold = best_split(X, Y)

    if feature is None: # Tidak ada pembagian yang baik
        return {'label': np.bincount(Y).argmax()}

    left_mask = X[:, feature] <= threshold
    right_mask = ~left_mask
    left_tree = build_tree(X[left_mask], Y[left_mask], depth + 1, max_depth)
    right_tree = build_tree(X[right_mask], Y[right_mask], depth + 1, max_depth)

    return {'feature': feature, 'threshold': threshold, 'left': left_tree, 'right': right_tree}

```

Gambar 4.9 *Decision Tree*

Proses ini menjadi inti dalam pembentukan pohon keputusan, fungsi *build\_tree()* pada Gambar 4.9 membangun struktur *Decision Tree* secara rekursif: jika semua data dalam satu node sudah berasal dari kelas yang sama, atau sudah mencapai batas kedalaman (*max\_depth*), maka node tersebut menjadi daun (leaf) dengan label kelas mayoritas. Jika tidak, data akan terus dibagi menjadi cabang kiri dan kanan berdasarkan hasil best split.

```

# Fungsi untuk membuat prediksi menggunakan decision tree
def predict_tree(tree, X):
    if 'label' in tree:
        return tree['label']

    if X[tree['feature']] <= tree['threshold']:
        return predict_tree(tree['left'], X)
    else:
        return predict_tree(tree['right'], X)

# Fungsi untuk membangun random forest
def random_forest(X, Y, n_trees=10, max_depth=None):
    trees = []
    for _ in range(n_trees):
        # Mengambil bootstrap sample (sampling dengan pengulangan)
        indices = np.random.choice(len(X), size=len(X), replace=True)
        X_sample = X[indices]
        Y_sample = Y[indices]

        # Membangun pohon keputusan
        tree = build_tree(X_sample, Y_sample, max_depth=max_depth)
        trees.append(tree)

    return trees

# Fungsi untuk prediksi dengan random forest
def predict_forest(forest, X):
    predictions = [predict_tree(tree, X) for tree in forest]
    return np.bincount(predictions).argmax() # Prediksi berdasarkan suara terbanyak

```

Gambar 4.10 *Random Forest*

Setelah pohon terbentuk, *fungsi predict\_tree()* digunakan untuk melakukan prediksi terhadap satu data dengan cara menelusuri pohon dari akar hingga ke daun berdasarkan aturan *threshold* yang sudah didefinisikan sebelumnya. Selanjutnya, fungsi *random\_forest()* membangun kumpulan pohon dengan teknik *bootstrap sampling* atau pengambilan sampel data secara acak dengan pengulangan, sehingga setiap pohon dilatih pada data yang sedikit berbeda. Fungsi *predict\_forest()* melakukan prediksi, yakni dengan menggabungkan hasil dari seluruh pohon menggunakan metode *voting* atau suara terbanyak.

#### 4.6 Splitting Data

Seluruh data berjumlah 1470 dengan 1233 berlabel negatif dan 237 data lainnya berlabel positif. Pembagian data latih dan data uji akan dilakukan dalam presentase 80:20, dimana 80% data dari 1470 seluruh data yang ada, akan digunakan sebagai data latih dan 20% lainnya akan digunakan sebagai data uji , berikut pembagiannya:

Tabel 4.6 Splitting data

| Splitting Data   | Data Latih | Label Negatif (Latih) | Label Positif (Latih) | Data Uji | Label Negatif (Uji) | Label Positif (Uji) |
|------------------|------------|-----------------------|-----------------------|----------|---------------------|---------------------|
| Original (80:20) | 1.176      | 978                   | 198                   | 294      | 255                 | 39                  |
| Smote (80:20)    | 1.972      | 983                   | 989                   | 494      | 250                 | 244                 |

```
# Menggunakan fungsi random_forest yang sudah didefinisikan sebelumnya
forest = random_forest(x_train, y_train, n_trees=50, max_depth=5)

# Prediksi dengan random forest pada data latih
y_pred_train = [predict_forest(forest, temp) for temp in x_train]

# Prediksi dengan random forest pada data uji
y_pred = [predict_forest(forest, temp) for temp in x_test]
```

Gambar 4.11 Penggunaan Fungsi Random Forest

Fungsi pada kode-kode yang sebelumnya dibuat, dipanggil untuk mengeksekusi berdasarkan parameter yang diinginkan. Pada Gambar 4.11 menjelaskan variabel *forest* dibuat untuk memanggil fungsi *random\_forest* dengan menggunakan data *training*. Selanjutnya memprediksi model pada data *training* dan *testing*.

#### 4.7 Hasil Uji Coba

Pengujian terdiri dari 12 model skenario pengujian. Model skenario tersebut diuji dengan 2 metode *splitting* data, yakni pertama menggunakan metode *train-test-split* dengan perbandingan pembagian data 80:20, dan kedua menggunakan metode validasi silang dengan lipatan atau *fold* sejumlah 5 atau  $k=5$  untuk keseluruhan data yang berjumlah 1470. Masing-masing akan diuji dengan tuning sebagai berikut:

Tabel 4.7 Optimasi parameter

| <i>Tuning Hyperparameter</i> |    |    |     |     |
|------------------------------|----|----|-----|-----|
| <i>max_dept</i>              | 5  | 10 | 15  | 20  |
| <i>n_estimator</i>           | 50 | 75 | 100 | 150 |

Berikut penjelasan hasil uji coba penelitian ini:

##### 4.7.1 Hasil Uji Skenario 1

Pada skenario pengujian ini, model random forest diuji tanpa menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dan juga tanpa adanya seleksi fitur. Berikut penjelasan hasil uji pada model skenario 1:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.8 Matrix evaluasi train-test-split skenario 1

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 86,73        | 50            | 5,13       | 9,3          | 253 | 2  | 37 | 2  |
| max_depth=10,<br>n_estimators=75  | 86,39        | 42,86         | 7,69       | 13,04        | 251 | 4  | 36 | 3  |
| max_depth=15,<br>n_estimators=100 | 87,76        | 80            | 10,26      | 18,18        | 254 | 1  | 35 | 4  |
| max_depth=20,<br>n_estimators=150 | 87,76        | 80            | 10,26      | 18,18        | 254 | 1  | 35 | 4  |

Performa terbaik pada model RF diperoleh pada konfigurasi *max\_depth=15* dan *n\_estimators=100*, dengan akurasi mencapai 87,76%. Meskipun akurasi menunjukkan hasil yang relatif tinggi, indikator lainnya, seperti presisi yang mencapai 80%, mencerminkan peningkatan dalam kemampuan model untuk memprediksi kelas positif secara lebih tepat dibandingkan pengujian menggunakan parameter lainnya. Namun, meskipun presisi meningkat, nilai *recall* yang masih rendah sekitar 10,26% dan F-1 score sebesar 18,18% mengindikasikan bahwa model cenderung gagal dalam mendeteksi sebagian besar contoh dari kelas positif. Hal ini menunjukkan adanya ketidakseimbangan performa antara kelas positif dan negatif, yang dapat mengarah pada model yang kurang sensitif terhadap kelas minoritas.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.9 Matrix evaluasi k-fold skenario 1

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 85,10             | 65,17              | 9,12            | 15,92             |
| max_depth=10,<br>n_estimators=75  | 86,39             | 85,58              | 18,37           | 29,87             |
| max_depth=15,<br>n_estimators=100 | 86,05             | 77,03              | 18,19           | 29,26             |
| max_depth=20,<br>n_estimators=150 | 86,19             | 80,79              | 18,44           | 29,73             |

Skenario menggunakan evaluasi *k-fold* dengan *hyperparameter* secara keseluruhan menunjukkan *hyperparameter* *max\_depth=10* dan *n\_estimators=75* pada model RF memberikan performa terbaik dengan akurasi 86,39%, precision

85,58%, *recall* 18,37%, dan F1-score 29,87%, menunjukkan kemampuan yang baik dalam memprediksi kelas positif meskipun *recall* masih rendah, sehingga masih dibutuhkan peningkatan lebih lanjut dapat dicapai dengan penanganan ketidakseimbangan data atau seleksi fitur.

#### 4.7.2 Hasil Uji Skenario 2

Pada skenario model ini model random forest diuji menggunakan seleksi fitur *pearson correlation* tanpa menggunakan penanganan ketidakseimbangan data atau tanpa SMOTE. Berikut penjelasan hasil uji pada model skenario 2:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.10 Matrix evaluasi train-test-split skenario 2

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 87,07        | 66,67         | 5,13       | 9,52         | 254 | 1  | 37 | 2  |
| max_depth=10,<br>n_estimators=75  | 87,41        | 66,67         | 10,26      | 17,78        | 253 | 2  | 35 | 4  |
| max_depth=15,<br>n_estimators=100 | 87,07        | 60            | 7,69       | 13,64        | 253 | 2  | 36 | 3  |
| max_depth=20,<br>n_estimators=150 | 87,76        | 71,43         | 12,82      | 21,74        | 253 | 2  | 34 | 5  |

Akurasi tertinggi tercatat pada *hyperparameter* *max\_depth*=20 dan *n\_estimators*=150 sebesar 87,76%. Meskipun akurasi yang dicapai cukup baik, performa dalam hal presisi dan *recall* menunjukkan ketidakseimbangan masih sama seperti skenario sebelumnya. Presisi sebesar 71,43%, namun *recall* tetap rendah, yaitu 12,82%. Hasil ini mencerminkan bahwa meskipun model cukup akurat dalam memprediksi kelas negatif, deteksi terhadap kelas positif tetap terbatas, yang tercermin dalam rendahnya nilai F1-score, yaitu 21,74%. Hal ini mengindikasikan

bahwa model ini masih menghadapi tantangan dalam menangani ketidakseimbangan kelas, meskipun akurasi secara keseluruhan stabil di atas 87%.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.11 Matrix evaluasi k-fold skenario 2

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 84,90             | 83,14              | 7,82            | 14,15             |
| max_depth=10,<br>n_estimators=75  | 86,26             | 84,86              | 17,28           | 28,09             |
| max_depth=15,<br>n_estimators=100 | 86,19             | 81,23              | 17,81           | 28,71             |
| max_depth=20,<br>n_estimators=150 | 85,85             | 78,83              | 15,90           | 25,66             |

Pada uji validasi silang skenario 2 ini, memperlihatkan performa yang lebih baik dan cukup stabil terutama dalam hal presisi. Akurasi tertinggi tercatat pada *hyperparameter* dengan *max\_depth*=10 dan *n\_estimators*=75 dengan nilai 86,26%, diikuti oleh peningkatan presisi yang cukup signifikan sebesar 84,86%, dan *recall* sebesar 17,28%. Nilai F1-score juga meningkat menjadi 28,09%, menunjukkan keseimbangan yang lebih baik antara presisi dan *recall*, meskipun masih rendah dengan perbandingan yang sedikit jika dibandingkan *max\_depth*=15 dan *n\_estimators*=100.

### 4.7.3 Hasil Uji Skenario 3

Pada skenario pengujian ini, model random forest diuji menggunakan seleksi fitur berbasis *filtering* ANOVA tanpa menggunakan penanganan

ketidakseimbangan data menggunakan SMOTE. Berikut penjelasan hasil uji pada model skenario 3:

1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.12 Matrix evaluasi train-test-split skenario 3

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 87,07        | 60            | 7,69       | 13,64        | 253 | 2  | 36 | 3  |
| max_depth=10,<br>n_estimators=75  | 86,73        | 50            | 12,82      | 20,41        | 250 | 5  | 34 | 5  |
| max_depth=15,<br>n_estimators=100 | 86,73        | 50            | 10,26      | 17,02        | 251 | 4  | 35 | 4  |
| max_depth=20,<br>n_estimators=150 | 87,07        | 55,56         | 12,82      | 20,83        | 251 | 4  | 34 | 5  |

Pada uji skenario 3 menunjukkan peningkatan *recall* pada *hyperparameter* dengan *max\_depth*=20 dan *n\_estimators*=150 sebesar 12,82% dengan presisi 55,56 dan F-1 score 20,83%, meskipun akurasi 87,07% tetap sedikit lebih rendah dari Skenario 2. Seleksi fitur dengan ANOVA membantu menyeimbangkan prediksi, namun hasil *recall* dan F-1 score menunjukkan bahwa model masih belum sepenuhnya efektif dalam mendeteksi kelas positif.

2. Validasi silang menggunakan *K-Fold*

Tabel 4.13 Matrix evaluasi k-fold skenario 3

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 85,44             | 75,64              | 12,37           | 21,04             |
| max_depth=10,<br>n_estimators=75  | 86,19             | 71,96              | 22,13           | 33,54             |
| max_depth=15,<br>n_estimators=100 | 86,12             | 70,70              | 21,68           | 33,03             |
| max_depth=20,<br>n_estimators=150 | 86,46             | 79,10              | 21,12           | 32,82             |

Pada pengujian validasi silang hasil terbaik ditunjukkan pada model dengan  $max\_depth=20$  dan  $n\_estimators=150$ , dengan akurasi mencapai 86,46%. Sedangkan presisi dan *recall* masing-masing mencapai 79,10% dan 21,12% , dengan F1-score sebesar 32,82%. Meskipun ada sedikit peningkatan dibandingkan dengan Skenario 2, model ini menunjukkan bahwa seleksi fitur dengan ANOVA memberikan keseimbangan yang lebih baik dalam mendeteksi kelas positif, meskipun masih belum cukup optimal dalam hal *recall* dan F1-score.

#### 4.7.4 Hasil Uji Skenario 4

Pada skenario pengujian ini, model random forest diuji menggunakan seleksi fitur berbasisi filtering ANOVA dengan penyesuaian, yakni hanya mengeliminasi fitur-fitur asli, tanpa menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dan juga tanpa adanya seleksi fitur. Berikut penjelasan hasil uji pada model skenario 4:

##### 1. Validasi menggunakan *train-test-split*

Tabel 4.14 Matrix evaluasi train-test-split skenario 4

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 86,73        | 50            | 5,13       | 9,3          | 253 | 2  | 37 | 2  |
| max_depth=10,<br>n_estimators=75  | 87,07        | 57,14         | 10,26      | 17,39        | 252 | 3  | 35 | 4  |
| max_depth=15,<br>n_estimators=100 | 85,71        | 36,36         | 10,26      | 16           | 248 | 7  | 35 | 4  |
| max_depth=20,<br>n_estimators=150 | 87,07        | 55,56         | 12,82      | 20,83        | 251 | 4  | 34 | 5  |

Pada uji skenario 4 menunjukan hasil kurang lebih hampir sama seperti pada hasil uji coba skenario 3 namun performa cenderung sedikit menurun, belum ada perubahan yang signifikan terutama pada uji *tuning hyoerparameter* terbaik dengan

$max\_depth=20$  dan  $n\_estimators=150$ , menunjukkan presisi sebesar 55,56% dan  $recall$  12,82%. F-1 score juga meningkat menjadi 0,208, meskipun  $recall$  masih rendah. Penyesuaian ulang pada ANOVA memperbaiki seleksi fitur, tetapi model ini masih terbatas dalam mendeteksi kelas positif.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.15 Matrix evaluasi k-fold skenario 4

| Model                                    | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|------------------------------------------|-------------------|--------------------|-----------------|-------------------|
| $max\_depth=5$ ,<br>$n\_estimators=50$   | 85,44             | 84,00              | 11,81           | 20,43             |
| $max\_depth=10$ ,<br>$n\_estimators=75$  | 86,12             | 76,40              | 19,36           | 30,72             |
| $max\_depth=15$ ,<br>$n\_estimators=100$ | 85,99             | 72,24              | 20,51           | 31,78             |
| $max\_depth=20$ ,<br>$n\_estimators=200$ | 86,26             | 76,29              | 21,37           | 33,07             |

Pengujian seleksi fitur dengan ANOVA yang diikuti dengan penyesuaian ulang menghasilkan akurasi tertinggi pada model  $max\_depth=20$  dan  $n\_estimators=200$  dengan nilai 86,26%. Presisi 76,29% dan  $recall$  21,37% menunjukkan sedikit perbaikan yang lebih baik dalam mendeteksi kelas positif dibandingkan dengan seleksi fitur ANOVA tanpa penyesuaian ulang pada skenario 3. Dan rata-rata nilai F1-score yang tercatat sebesar 33,07% dengan performa yang masih sama dengan skenario sebelumnya.

### 4.7.5 Hasil Uji Skenario 5

Pada skenario pengujian ini, model random forest diuji menggunakan seleksi fitur berbasis *filtering pearson correlation* dan ANOVA, tanpa

menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dan juga tanpa adanya seleksi fitur. Berikut penjelasan hasil uji pada model skenario 5:

1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.16 Matrix evaluasi train-test-split skenario 5

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 87,07        | 66,67         | 5,13       | 9,52         | 254 | 1  | 37 | 2  |
| max_depth=10,<br>n_estimators=75  | 87,41        | 62,5          | 12,82      | 21,28        | 252 | 3  | 34 | 5  |
| max_depth=15,<br>n_estimators=100 | 87,76        | 71,43         | 12,82      | 21,74        | 253 | 2  | 34 | 5  |
| max_depth=20,<br>n_estimators=150 | 87,41        | 62,5          | 12,82      | 21,28        | 252 | 3  | 34 | 5  |

Gabungan seleksi fitur *Pearson Corr* dan ANOVA menghasilkan akurasi tertinggi 87,76% pada *max\_depth*=15 dan *n\_estimators*=100, dengan presisi yang sangat baik 71,43% dan *recall* 12,82%. Meskipun F-1 score lebih tinggi dari skenario sebelumnya yakni 21,74%, namun *recall* tetap rendah. Gabungan kedua teknik seleksi fitur membantu model lebih baik dalam menyeimbangkan prediksi antara kelas positif dan negatif.

2. Validasi silang menggunakan *K-Fold*

Tabel 4.17 Matrix evaluasi k-fold skenario 5

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 85,65             | 88,82              | 12,26           | 21,33             |
| max_depth=10,<br>n_estimators=75  | 86,33             | 72,34              | 21,89           | 33,30             |
| max_depth=15,<br>n_estimators=100 | 86,12             | 75,92              | 19,65           | 30,91             |
| max_depth=20,<br>n_estimators=200 | 86,39             | 75,20              | 22,01           | 33,64             |

Pada evaluasi skenario ini memberikan akurasi terbaik pada model dengan *hyperparameter* *max\_depth*=10 dan *n\_estimators*=100, dengan nilai akurasi sebesar 86,39%. Presisi 75,20% dan *recall* 22,01% memberikan keseimbangan yang lebih baik dalam mendeteksi kelas positif, dengan F1-score mencapai 33,64%. Kombinasi ini meningkatkan performa model dalam mendeteksi kelas minoritas, meskipun *recall* dan F1-score masih menunjukkan ruang untuk perbaikan.

#### 4.7.6 Hasil Uji Skenario 6

Pada skenario pengujian ini, model random forest diuji menggunakan seleksi fitur berbasis *filtering pearson correlation* dan ANOVA dengan penyesuaian, yakni hanya mengeliminasi fitur-fitur asli, tanpa menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dan juga tanpa adanya seleksi fitur. Berikut penjelasan hasil uji pada model skenario 6:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.18 Matrix evaluasi train-test-split skenario 6

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|----|
| max_depth=5,<br>n_estimators=50   | 87,07        | 66,67         | 5,13       | 9,52         | 254 | 1  | 37 | 2  |
| max_depth=10,<br>n_estimators=75  | 86,39        | 44,44         | 10,26      | 16,67        | 250 | 5  | 35 | 4  |
| max_depth=15,<br>n_estimators=100 | 86,73        | 50            | 12,82      | 20,41        | 250 | 5  | 34 | 5  |
| max_depth=20,<br>n_estimators=150 | 86,73        | 50            | 12,82      | 20,41        | 250 | 5  | 34 | 5  |

Percobaan dengan gabungan ANOVA penyesuaian ini menunjukkan adanya sedikit penurunan, dan memiliki performa yang tidak jauh beda dengan skenario 5. Pada *hyperparameter* dengan *max\_depth*=20 dan *n\_estimators*=150, menunjukkan nilai akurasi sebesar 86,73%, presisi 50%, cenderung lebih kecil jika

dibanding *hyperparameter* lainnya. Namun *recall* F-1 score tidak lebih rendah, gabungan kedua teknik seleksi fitur dengan penyesuaian ulang tidak memberikan peningkatan signifikan dalam performa model.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.19 Matrix evaluasi k-fold skenario 6

| Model                          | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|--------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5, n_estimators=50   | 85,31             | 82,11              | 10,90           | 18,97             |
| max_depth=10, n_estimators=75  | 86,39             | 76,52              | 21,73           | 33,42             |
| max_depth=15, n_estimators=100 | 86,33             | 74,44              | 21,52           | 32,86             |
| max_depth=20, n_estimators=200 | 86,19             | 72,13              | 21,94           | 33,33             |

Gabungan seleksi fitur *Pearson correlation* dan ANOVA dengan penyesuaian ulang memberikan hasil terbaik pada model dengan *hyperparameter* max\_depth=10 dan n\_estimators=150, dengan akurasi mencapai 86,53%, preisi 74,44% dan *recall* 21,52% dengan F1-score sebesar 32,86%. Meskipun ada peningkatan dalam deteksi kelas positif dibandingkan dengan seleksi fitur *Pearson correlation* saja, performa model masih dapat ditingkatkan lebih lanjut.

### 4.7.7 Hasil Uji Skenario 7

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data atau menggunakan SMOTE, tanpa adanya seleksi fitur. Berikut penjelasan hasil uji pada model skenario 7:

## 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.20 Matrix evaluasi train-test-split skenario 7

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| max_depth=5,<br>n_estimators=50   | 87,45        | 88,56         | 85,66      | 87,08        | 223 | 27 | 35 | 209 |
| max_depth=10,<br>n_estimators=75  | 92,51        | 94,04         | 90,57      | 92,28        | 236 | 14 | 23 | 221 |
| max_depth=15,<br>n_estimators=100 | 94,53        | 95,78         | 93,03      | 94,39        | 240 | 10 | 17 | 227 |
| max_depth=20,<br>n_estimators=150 | 94,33        | 95,38         | 93,03      | 94,19        | 239 | 11 | 17 | 227 |

Penanganan ketidakseimbangan data menggunakan SMOTE memberikan hasil terbaik dengan *max\_depth*=15 dan *n\_estimators*=100, mencapai akurasi tertinggi sebesar 94,53%, presisi 95,78%, dan *recall* yang sangat baik sebesar 93,03%. Serta F-1 score yang tinggi yakni 94,93%, mengindikasikan bahwa penanganan ketidakseimbangan data secara efektif meningkatkan kemampuan model dalam mendeteksi kelas positif, menjadikannya skenario terbaik dalam hal keseimbangan performa dibanding dengan 6 skenario sebelumnya.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.21 Matrix evaluasi k-fold skenario 7

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 86,50             | 89,01              | 83,44           | 86,10             |
| max_depth=10,<br>n_estimators=75  | 93,15             | 94,96              | 91,15           | 93,01             |
| max_depth=15,<br>n_estimators=100 | 93,71             | 95,63              | 91,64           | 93,59             |
| max_depth=20,<br>n_estimators=200 | 93,92             | 95,98              | 91,74           | 93,81             |

Penanganan ketidakseimbangan data memberikan hasil terbaik pada model pengujian *tuning* dengan *max\_depth*=15 dan *n\_estimators*=100, dengan akurasi yang sangat tinggi yakni sebesar 93,71% dan presisi 95,63% serta *recall* 91,64% yang sangat baik. F1-score yang tinggi sebesar 93,59% menunjukkan keseimbangan yang sangat baik antara presisi dan *recall*. Skenario ini memberikan performa terbaik dalam mendeteksi kelas positif dan minoritas, menunjukkan bahwa penanganan ketidakseimbangan data adalah faktor kunci dalam meningkatkan kinerja model.

#### 4.7.8 Hasil Uji Skenario 8

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dengan kombinasi pada skenario 2, yakni menggunakan seleksi fitur berbasis *filtering pearson correlation*. Berikut penjelasan hasil uji pada model skenario 8:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.22 Matrix evaluasi train-test-split skenario 8

| Model                                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|---------------------------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| <i>max_depth</i> =5,<br><i>n_estimators</i> =50   | 87,85        | 88,98         | 86,07      | 87,5         | 224 | 26 | 34 | 210 |
| <i>max_depth</i> =10,<br><i>n_estimators</i> =75  | 92,91        | 94,47         | 90,98      | 92,69        | 237 | 13 | 22 | 222 |
| <i>max_depth</i> =15,<br><i>n_estimators</i> =100 | 93,52        | 95,69         | 90,98      | 93,28        | 240 | 10 | 22 | 222 |
| <i>max_depth</i> =20,<br><i>n_estimators</i> =150 | 94,53        | 96,57         | 92,21      | 94,34        | 242 | 8  | 19 | 225 |

Pada skenario uji ini menghasilkan akurasi tertinggi sebesar 94,53% pada *max\_depth*=20 dan *n\_estimators*=150, dengan presisi 96,57% dan *recall* 92,21%. F-1 score yang mencapai 94,34% mencerminkan keseimbangan optimal antara

precision dan *recall*, menunjukkan bahwa kombinasi penanganan ketidakseimbangan data dan seleksi fitur ini memberikan performa terbaik.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.23 Matrix evaluasi k-fold skenario 8

| Model                          | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|--------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5, n_estimators=50   | 87,27             | 89,57              | 84,43           | 86,88             |
| max_depth=10, n_estimators=75  | 93,39             | 95,46              | 91,19           | 93,27             |
| max_depth=15, n_estimators=100 | 93,71             | 95,85              | 91,43           | 93,58             |
| max_depth=20, n_estimators=200 | 93,80             | 96,34              | 91,09           | 93,64             |

Hasil terbaik pada model dengan percobaan *tuning max\_depth=15* dan *n\_estimators=100*, dengan akurasi 93,71%. Presisi sebesar 95,85% dan *recall* 91,43% terus menunjukkan performa yang sangat baik, dengan F1-score 93,58%. Hal ini mengindikasikan bahwa kombinasi antara penanganan ketidakseimbangan data dan seleksi fitur memberikan peningkatan yang signifikan dalam mendeteksi kelas positif.

### 4.7.9 Hasil Uji Skenario 9

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dengan kombinasi pada skenario 3, yakni menggunakan seleksi fitur berbasis *filtering* ANOVA. Berikut penjelasan hasil uji pada model skenario 9:

## 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.24 Matrix evaluasi train-test-split skenario 9

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| max_depth=5,<br>n_estimators=50   | 84,82        | 85,36         | 83,61      | 84,47        | 215 | 35 | 40 | 204 |
| max_depth=10,<br>n_estimators=75  | 91,09        | 92,37         | 89,34      | 90,83        | 232 | 18 | 26 | 218 |
| max_depth=15,<br>n_estimators=100 | 92,11        | 92,89         | 90,98      | 91,93        | 233 | 17 | 22 | 222 |
| max_depth=20,<br>n_estimators=150 | 91,9         | 92,15         | 91,39      | 91,77        | 231 | 19 | 21 | 223 |

Pada skenario ini menghasilkan akurasi 92,11% dengan presisi 92,89% dan *recall* 90,98% pada *max\_depth*=15 dan *n\_estimators*=100. F-1 score yang mencapai 91,93% mengindikasikan keseimbangan yang baik meskipun tidak sebaik pada Skenario 8. Kombinasi penanganan ketidakseimbangan data dan seleksi fitur ANOVA juga memberikan peningkatan signifikan dalam deteksi kelas positif.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4 25 Matrix evaluasi k-fold skenario 9

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 86,38             | 87,29              | 85,20           | 86,18             |
| max_depth=10,<br>n_estimators=75  | 92,09             | 93,30              | 90,76           | 92,00             |
| max_depth=15,<br>n_estimators=100 | 93,15             | 94,03              | 92,23           | 93,11             |
| max_depth=20,<br>n_estimators=200 | 93,51             | 94,93              | 91,97           | 93.42             |

Hasil terbaik ditunjukkan pada model percobaan *tuning max\_depth*=15 dan *n\_estimators*=100, dengan akurasi 93,15%. Precision 94,03% dan *recall* 92,23%

menunjukkan hasil yang sangat baik dalam mendeteksi kelas positif, dengan F1-score 93,11%. Gabungan ini menunjukkan bahwa penanganan ketidakseimbangan data bersama dengan seleksi fitur ANOVA memberikan keseimbangan yang sangat baik dalam deteksi kelas positif dan minoritas.

#### 4.7.10 Hasil Uji Skenario 10

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dengan kombinasi pada skenario 4, yakni menggunakan seleksi fitur berbasis *filtering* ANOVA dengan penyesuaian. Berikut penjelasan hasil uji pada model skenario 10:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.26 Matrix evaluasi train-test-split skenario 10

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| max_depth=5,<br>n_estimators=50   | 84,01        | 84,81         | 82,38      | 83,58        | 214 | 36 | 43 | 201 |
| max_depth=10,<br>n_estimators=75  | 91,5         | 93,16         | 89,34      | 91,21        | 234 | 16 | 26 | 218 |
| max_depth=15,<br>n_estimators=100 | 92,31        | 94,02         | 90,16      | 92,05        | 236 | 14 | 24 | 220 |
| max_depth=20,<br>n_estimators=150 | 93,12        | 95,26         | 90,57      | 92,86        | 239 | 11 | 23 | 221 |

Pada uji skenario ini dengan kombinasi penanganan ketidak seimbangan data dengan SMOTE dan seleksi fitur ANOVA dengan penyesuaian ulang, menunjukkan akurasi tertinggi sebedar 93,12% pada *max\_depth*=20 dan *n\_estimators*=150, dengan presisi 95,26% dan *recall* (90,57%. Serta F-1 score yang mencapai 92,86% menunjukkan keseimbangan yang sangat baik antara presisi dan *recall*, menjadikan kombinasi ini sangat efektif dalam menangani ketidakseimbangan data dan seleksi fitur.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.27 Matrix evaluasi k-fold skenario 10

| <b>Model</b>                      | <b>Mean Accuracy (%)</b> | <b>Mean Precision (%)</b> | <b>Mean Recall (%)</b> | <b>Mean F1-Score (%)</b> |
|-----------------------------------|--------------------------|---------------------------|------------------------|--------------------------|
| max_depth=5,<br>n_estimators=50   | 85,32                    | 86,91                     | 83,29                  | 85,03                    |
| max_depth=10,<br>n_estimators=75  | 92,74                    | 94,19                     | 91,19                  | 92,66                    |
| max_depth=15,<br>n_estimators=100 | 93,51                    | 95,08                     | 91,84                  | 93,42                    |
| max_depth=20,<br>n_estimators=200 | 93,23                    | 95,06                     | 91,26                  | 93,12                    |

Gabungan penanganan ketidakseimbangan data dengan seleksi fitur ANOVA dan penyesuaian ulang menghasilkan akurasi tertinggi pada model *max\_depth*=10 dan *n\_estimators*=75, dengan akurasi 92,74%. Presisi 94,19% dan *recall* 91,19% menunjukkan keseimbangan yang baik dalam mendeteksi kelas positif, dengan F1-score mencapai 92,66%. Hal ini mengindikasikan bahwa penyesuaian ulang setelah seleksi fitur ANOVA meningkatkan kemampuan model dalam mendeteksi kelas positif dengan lebih akurat.

### 4.7.11 Hasil Uji Skenario 11

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dengan kombinasi pada skenario 5, yakni menggunakan seleksi fitur berbasis *filtering pearson correlation* dan ANOVA. Berikut penjelasan hasil uji pada model skenario 11:

### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.28 Matrix evaluasi train-test-split skenario 11

| Model                             | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|-----------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| max_depth=5,<br>n_estimators=50   | 84,01        | 84,23         | 83,2       | 83,71        | 212 | 38 | 41 | 203 |
| max_depth=10,<br>n_estimators=75  | 90,89        | 91,63         | 89,75      | 90,68        | 230 | 20 | 25 | 219 |
| max_depth=15,<br>n_estimators=100 | 92,91        | 94,09         | 91,39      | 92,72        | 236 | 14 | 21 | 223 |
| max_depth=20,<br>n_estimators=150 | 91,7         | 91,77         | 91,39      | 91,58        | 230 | 20 | 21 | 223 |

Pada skenario 11 memberikan akurasi sebesar 92,91% pada *max\_depth*=15 dan *n\_estimators*=100, dengan presisi 94,91 dan *recall* 94,09. Serta F-1 score yang mencapai 92,72% menunjukkan bahwa kombinasi seleksi fitur dan penanganan ketidakseimbangan data bekerja dengan sangat baik untuk meningkatkan performa model. Tarutama dalam gabuangan seluruh kombinasi penanganan data baik *balancing* dan seleksi.

### 2. Validasi silang menggunakan *K-Fold*

Tabel 4.29 Matrix evaluasi k-fold skenario 11

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 85,36             | 86,96              | 83,23           | 85,01             |
| max_depth=10,<br>n_estimators=75  | 91,28             | 92,19              | 90,26           | 91,20             |
| max_depth=15,<br>n_estimators=100 | 92,78             | 93,86              | 91,57           | 92,69             |
| max_depth=20,<br>n_estimators=200 | 93,27             | 94,09              | 92,38           | 93,22             |

Hasil terbaik pada model  $max\_depth=20$  dan  $n\_estimators=150$ , dengan akurasi 93,27%. Presisi 94,09% dan  $recall$  92,38% menunjukkan performa yang sangat baik dalam mendeteksi kelas positif, dengan F1-score 93,22%. Gabungan ini menunjukkan peningkatan signifikan dalam keseimbangan antara precision dan  $recall$ , menjadikannya salah satu skenario terbaik.

#### 4.7.12 Hasil Uji Skenario 12

Pada skenario pengujian ini, model random forest diuji menggunakan penanganan ketidakseimbangan data menggunakan SMOTE, dengan kombinasi pada skenario 6, yakni menggunakan seleksi fitur berbasis *filtering pearson correlation* dan ANOVA dengan penyesuaian. Berikut penjelasan hasil uji pada model skenario 12:

##### 1. Validasi menggunakan *train-test-split* 80:20

Tabel 4.30 Matrix evaluasi train-test-split skenario 12

| Model                                    | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | TN  | FP | FN | TP  |
|------------------------------------------|--------------|---------------|------------|--------------|-----|----|----|-----|
| $max\_depth=5$ ,<br>$n\_estimators=50$   | 89,27        | 94,01         | 83,61      | 88,5         | 237 | 13 | 40 | 204 |
| $max\_depth=10$ ,<br>$n\_estimators=75$  | 92,11        | 95,56         | 88,11      | 91,68        | 240 | 10 | 29 | 215 |
| $max\_depth=15$ ,<br>$n\_estimators=100$ | 92,31        | 95,58         | 88,52      | 91,91        | 240 | 10 | 28 | 216 |
| $max\_depth=20$ ,<br>$n\_estimators=150$ | 92,51        | 96            | 88,52      | 92,11        | 214 | 9  | 28 | 216 |

Pada skenario 12 ini, performa terbaik pada  $max\_depth=20$  dan  $n\_estimators=150$ , dengan akurasi 92,51%, presisi 96%,  $recall$  88,52%, dan F-1 score 92,11%. Gabungan penanganan ketidakseimbangan data, seleksi fitur *Pearson Corr* dan ANOVA dengan penyesuaian ulang menunjukkan keseimbangan

optimal antara precision dan *recall*, namun jika dibandingkan dengan skenario 11, performanya masih lebih baik meskipun masih dalam rata-rata yang hampir sama.

## 2. Validasi silang menggunakan *K-Fold*

Tabel 4.31 Matrix evaluasi k-fold skenario 12

| Model                             | Mean Accuracy (%) | Mean Precision (%) | Mean Recall (%) | Mean F1-Score (%) |
|-----------------------------------|-------------------|--------------------|-----------------|-------------------|
| max_depth=5,<br>n_estimators=50   | 89,54             | 94,11              | 84,31           | 88,93             |
| max_depth=10,<br>n_estimators=75  | 92,58             | 96,66              | 88,23           | 92,25             |
| max_depth=15,<br>n_estimators=100 | 92,66             | 96,57              | 88,47           | 92,34             |
| max_depth=20,<br>n_estimators=200 | 93,07             | 97,09              | 88,80           | 92,75             |

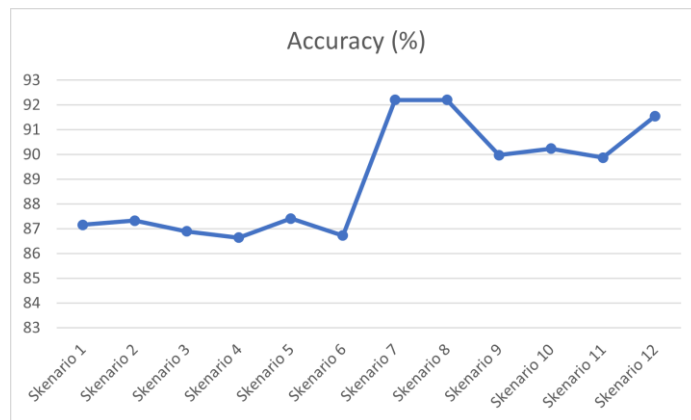
Hasil terbaik pada *hyperparameter* dengan *max\_depth*=15 dan *n\_estimators*=100, dengan akurasi 92,66%. Presisi 96,57% dan *recall* 88,47% menunjukkan deteksi kelas positif yang sangat baik, dengan F1-score 92,34%.

## 4.8 Analisis Hasil Uji Coba

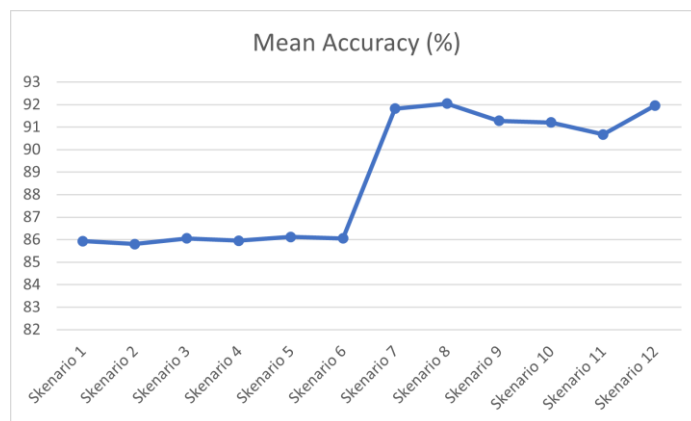
Pada analisis hasil uji coba ini akan dibagi berdasarkan matrik evaluasi yang digunakan dengan merata-rata hasil semua *hyperparameter* yang digunakan dalam uji coba pada setiap skenario, analisis tersebut diantaranya:

### 4.8.1 Analisis Evaluasi Matrik Akurasi

Analisis pada matrix evaluasi akurasi dilakukan pada 2 pengujian pembagian data, yakni diantaranya *train-test-split* dan *k-fold*, berikut penjelasannya:



Gambar 4.12 Rata-rata akurasi pada evaluasi train-test-split



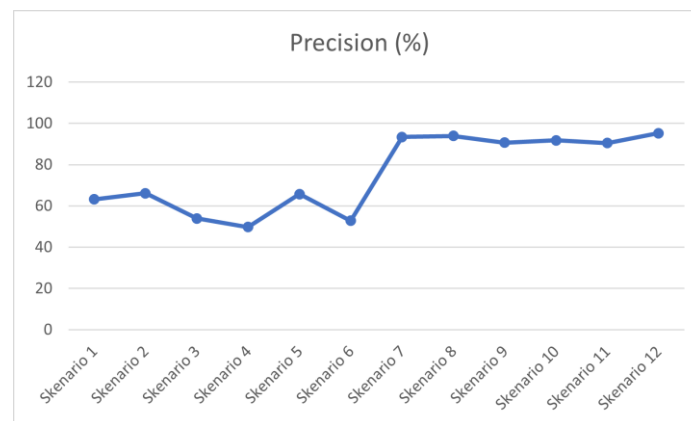
Gambar 4.13 Rata-rata akurasi pada evaluasi k-fold

Berdasarkan analisis akurasi pada seluruh skenario, Skenario 7 menunjukkan perubahan signifikan dibanding dengan skenario lainnya dengan memberikan akurasi tertinggi sebesar 93,71% untuk evaluasi pada *k-fold*. Penanganan ketidakseimbangan data terbukti menjadi faktor kunci dalam meningkatkan akurasi model, terutama dalam mendeteksi kelas minoritas. Skenario lain yang melibatkan seleksi fitur atau gabungan antara seleksi fitur dan penyesuaian ulang, seperti Skenario 2 hingga Skenario 12, memberikan peningkatan kecil atau stabil pada akurasi, tetapi tidak mencapai level yang sama dengan Skenario 7. Secara keseluruhan, penanganan ketidakseimbangan data

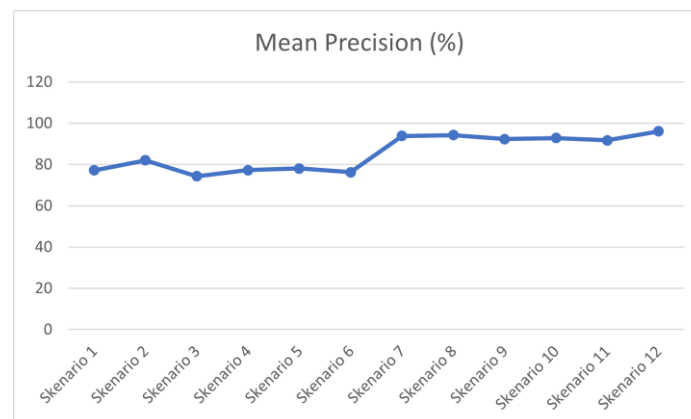
memberikan dampak paling besar terhadap performa model, mengoptimalkan deteksi kelas positif dan minoritas, sementara seleksi fitur tidak memberikan kontribusi yang signifikan dalam meningkatkan akurasi model.

#### 4.8.2 Analisis Evaluasi Matrik Presisi

Analisis pada matrix evaluasi presisi dilakukan pada 2 pengujian pembagian data, yakni diantaranya *train-test-split* dan *k-fold*, berikut penjelasannya:



Gambar 4.14 Rata-rata presisi pada evaluasi train-test-split



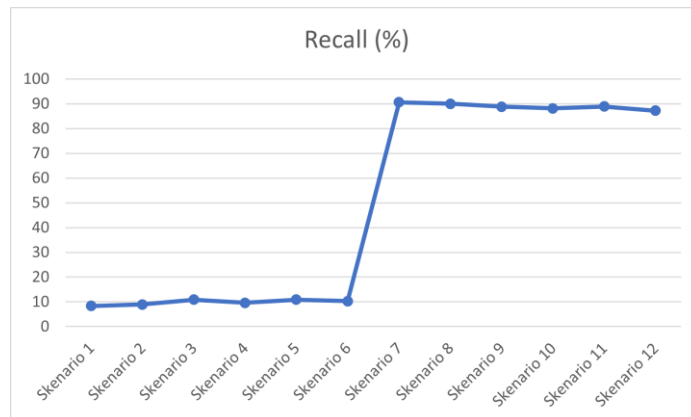
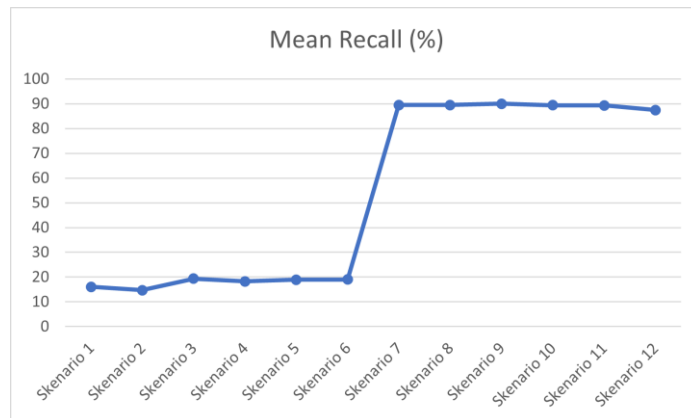
Gambar 4.15 Rata-rata presisi pada evaluasi k-fold

Berdasarkan analisis precision pada seluruh skenario, Skenario 7, yang melibatkan penanganan ketidakseimbangan data, memberikan hasil terbaik dengan presisi tertinggi yang tercatat pada model pembagian *train-test-split*, yaitu 95%.

Presisi yang tinggi ini mencerminkan kemampuan model untuk secara akurat memprediksi kelas positif, yang sangat penting dalam situasi ketidakseimbangan kelas. Skenario 12, yang menggabungkan penanganan ketidakseimbangan data dengan semua teknik seleksi fitur, juga menunjukkan presisi yang sangat baik 96,57%, menunjukkan bahwa kombinasi penanganan ketidakseimbangan dan seleksi fitur dapat lebih lanjut meningkatkan akurasi prediksi kelas positif. Skenario 8 dan Skenario 9 juga menunjukkan presisi yang tinggi dengan nilai masing-masing sebesar 95,85% dan 94,03%, yang mengindikasikan bahwa penanganan ketidakseimbangan data bersama dengan seleksi fitur mampu menghasilkan model yang lebih fokus pada prediksi kelas positif. Sebaliknya, skenario tanpa penanganan ketidakseimbangan data, seperti Skenario 1 hingga 6, memiliki presisi yang lebih rendah, dengan nilai maksimum berada di kisaran 80,79% hingga 84,86%, yang mengindikasikan bahwa model cenderung kesulitan dalam mendeteksi kelas positif tanpa penyesuaian terhadap ketidakseimbangan data.

#### **4.8.3 Analisis Evaluasi Matrik *Recall***

Analisis pada matrix evaluasi *recall* dilakukan pada 2 pengujian pembagian data, yakni diantaranya *train-test-split* dan *k-fold*, berikut penjelasannya:

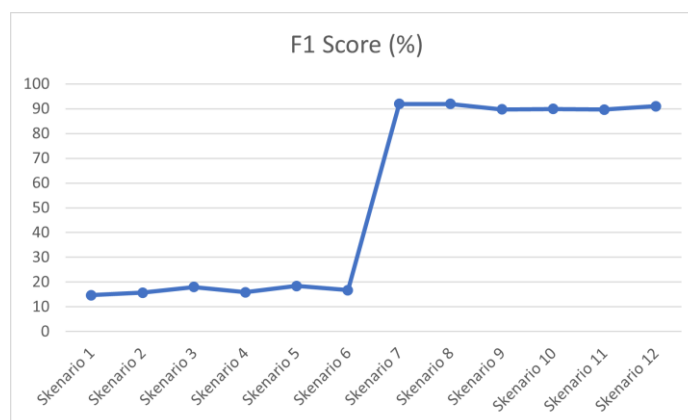
Gambar 4.16 Rata-rata *recall* pada evaluasi train-test-splitGambar 4.17 Rata-rata *recall* pada evaluasi k-fold

Berdasarkan *recall* Skenario 7 masih menunjukkan performa terbaik dengan model  $max\_depth=15$  dan  $n\_estimators=100$ , yang menghasilkan *recall* sangat tinggi 91,64%, mengindikasikan kemampuan model yang luar biasa dalam mendeteksi kelas positif, terutama setelah penanganan ketidakseimbangan data. Skenario 8, yang menggabungkan penanganan ketidakseimbangan data dengan seleksi fitur *Pearson correlation*, juga menunjukkan *recall* yang baik 91,43%, diikuti oleh Skenario 9 dengan *recall* sebesar 92,23% setelah gabungan penanganan ketidakseimbangan dan seleksi fitur ANOVA. Skenario lainnya, seperti Skenario 1 hingga Skenario 6, menunjukkan *recall* yang jauh lebih rendah, dengan nilai *recall* tertinggi mencapai 18,44% dan menurun pada skenario seleksi fitur *filtering*, seperti

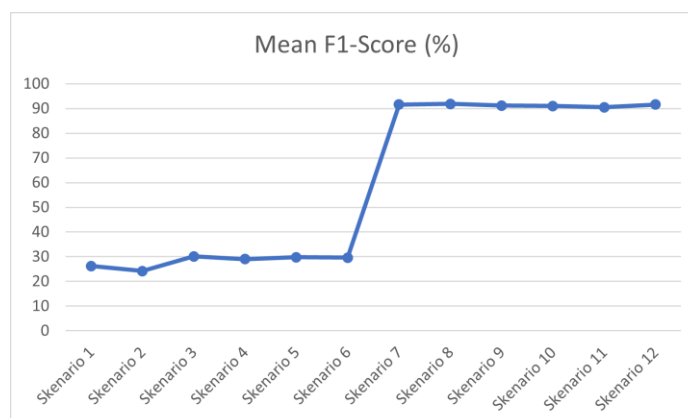
ANOVA dan *Pearson*, serta gabungannya, yang menunjukkan bahwa tanpa penanganan ketidakseimbangan data, model kesulitan dalam mendeteksi kelas positif.

#### 4.8.4 Analisis Evaluasi Matrik F1-Score

Analisis pada matrix evaluasi F1-Score dilakukan pada 2 pengujian pembagian data, yakni diantaranya *train-test-split* dan *k-fold*, berikut penjelasannya:



Gambar 4.18 Rata-rata f1-Score pada evaluasi train-test-split



Gambar 4.19 Rata-rata f1-Score pada evaluasi k-fold

Berdasarkan nilai F1-score, Skenario 7 yang menerapkan penanganan ketidakseimbangan data menunjukkan performa terbaik dengan F1-score tertinggi

sebesar 93,59% pada model *tuning* dengan *max\_depth*=15 dan *n\_estimators*=100, mencerminkan keseimbangan yang sangat baik antara presisi dan *recall* dalam mendeteksi kelas positif. Skenario 8 dan 9, yang menggabungkan penanganan ketidakseimbangan data dengan seleksi fitur *Pearson correlation* dan ANOVA, masing-masing menghasilkan F1-score sebesar 93,58% dan 93,11%, menunjukkan bahwa kombinasi antara penanganan ketidakseimbangan data dan seleksi fitur memberikan perbaikan signifikan dalam keseimbangan antara *precision* dan *recall*. Sedangkan skenario lainnya, seperti Skenario 5 dan Skenario 6, menunjukkan F1-score lebih rendah yakni masing-masing 33,64% dan 32,86%, menandakan bahwa meskipun ada beberapa perbaikan, model ini masih kesulitan dalam mendeteksi kelas positif dengan baik. Secara keseluruhan, penanganan ketidakseimbangan data terbukti menjadi faktor kunci yang meningkatkan F1-score, dengan kombinasi teknik seleksi fitur lebih lanjut memberikan hasil yang lebih seimbang dan akurat.

## 4.9 Pembahasan

### 4.9.1 Pengaruh Variasi Hyperparameter Random Forest

Pada seluruh skenario, peningkatan nilai *max\_depth* dan *n\_estimators* cenderung menghasilkan performa yang lebih baik. Model dengan kombinasi *max\_depth* 20 dan *n\_estimators* 150 secara umum menghasilkan nilai akurasi, *precision*, *recall*, dan *F1-score* yang lebih tinggi dibandingkan konfigurasi lainnya.

Hal ini menunjukkan bahwa jumlah pohon yang lebih banyak memungkinkan RF menghasilkan keputusan yang lebih stabil, sedangkan kedalaman pohon yang lebih tinggi memungkinkan model mempelajari pola data yang lebih kompleks. Namun demikian, peningkatan performa mulai melambat

pada konfigurasi tertinggi yang diuji, sehingga dapat disimpulkan bahwa penambahan *hyperparameter* secara berlebihan belum tentu menghasilkan peningkatan performa yang signifikan.

#### 4.9.2 Perbandingan Train-Test Split dan K-Fold Cross Validation

Hasil pengujian menunjukkan bahwa nilai performa yang diperoleh dari *K-Fold Cross Validation* memiliki pola yang relatif konsisten. Namun, *K-Fold Cross Validation* menghasilkan variasi nilai yang lebih kecil dan lebih representatif karena seluruh data berkesempatan menjadi data latih maupun data uji. Sebaliknya, *Train-Test Split* sangat bergantung pada pembagian data yang digunakan. Oleh karena itu, *K-Fold Cross Validation* dinilai lebih mampu menggambarkan kemampuan generalisasi model secara menyeluruh.

Secara keseluruhan, penelitian ini menunjukkan bahwa faktor yang paling berpengaruh terhadap peningkatan performa RF adalah balancing data menggunakan SMOTE. Seleksi fitur *Pearson Correlation* dan ANOVA berperan sebagai metode pendukung yang membantu meningkatkan *precision* dan *F1-score*, sedangkan tuning *hyperparameter* berkontribusi terhadap optimalisasi performa model. Kombinasi ketiga tahapan tersebut menghasilkan model prediksi *attrition* yang memiliki akurasi tinggi, kemampuan deteksi yang baik terhadap kelas minoritas, serta performa yang lebih stabil.

#### 4.9.3 Analisis Skenario Tanpa Balancing Data (Skenario 1–6)

Berdasarkan hasil pengujian, model RF tanpa *balancing* data menghasilkan nilai akurasi yang relatif tinggi, yaitu berkisar antara 84% hingga 87%. Namun,

nilai *recall* yang diperoleh masih sangat rendah, yaitu berada pada rentang 7% hingga 22%. Kondisi ini menunjukkan bahwa model cenderung mengklasifikasikan data ke kelas mayoritas dan mengalami kesulitan dalam mendeteksi karyawan yang berpotensi mengalami *attrition*.

Penambahan metode seleksi fitur menggunakan *Pearson Correlation* maupun ANOVA pada skenario 2 hingga skenario 6 belum mampu meningkatkan *recall* secara signifikan. Hal ini menunjukkan bahwa permasalahan utama pada dataset bukan terletak pada banyaknya fitur yang digunakan, melainkan pada ketidakseimbangan distribusi kelas. Dengan kata lain, meskipun fitur yang kurang relevan telah dihilangkan, model tetap mengalami bias terhadap kelas mayoritas karena jumlah data *attrition* yang lebih sedikit dibandingkan kelas non-*attrition*.

Selain itu, penggunaan seleksi fitur pada kondisi data yang belum seimbang hanya memberikan peningkatan kecil terhadap *precision* dan *F1-score*. Menunjukkan bahwa seleksi fitur saja belum cukup untuk meningkatkan kemampuan model dalam mendeteksi kelas minoritas.

#### **4.9.4 Pengaruh Balancing Data Menggunakan SMOTE**

Perubahan performa paling signifikan terjadi setelah penerapan SMOTE pada skenario 7 hingga skenario 12. Nilai *recall* meningkat secara drastis dari rata-rata di bawah 25% menjadi lebih dari 88%, bahkan mencapai 92,38% pada beberapa skenario. Peningkatan tersebut menunjukkan bahwa model menjadi lebih mampu mengenali pola pada kelas *attrition* yang sebelumnya kurang terwakili dalam dataset.

Selain recall, nilai *F1-score* juga meningkat secara signifikan dari kisaran 15% hingga 33% menjadi lebih dari 90%. Hasil ini menunjukkan bahwa penanganan *balancing* data menggunakan SMOTE merupakan faktor yang paling berpengaruh terhadap peningkatan performa model dibandingkan metode *preprocessing* lainnya.

Temuan ini sejalan dengan penelitian (Kaya & Şimşek, 2023) yang menunjukkan bahwa penerapan *balancing* data mampu meningkatkan kemampuan model dalam menangani dataset yang tidak seimbang. Dengan tersedianya jumlah sampel yang lebih seimbang pada masing-masing kelas, Random Forest dapat membangun pohon keputusan yang lebih representatif terhadap kedua kelas. Pada penelitian (Kaya & Şimşek, 2023), kombinasi *Random Forest* dan SMOTE menghasilkan akurasi sebesar 86%, sedangkan kombinasi terbaik ditemukan setelah penerapan *GridSearchCV* pada *Random Forest* dengan *Random Over Sampling* (ROS) menghasilkan *F1-score* sebesar 96%. Sementara itu, penelitian ini menunjukkan bahwa penerapan SMOTE pada *Random Forest* mampu menghasilkan performa yang kompetitif dengan akurasi mencapai 93,92%, dan *F1-score* 93,81% pada skenario 11 pada penerapan *hyperparameter* 20:150. Perbedaan utama penelitian ini terletak pada analisis yang lebih rinci melalui 12 skenario pengujian yang memungkinkan evaluasi kontribusi masing-masing tahapan *preprocessing*, sehingga tidak hanya menghasilkan model yang optimal tetapi juga memberikan pemahaman mengenai pengaruh *balancing* data, seleksi fitur, dan *hyperparameter* terhadap kualitas model.

#### 4.9.5 Pengaruh Seleksi Fitur Setelah Penerapan SMOTE

Pada skenario 8 hingga skenario 12, seleksi fitur dikombinasikan dengan balancing data menggunakan SMOTE. Hasil pengujian menunjukkan bahwa nilai akurasi dan F1-score mengalami peningkatan dibandingkan penggunaan SMOTE saja. Skenario 8 yang menggunakan kombinasi *Pearson Correlation* dan SMOTE menghasilkan akurasi tertinggi pada evaluasi *Train-Test Split* sebesar 94,53%, *precision* 96,57%, *recall* 92,21%, dan *F1-score* 94,34%. Hasil tersebut menunjukkan bahwa *Pearson Correlation* mampu membantu RF dalam mengurangi fitur yang kurang relevan sehingga proses pembentukan pohon keputusan menjadi lebih efektif. Sementara itu, skenario 12 yang menggunakan kombinasi *Pearson Correlation* dan ANOVA Penyesuaian menghasilkan *precision* tertinggi sebesar 97,09% pada evaluasi K-Fold. Nilai *precision* yang tinggi menunjukkan bahwa model memiliki tingkat kesalahan positif yang rendah dalam memprediksi karyawan yang berpotensi mengalami *attrition*. Berdasarkan hasil tersebut dapat disimpulkan bahwa seleksi fitur tidak memberikan pengaruh sebesar SMOTE, namun mampu meningkatkan kualitas model ketika dikombinasikan dengan balancing data.

#### 4.10 Integrasi Islam

Penelitian ini tidak hanya berupaya menghasilkan solusi ilmiah terhadap permasalahan prediksi penurunan jumlah karyawan, tetapi juga mencerminkan nilai-nilai Islam yang luhur, khususnya dalam hal amanah, tanggung jawab, dan keadilan. Dalam QS. An-Nisa ayat 58, Allah *Subhanahu wa Ta'ala* berfirman:

إِنَّ اللَّهَ يَأْمُرُكُمْ أَنْ تُؤَدُّوا الْأَمَانَاتِ إِلَىٰ أَهْلِهَا وَإِذَا حَكَمْتُمْ بَيْنَ النَّاسِ أَنْ تَحْكُمُوا بِالْعَدْلِ ۚ إِنَّ اللَّهَ نِعِمَّا يَعِظُكُمْ

بِهِ ۚ إِنَّ اللَّهَ كَانَ سَمِيعًا بَصِيرًا

*"Sesungguhnya Allah menyuruh kamu untuk menyampaikan amanat kepada yang berhak menerimanya, dan apabila kamu menetapkan hukum di antara manusia, maka hendaklah kamu memutuskan dengan adil." (QS. An-Nisa ayat 58).*

Menurut tafsir Ibnu Katsir pada buku berjudul *"Lubaabut Tafsirr Min Ibn Katsiir"*, didalamnya dijelaskan bahwa ayat ini menekankan kewajiban menyampaikan amanah kepada pihak yang berhak dan berlaku adil dalam pengambilan keputusan, baik dalam urusan publik maupun urusan pribadi. Amanah mencakup segala tanggung jawab yang dipercayakan, termasuk dalam pengelolaan manusia dan sumber daya, sehingga setiap keputusan harus mempertimbangkan hak-hak orang lain dan keadilan (Ghoffar et al., 2001).

Dalam konteks manajemen proyek dan sumber daya manusia, termasuk prediksi penurunan karyawan, perusahaan harus menjaga amanah dan bertindak adil. Prediksi penurunan karyawan dengan menggunakan algoritma Random Forest dapat dilihat sebagai implementasi amanah untuk menjaga keberlanjutan proyek dan stabilitas perusahaan. Alat prediksi ini bukan sekadar teknis, tetapi juga wujud nyata tanggung jawab moral untuk melindungi kesejahteraan karyawan dan mencegah kerugian organisasi. Nabi Muhammad *Shallallahu 'Alaihi wa Sallam* bersabda:

أَنَّ رَسُولَ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ أَلَا كُفُّكُمْ رَاعٍ وَكُلُّكُمْ مَسْئُولٌ عَنْ رَعِيَّتِهِ فَإِلَامَامٌ الَّذِي عَلَى النَّاسِ رَاعٍ

وَهُوَ مَسْئُولٌ عَنْ رَعِيَّتِهِ وَالرَّجُلُ رَاعٍ عَلَى أَهْلِ بَيْتِهِ وَهُوَ مَسْئُولٌ عَنْ رَعِيَّتِهِ وَالْمَرْأَةُ رَاعِيَةٌ عَلَى أَهْلِ بَيْتِ رَوْحِهَا

وَوَلَدِهِ وَهِيَ مَسْئُولَةٌ عَنْهُمْ وَعَبْدُ الرَّجُلِ رَاعٍ عَلَى مَالِ سَيِّدِهِ وَهُوَ مَسْئُولٌ عَنْهُ أَلَا فَكُلُّكُمْ رَاعٍ وَكُلُّكُمْ مَسْئُولٌ عَنْ رَعِيَّتِهِ

رَعِيَّتِهِ

*"Ketahuilah! Masing-masing kamu adalah pemimpin, dan masing-masing kamu akan dimintai pertanggung jawaban terhadap apa yang dipimpin. Seorang raja yang memimpin rakyat adalah pemimpin, dan ia akan dimintai pertanggung jawaban terhadap yang dipimpinnya. Seorang suami adalah pemimpin anggota keluarganya, dan ia akan dimintai pertanggung jawaban terhadap mereka. Seorang istri juga pemimpin bagi rumah tangga serta anak-suaminya, dan ia akan dimintai pertanggung jawaban terhadap yang dipimpinnya. Seorang budak juga pemimpin atas tuannya, dan ia akan dimintai pertanggung jawaban terhadap apa yang dipimpinnya. Ingatlah, masing-masing kamu adalah pemimpin dan masing-masing kamu akan dimintai pertanggung jawaban atas apa yang dipimpinnya."* (HR. Ibnu Umar).

Menurut Syarah Al-Bukhari oleh Imam Al-Khattabi dan Syarah Muslim oleh An-Nawawi, hadist ini menegaskan bahwa setiap orang yang diberi tanggung jawab apakah sebagai pemimpin organisasi, kepala rumah tangga, atau pengelola harta akan dimintai pertanggungjawaban di hadapan Allah (Istikhori et al., 2026). Imam An-Nawawi menjelaskan bahwa makna “ra’iyyah” adalah mereka yang berada dalam pengawasan dan tanggung jawab seseorang, sehingga seorang manajer atau HR yang melakukan prediksi penurunan karyawan bertindak sebagai pemimpin yang memikul amanah. Keputusan yang diambil harus adil, mempertimbangkan kesejahteraan karyawan, dan menjaga hak-hak mereka agar tidak ada pihak yang dirugikan (Istikhori et al., 2026).

Dengan demikian, integrasi nilai-nilai Islam menurut QS. An-Nisa 58 dari tafsir Ibnu Katsir dan Hadist Shahih Bukhari dan Muslim menekankan bahwa prediksi penurunan karyawan bukan sekadar aspek teknis, tetapi juga wujud

tanggung jawab moral, amanah, dan keadilan dalam pengelolaan sumber daya manusia.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1 Kesimpulan

Berdasarkan hasil penelitian terkait implementasi algoritma *Random Forest* dalam memprediksi penurunan jumlah karyawan dengan mempertimbangkan balancing data menggunakan SMOTE, seleksi fitur dengan metode filtering (Pearson Correlation dan ANOVA), serta variasi hyperparameter, dapat ditarik beberapa kesimpulan sebagai berikut:

##### 1. Efektivitas Random Forest dalam Prediksi Penurunan Jumlah Karyawan

Random Forest terbukti mampu memprediksi karyawan yang berisiko mengalami penurunan jumlah secara efektif, dengan model yang memiliki hyperparameter yang lebih tinggi, seperti *max\_depth* 20 dan *n\_estimators* 150, umumnya menghasilkan performa yang lebih baik, terutama pada skenario yang menggunakan balancing dengan rata-rata peningkatan presentase senilai 10%. Penggunaan metode evaluasi menggunakan *k-fold* cenderung lebih stabil dari pada penggunaan metode evaluasi menggunakan *train-test-split*.

##### 2. Pengaruh Balancing Data dan Seleksi Fitur

Balancing data menggunakan SMOTE secara signifikan meningkatkan nilai *recall* dan F1-score, terutama terlihat pada skenario 7 dengan akurasi, presisi, *recall*, dan F1-score senilai 93,92%, 94,09%, 92,38%, 93,22%, di mana *recall* meningkat drastis dibanding skenario tanpa SMOTE (skenario 1– skenario 6). Hal ini menunjukkan bahwa model lebih mampu mendeteksi karyawan yang berisiko keluar.

Seleksi fitur dengan *Pearson Correlation* dan ANOVA juga memberikan dampak positif terhadap presisi dan F1-score, terutama ketika dikombinasikan dengan SMOTE (skenario 8 – skenario 12). Kombinasi ini memungkinkan model untuk fokus pada fitur paling relevan dan mengurangi noise dari fitur yang kurang informatif. Sehingga penggunaan balancing data SMOTE dan seleksi fitur memiliki pengaruh yang signifikan terutama dalam mendeteksi

### 3. Performa Model Berdasarkan Metrik Evaluasi

Skenario terbaik secara keseluruhan ditunjukkan oleh Skenario 12 yakni kombinasi SMOTE dengan seleksi fitur gabungan menggunakan penyesuaian, dengan performa tertinggi akurasi senilai 93,07%, presisi 97,09%, *recall* 88,80%, dan F1-score 92,75%. Skenario tanpa *balancing* data menunjukkan nilai *recall* yang sangat rendah misalnya pada Skenario 6 dengan nilai akurasi, presisi, *recall*, F1-Score, berturut-turut senilai 86,19%, 72,13%, 21,94%, 33,33%, hal ini mengindikasikan model masih cenderung bias terhadap kelas mayoritas.

Dengan demikian, kombinasi balancing data dan seleksi fitur terbukti optimal dalam meningkatkan kemampuan *Random Forest* untuk memprediksi penurunan jumlah karyawan secara seimbang antara deteksi positif dan negatif. Pola analisis yang dapat dikembangkan yakni seperti penggunaan pipeline kombinasi preprocessing, seleksi fitur, dan tuning hyperparameter terbukti dapat membentuk pola analisis prediktif yang stabil. Model dapat digunakan sebagai alat pendukung pengambilan keputusan dalam manajemen SDM, khususnya untuk identifikasi risiko turnover dan strategi retention.

## 5.2 Saran

Penelitian berikutnya dapat mengeksplorasi algoritma *ensemble* lain seperti *XGBoost* atau *LightGBM* untuk membandingkan performa dengan *Random Forest* atau dengan algoritma *Machine Learning* bahkan *Deep Learning* lainnya. Melakukan uji coba komputasi antar metode untuk menentukan keefektifan model atau dengan menambahkan evaluasi matrix lainnya seperti ROC-AUC. Menambahkan atupun membandingkan dengan metode seleksi fitur lainnya, seperti metode *Wrapped Selection* atau *Embedded Selection*.

Penelitian selanjutnya juga bisa menambahkan terkait penanganan *outliers*, serta analisis fitur tambahan, seperti fitur berbasis perilaku atau psikometrik karyawan, juga dapat ditambahkan untuk meningkatkan prediktabilitas model. Eksperimen dengan teknik balancing lain seperti *Random Over Sampling* (ROS), *Adasyn*, atau metode hybrid seperti gabungan metode SMOTE untuk *oversampling* dengan metode *undersampling*, dapat dilakukan untuk melihat pengaruhnya terhadap metrik evaluasi. Diperlukan percobaan optimasi hyperparameter lebih lanjut seperti penggunaan *GridSearchCV* atau *RandomSearchCV*, untuk menemukan konfigurasi parameter terbaik. Disarankan untuk mengintegrasikan model ke dalam dashboard HR Analytics agar dapat digunakan oleh manajer HR dalam pengambilan keputusan strategis, misal untuk identifikasi karyawan berisiko tinggi dan intervensi proaktif.

## DAFTAR PUSTAKA

- Al-Alawi, A. I., & Albuainain, M. S. (2024). Machine Learning in Human Resource Analytics: Promotion Classification using Data Balancing Techniques. *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSYS)*, 1–5. <https://doi.org/10.1109/ICETSYS61505.2024.10459566>
- Alfaridzy, M. A., Haerani, E., Jasril, & Oktavia, L. (2025). Klasifikasi Sentimen Masyarakat Terhadap Efisiensi Anggaran Pemerintah Menggunakan Metode Naïve Bayes Classifier. *Journal Education and Technology*, 6, 169–182. <https://jurnal.stkipppersada.ac.id/jurnal/index.php/jutech/article/view/4969>
- Annisa, T. N., Jasmir, J., & Nurhadi, N. (2025). Comparison of ANOVA and Chi-Square Feature Selection Methods to Improve Machine Learning Performance in Anemia Classification. *Jurnal Teknik Informatika (Jutif)*, 6(4), 1925–1940. <https://doi.org/10.52436/1.jutif.2025.6.4.5017>
- Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Budiman, H., Nurohmah, A., Nadia Pratiwi, C., & Nur Islami, T. (2025). Implikasi Instruksi Presiden Nomor 1 Tahun 2025 Terhadap Efektivitas Dan Kinerja Pemerintahan Daerah. *Justicia Journal*, 14(2), 187–197. <https://doi.org/10.32492/jj.v14i2.14201>
- Bureau Labor Statistics U. S. Departement Of Labor. (2024). Employee Tenure In 2024. *Bureau Labor Statistics U. S. Departement Of Labor*, 42(5).
- Buzeti, J., Klun, M., & Stare, J. (2016). Evaluation of measures to reduce employee turnover in Slovenian organisations. *E+M Ekonomie a Management*, 19(1), 121–131. <https://doi.org/10.15240/tul/001/2016-1-009>
- Candrawati, E. K. A. (2025). *PENGUATAN KAPASITAS SDM DALAM MENDUKUNG PENERAPAN SISTEM MANAJEMEN K3 UNTUK MENINGKATKAN PRODUKTIVITAS (Fenomelogi di PT. ABC)*. <https://etheses.uin-malang.ac.id/75677/>
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321–357. <https://doi.org/10.1613/jair.953>
- Cheng, X. (2024). A Comprehensive Study of Feature Selection Techniques in Machine Learning Models. *Insights in Computer, Signals and Systems*, 1(1), 65–78. <https://doi.org/10.70088/xpf2b276>
- Cherian, J. A., Simon, S., & Sreelatha, A. P. (2025). Predictive Modeling of Employee Attrition Using Tree-Based Machine Learning Classifiers. *2025 Emerging Technologies for Intelligent Systems (ETIS)*, 1–6.

<https://doi.org/10.1109/ETIS64005.2025.10961183>

- Churchill, M., Ginting, H., & Wicaksono, A. (2023). The Determinant Factors for Employees to Leave (A Case Study in a Paint Chain Retailer). *International Journal of Current Science Research and Review*, 06(01), 801–820. <https://doi.org/10.47191/ijcsrr/V6-i1-87>
- Crawford, S. L. (2006). Correlation and Regression. *Circulation*, 114(19), 2083–2088. <https://doi.org/10.1161/CIRCULATIONAHA.105.586495>
- Direktorat Statistik Kependudukan dan Ketenagakerjaan. (2023). Booklet Survei Angkatan Kerja Nasional Agustus 2021. *Badan Pusat Statistik*, 11(1), ii–24.
- Elreedy, D., & Atiya, A. F. (2019). A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance. *Information Sciences*, 505, 32–64. <https://doi.org/10.1016/j.ins.2019.07.070>
- Erbani, J., Portier, P.-É., Egyed-Zsigmond, E., & Nurbakova, D. (2024). Confusion Matrices: A Unified Theory. *IEEE Access*, 12(November), 181372–181419. <https://doi.org/10.1109/ACCESS.2024.3507199>
- Fernandez, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *Journal of Artificial Intelligence Research*, 61, 863–905. <https://doi.org/10.1613/jair.1.11192>
- Ghoffar, M. A., Mu'thi, A., & Al-Atsari, A. I. (2001). *Tafsir Ibnu Katsir*. Pustaka Imam asy-Syafi'i.
- Goswami, T., & Roy, U. B. (2021). Classification Accuracy Comparison for Imbalanced Datasets with Its Balanced Counterparts Obtained by Different Sampling Techniques. In A. Kumar & S. Mozar (Eds.), *ICCCE 2020* (pp. 45–54). Springer Nature Singapore.
- Hadi, D. A., & Sirodj, D. A. N. (2023). Metode Random Forest untuk Klasifikasi Penyakit Diabetes. *Bandung Conference Series: Statistics*, 3(2), 428–435. <https://doi.org/10.29313/bcss.v3i2.8354>
- Hamdard, A. P. M. S., & Lodin, A. P. H. (2023). Effect of Feature Selection on the Accuracy of Machine Learning Model. *INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS*, 06(09), 4460–4466. <https://doi.org/10.47191/ijmra/v6-i9-66>
- Handayani, I., & Ikrimach, I. (2020). Accuracy Analysis of K-Nearest Neighbor and Naïve Bayes Algorithm in the Diagnosis of Breast Cancer. *JURNAL INFOTEL*, 12(4), 151–159. <https://doi.org/10.20895/infotel.v12i4.547>
- Hutahaean, N. D. (2025, June 11). Simak Alasan Perusahaan Menambah dan Mengurangi Karyawan. *GoodStats*. <https://goodstats.id/article/survei-alasan-perusahaan-melakukan-peningkatan-dan-pengurangan-karyawan-tetap-IvbqV>

- Istikhori, I., Suhendar, D., Nurainillah, E., Irawati, I., Anwar, H., & Hasibuan, S. C. (2026). Pertanggungjawaban Setiap Orang kepada Allah Swt (Analisis Tematik dan Kritis terhadap Riwayat Al-Bukhari, Muslim, Abu Dawud, dan Al-Tirmidzi). *RIGGS: Journal of Artificial Intelligence and Digital Business*, 4(4), 15285–15297. <https://doi.org/10.31004/riggs.v4i4.6155>
- Jia, W. (2024). Research on the Influencing Factors of Job Satisfaction of Grassroots Employee of R Project Management Company. *Advances in Economics, Management and Political Sciences*, 100(1), 104–113. <https://doi.org/10.54254/2754-1169/100/20240878>
- Karimi, M., & Viliyani, K. S. (2024). *Employee Turnover Analysis Using Machine Learning Algorithms*. <https://doi.org/10.48550/arXiv.2402.03905>
- Kaufmann, J., & Schering, A. (2014). Analysis of Variance <sc>ANOVA</sc>. In *Wiley StatsRef: Statistics Reference Online* (pp. 1–12). Wiley. <https://doi.org/10.1002/9781118445112.stat06938>
- Kavlakoglu, E. (2026). *What is a decision tree?* The 2026 Guide to Machine Learning. <https://www.ibm.com/think/topics/decision-trees>
- Kaya, Ç., & Şimşek, M. (2023). Prediction of Employee Turnover with Imbalance Dataset Using Machine Learning Methods. *International Journal of Advanced Natural Sciences and Engineering Researches*. <https://doi.org/10.59287/ijanser.1495>
- Kementrian Tenaga Kerja Republik Indonesia. (2025). *Tenaga Kerja Ter-PHK, Februari Tahun 2025*. Kementrian Tenaga Kerja RI. <https://satudata.kemnaker.go.id/data/kumpulan-data/2742>
- Kitole, F. A., & Utouh, H. M. L. (2024). Foreign direct investment and industrialization in Tanzania admixture time series forecast analysis 1960 - 2020. *Applied Economics Letters*, 31(20), 2110–2117. <https://doi.org/10.1080/13504851.2023.2211324>
- Lee Rodgers, J., & Nicewander, W. A. (1988). Thirteen Ways to Look at the Correlation Coefficient. *The American Statistician*, 42(1), 59–66. <https://doi.org/10.1080/00031305.1988.10475524>
- Lugoma, M., Zimbili, A. O., Ilunga, M., Mosia, N., & Abhishek, A. (2025). Effect of Data Imbalance in Predicting Student Performance in a Structural Analysis Graduate Attribute-Based Module Using Random Forest Machine Learning. *Proceedings IMCIC - International Multi-Conference on Complexity, Informatics and Cybernetics, Imcic*, 85–92. <https://doi.org/10.54808/IMCIC2025.01.85>
- Lukashevich, M. M., & Klitsunova, K. (2025). Experimental Studies on the Application of Data Balancing Methods in Classification Problems. *Doklady BGUIR*, 23(5), 66–74. <https://doi.org/10.35596/1729-7648-2025-23-5-66-74>
- Miller, C., Portlock, T., Nyaga, D. M., & O’Sullivan, J. M. (2024). A review of

- model evaluation metrics for machine learning in genetics and genomics. *Frontiers in Bioinformatics*, 4(September), 1–13. <https://doi.org/10.3389/fbinf.2024.1457619>
- Mwakiluma, L. (2024). The Impact of Employee Turnover on Organizational Performance in the Service Sector of Tanzania: A Case Study of Tanzania Ports Authority. *NG Journal of Social Development*, 14(1), 1–18. <https://doi.org/10.4314/ngjsd.v14i1.1>
- Pace, L. (2012). One-Way Analysis of Variance. In *Beginning R* (pp. 139–147). Apress. [https://doi.org/10.1007/978-1-4302-4555-1\\_10](https://doi.org/10.1007/978-1-4302-4555-1_10)
- Pagan, M., Zarlis, M., & Candra, A. (2023). Investigating the impact of data scaling on the k-nearest neighbor algorithm. *Computer Science and Information Technologies*, 4(2), 135–142. <https://doi.org/10.11591/csit.v4i2.pp135-142>
- Pandey, D., Vishwakarma, Z., & Dwivedi, M. (2024). Attrition Analytics: Unveiling the Best Model for Predicting Employee Retention. *International Journal of Innovative Research in Computer and Communication Engineering*, 12(Special Is), 30–37. <https://doi.org/10.15680/IJIRCCE.2024.1203505>
- Pulungan, A. F., Zarlis, M., & Suwilo, S. (2019). Analysis of Braycurtis, Canberra and Euclidean Distance in KNN Algorithm. *Sinkron*, 4(1), 74. <https://doi.org/10.33395/sinkron.v4i1.10207>
- Qiu, J. (2024). An Analysis of Model Evaluation with Cross-Validation: Techniques, Applications, and Recent Advances. *Advances in Economics, Management and Political Sciences*, 99(1), 69–72. <https://doi.org/10.54254/2754-1169/99/2024OX0213>
- Rainio, O., Teuho, J., & Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, 14(1), 6086. <https://doi.org/10.1038/s41598-024-56706-x>
- Raza, A., Munir, K., Almutairi, M., Younas, F., & Fareed, M. M. S. (2022). Predicting Employee Attrition Using Machine Learning Approaches. *Applied Sciences*, 12(13), 6424. <https://doi.org/10.3390/app12136424>
- Sahoo, R. (2020). *IBM Employee Dataset*. <https://www.kaggle.com/datasets/rohitsahoo/employee>
- Salman, H. A., Kalakech, A., & Steiti, A. (2024). Random Forest Algorithm Overview. *Babylonian Journal of Machine Learning*, 2024, 69–79. <https://doi.org/10.58496/BJML/2024/007>
- Schober, P., Boer, C., & Schwarte, L. A. (2018). Correlation Coefficients: Appropriate Use and Interpretation. *Anesthesia & Analgesia*, 126(5), 1763–1768. <https://doi.org/10.1213/ANE.0000000000002864>
- Shabdar, A. (2009). Manage Your Projects. In *Foundation Zoho* (pp. 625–665).

Apress. [https://doi.org/10.1007/978-1-4302-1992-7\\_13](https://doi.org/10.1007/978-1-4302-1992-7_13)

- Shaker, F., Raad Shaker Alnaily, R., Naeem Turkey, S., Kareem Wanas, E., & Sadiq Sadon, S. (2025). Machine Learning-Based Heart Disease Detection with ANOVA Feature Selection. *Journal of Al-Qadisiyah for Computer Science and Mathematics*, 17(3), 1–11. <https://doi.org/10.29304/jqcs.2025.17.32427>
- Sharma, S., & Sharma, K. (2023). Analyzing Employee's Attrition and Turnover at Organization Using Machine learning Technique. *2023 3rd International Conference on Intelligent Technologies (CONIT)*, 1–7. <https://doi.org/10.1109/CONIT59222.2023.10205676>
- Sibirina, L. A., & Gladkova, G. K. A. (2012). Chapter 25. In *"The Story of Apollonius, King of Tyre"* (pp. 311–328). DE GRUYTER. <https://doi.org/10.1515/9783110214130.311>
- Silpa, N., Maheswara Rao, V. V. R., Subbarao, M. V., Kurada, R. R., Reddy, S. S., & Uppalapati, P. J. (2023). An Enriched Employee Retention Analysis System with a Combination Strategy of Feature Selection and Machine Learning Techniques. *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 142–149. <https://doi.org/10.1109/ICICCS56967.2023.10142473>
- Sirshar, M., Liaqat, R., & Siddique, S. (2019). Effective Human Resource Management in Project Management. *Business and Engineering*, 1(9), 5. <https://api.semanticscholar.org/CorpusID:209165346>
- Stanton, J. M. (2001). Galton, Pearson, and the Peas: A Brief History of Linear Regression for Statistics Instructors. *Journal of Statistics Education*, 9(3). <https://doi.org/10.1080/10691898.2001.11910537>
- Subdirektorat Statistik Ketenagakerjaan. (2019). Booklet Survei Angkatan Kerja Nasional Februari 2020. *Badan Pusat Statistik*, 1–9.
- Tang, Z., Gu, J., & Kelkar, M. (2025). Enhancing Employee Retention: Predicting Attrition Using Machine Learning Models. *Journal of Applied Business and Economics*, 27(3), 41–53. <https://doi.org/10.33423/jabe.v27i3.7644>
- Tangirala, S. (2020). Evaluating the Impact of GINI Index and Information Gain on Classification using Decision Tree Classifier Algorithm\*. *International Journal of Advanced Computer Science and Applications*, 11(2), 612–619. <https://doi.org/10.14569/IJACSA.2020.0110277>
- Usha, P. M., & Balaji, N. V. (2021). A comparative study on machine learning algorithms for employee attrition prediction. *IOP Conference Series: Materials Science and Engineering*, 1085(1), 012029. <https://doi.org/10.1088/1757-899X/1085/1/012029>
- Venema, K., & Gueniche, A. (2014). Study on project schedule and cost overruns. In *Project Management Institute* (Vol. 5, Issue 2). <http://www.ncbi.nlm.nih.gov/pubmed/24699022>

- Wafa, I. (2025). *Ini Dia 7 Alasan Banyak PHK di 2025 Menurut Kemnaker*. GoodStats. <https://data.goodstats.id/statistic/ini-dia-7-alasan-banyak-phk-di-2025-menurut-kemnaker-TLFhN>
- Wan, S., & Yang, H. (2013). Comparison among Methods of Ensemble Learning. *2013 International Symposium on Biometrics and Security Technologies*, 286–290. <https://doi.org/10.1109/ISBAST.2013.50>
- Wilson, L. T. (2018). *Pearson Product-Moment Correlation*. <https://api.semanticscholar.org/CorpusID:4652750>