



## BAB III

### METODE PENELITIAN

Bab ini menjelaskan metode yang digunakan dalam penelitian analisis sentimen ulasan aplikasi JobStreet pada Google Play Store menggunakan SMOTE dengan algoritma XGBoost dan Random Forest. Metodologi penelitian disusun secara sistematis mulai dari jenis penelitian, sumber data, pengumpulan data, preprocessing teks, ekstraksi fitur TF-IDF, penerapan SMOTE, proses klasifikasi menggunakan XGBoost dan Random Forest, hingga evaluasi model.

#### 3.1 Jenis Penelitian

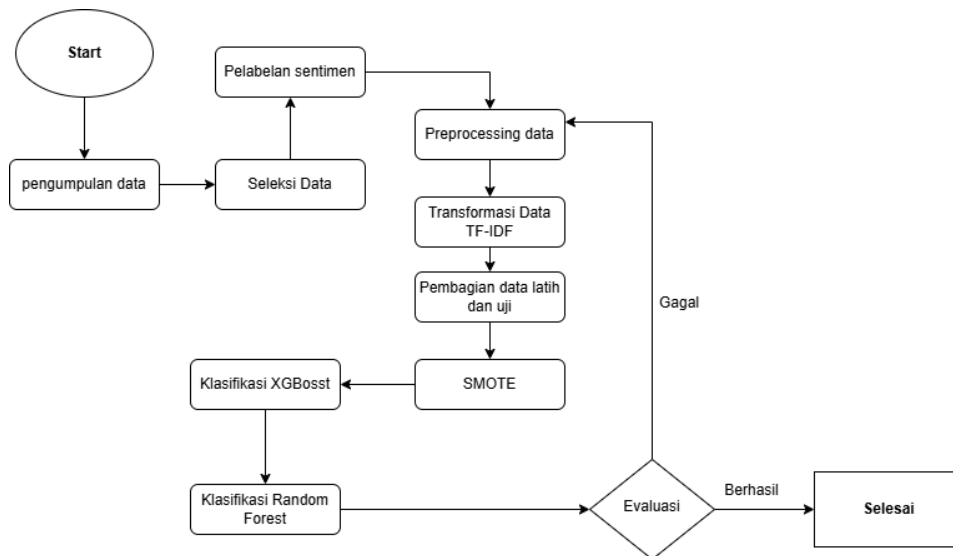
Penelitian ini merupakan penelitian kuantitatif dengan pendekatan eksperimen. Pendekatan kuantitatif digunakan karena data ulasan pengguna diolah dalam bentuk angka melalui proses ekstraksi fitur, sedangkan pendekatan eksperimen digunakan untuk menerapkan teknik SMOTE serta algoritma XGBoost dan Random Forest dalam klasifikasi sentimen.

Penelitian ini bertujuan menganalisis sentimen ulasan aplikasi JobStreet pada Google Play Store ke dalam kelas positif dan negatif, serta membandingkan performa algoritma XGBoost dan Random Forest berdasarkan nilai accuracy, precision, recall, dan F1-score.

#### 3.2 Kerangka Penelitian

menggambarkan alur penelitian secara sistematis, mulai dari pengumpulan data, pelabelan sentimen, preprocessing, transformasi data menggunakan TF-IDF, pembagian data latih dan data uji, penerapan SMOTE, hingga klasifikasi menggunakan algoritma XGBoost dan Random Forest. Selanjutnya, hasil

klasifikasi dievaluasi untuk mengetahui performa masing-masing algoritma.



**Gambar 3. 1 Kerangka Penelitian**

### 3.3 Teknik Pengumpulan Data

Sebelum menjelaskan teknik pengumpulan data, berikut disampaikan gambaran umum mengenai objek penelitian, yaitu aplikasi JobStreet, beserta panduan penggunaannya oleh pengguna, sebagai konteks terhadap data ulasan yang menjadi bahan analisis dalam penelitian ini. JobStreet merupakan aplikasi pencarian kerja yang mempertemukan pencari kerja dengan perusahaan penyedia lowongan melalui layanan pencarian lowongan, pengiriman lamaran, dan pengelolaan profil secara daring.

Secara umum, panduan penggunaan aplikasi JobStreet oleh pengguna meliputi tahapan sebagai berikut:

1. Mengunduh dan memasang (*install*) aplikasi JobStreet melalui Google Play Store pada perangkat pengguna.
2. Melakukan pendaftaran akun (registrasi) atau masuk (*login*) menggunakan alamat email maupun akun media sosial.

3. Melengkapi profil pengguna, seperti data diri, riwayat pendidikan, dan pengalaman kerja, serta mengunggah *curriculum vitae* (CV).
4. Melakukan pencarian lowongan pekerjaan berdasarkan kata kunci, lokasi, maupun kategori bidang pekerjaan yang diinginkan.
5. Memilih lowongan yang sesuai dan mengirimkan lamaran kerja (*apply*) secara langsung melalui aplikasi.
6. Memantau status lamaran serta menerima notifikasi terkait perkembangan proses rekrutmen.
7. Memberikan ulasan dan rating terhadap aplikasi JobStreet melalui Google Play Store berdasarkan pengalaman penggunaan.

Tahapan penggunaan aplikasi tersebut menjadi dasar bagi peneliti dalam memahami konteks ulasan yang diberikan pengguna, baik yang berkaitan dengan kemudahan pencarian kerja maupun kendala teknis yang dialami selama menggunakan aplikasi

Teknik pengumpulan data dilakukan dengan mengambil ulasan pengguna aplikasi JobStreet dari Google Play Store menggunakan teknik web scraping dengan bantuan library Python, seperti google-play-scraper.

Data yang dikumpulkan berupa teks ulasan, rating, dan tanggal ulasan. Setelah itu, dilakukan seleksi data untuk menghapus data kosong, duplikat, dan ulasan yang tidak relevan.

**Tabel 3. 1 Format data awal**

| No  | Review                         | Rating | Tanggal  |
|-----|--------------------------------|--------|----------|
| (1) | (2)                            | (3)    | (4)      |
| 1   | Aplikasi Sangat Membantu Kerja | 5      | 2026     |
| 2   | Susah Login                    | 2      | 2025.... |

### 3.4 Seleksi Data

Seleksi data dilakukan untuk memastikan data yang digunakan sesuai dengan kebutuhan penelitian. Data yang tidak lengkap, kosong, duplikat, atau tidak menggunakan bahasa Indonesia dapat dihapus.

Tahaan seleksi data meliputi:

**Tabel 3. 2 Tahapan Seleksi Data**

| <b>Tahapan</b>           | <b>Keterangan</b>                                             |
|--------------------------|---------------------------------------------------------------|
| <b>(1)</b>               | <b>(2)</b>                                                    |
| Menghapus data kosong    | Menghapus ulasan yang tidak memiliki isi teks                 |
| Menghapus duplikat       | Menghapus ulasan yang sama agar tidak memengaruhi hasil model |
| Memilih bahasa Indonesia | Memilih ulasan berbahasa Indonesia                            |
| Memilih atribut penting  | Menggunakan atribut review, rating, tanggal, dan label        |

### 3.5 Pelabelan Sentimen

Pelabelan sentimen dilakukan untuk menentukan kategori sentimen dari setiap ulasan. Pada penelitian ini, label sentimen dapat dibagi menjadi 2 kategori, positif dan negatif. Pelabelan awal dapat dilakukan berdasarkan rating pengguna sebagai berikut:

**Tabel 3. 3 Pelabelan Awal**

| <b>Rating</b> | <b>Label Sentimen</b> |
|---------------|-----------------------|
| <b>(1)</b>    | <b>(2)</b>            |
| 1-3           | Negatif               |
| 4-5           | Positif               |

Tahap ini bertujuan untuk memastikan kualitas data yang digunakan dalam penelitian.

### 3.6 Preprocessing Teks

Preprocessing data merupakan tahap pengolahan teks agar data ulasan menjadi lebih bersih dan siap digunakan dalam proses klasifikasi. Data ulasan dari

Google Play Store umumnya masih mengandung huruf kapital, tanda baca, angka, emoji, URL, simbol, dan kata tidak baku. Berikut preprocessing dalam penelitian ini meliputi:

**Tabel 3. 4 Tahapan Proprocessing**

| No  | Tahapan          | Keterangan                                                                  |
|-----|------------------|-----------------------------------------------------------------------------|
| (1) | (2)              | (3)                                                                         |
| 1   | Case folding     | Mengubah seluruh huruf menjadi huruf kecil                                  |
| 2   | Cleaning         | Menghapus angka, tanda baca, simbol, emoji, URL, dan karakter tidak penting |
| 3   | Tokenizing       | Memecah kalimat menjadi kata atau token                                     |
| 5   | Stopword removal | Menghapus kata umum yang kurang bermakna                                    |
| 6   | Stemming         | Mengubah kata berimbuhan menjadi kata dasar                                 |

### 3.7 Ekstraksi Fitur Menggunakan TF-IDF

Setelah preprocessing selesai, data teks diubah ke dalam bentuk numerik agar dapat diproses oleh algoritma machine learning. Pada penelitian ini, metode yang digunakan adalah TF-IDF (Term Frequency-Inverse Document Frequency). TF-IDF digunakan untuk memberikan bobot pada kata berdasarkan frekuensi kemunculan dan tingkat kepentingannya dalam dokumen. Semakin penting suatu kata dalam membedakan dokumen, maka semakin besar bobot yang diberikan. Hasil proses TF-IDF berupa matriks numerik yang digunakan sebagai input bagi algoritma XGBoost dan Random Forest. Dokumentasi Scikit-learn menjelaskan bahwa TfidfVectorizer digunakan untuk mengubah kumpulan dokumen menjadi matriks fitur TF-IDF[7].

### 3.8 Pembagian Data Training dan Testing

Data yang telah melalui proses ekstraksi fitur kemudian dibagi menjadi data training dan data testing. Data training digunakan untuk melatih model, sedangkan data testing digunakan untuk menguji performa model pada data yang belum pernah digunakan sebelumnya. Pembagian data dapat menggunakan skenario berikut :

**Tabel 3. 5 Skenario Pembagian Data Training dan Testing**

| <b>Skenario</b> | <b>Data Training</b> | <b>Data Testing</b> |
|-----------------|----------------------|---------------------|
| <b>(1)</b>      | <b>(2)</b>           | <b>(3)</b>          |
| 1               | 80%                  | 20%                 |

Dalam penelitian ini, data dibagi menjadi 80% data training dan 20% data testing. Pembagian ini umum digunakan karena menyediakan data training yang cukup untuk proses pembelajaran model serta data testing untuk evaluasi performa.

### **3.9 Penerapan SMOTE**

SMOTE (Synthetic Minority Over-sampling Technique) digunakan untuk mengatasi ketidakseimbangan data, yaitu kondisi ketika jumlah data pada setiap kelas sentimen tidak seimbang, seperti kelas positif yang lebih banyak dibandingkan kelas negatif. SMOTE bekerja dengan menghasilkan data sintetis pada kelas minoritas sehingga distribusi data menjadi lebih seimbang. Dokumentasi imbalanced-learn menjelaskan bahwa SMOTE merupakan teknik oversampling untuk menghasilkan sampel sintetis pada kelas minoritas[9].

Dalam penelitian ini, SMOTE diterapkan hanya pada data training agar data testing Pembagian dilakukan dengan rasio tertentu, misalnya 80:20. tetap merepresentasikan kondisi data asli dan menghindari kebocoran data. Contoh distrubsi SMOTE:

**Tabel 3. 6 Distribusi Data Sebelum dan Sesudah Penerapan SMOTE**

| <b>Sentimen</b> | <b>Sebelum SMOTE</b> | <b>Sesudah SMOTE</b> |
|-----------------|----------------------|----------------------|
| <b>(1)</b>      | <b>(2)</b>           | <b>(3)</b>           |
| Positif         | 2.000                | 2.000                |
| Negatif         | 500                  | 2.000                |

### **3.10 Klasifikasi Menggunakan XGBoost**

XGBoost (Extreme Gradient Boosting) merupakan algoritma machine learning berbasis gradient boosting yang bekerja dengan membangun model secara

bertahap untuk memperbaiki kesalahan model sebelumnya. Dalam penelitian ini, XGBoost digunakan untuk mengklasifikasikan ulasan aplikasi JobStreet ke dalam kelas sentimen positif dan negatif menggunakan data training yang telah diseimbangkan dengan SMOTE.

implementasinya dilakukan melalui library XGBoost Python dengan parameter bawaan ( $n\_estimators=100$ ,  $learning\_rate=0.3$ ,  $max\_depth=6$ ,  $subsample=1.0$ ), karena fokus penelitian adalah membandingkan performa XGBoost dengan Random Forest setelah SMOTE, bukan mengoptimalkan parameter.

Model dilatih menggunakan data yang telah diproses dengan TF-IDF dan SMOTE, melalui fungsi `XGBClassifier()` dan metode `.fit()`. Setelah pelatihan, model diuji dengan data uji dan dievaluasi menggunakan *confusion matrix* serta *classification report* untuk memperoleh nilai akurasi, presisi, *recall*, dan F1-score

Dokumentasi XGBoost menjelaskan bahwa XGBoost merupakan library gradient boosting yang dirancang secara efisien, fleksibel, dan portabel untuk berbagai tugas machine learning[22].

**Tabel 3. 7 Tahapan Klasifikasi Menggunakan XGBoost**

| <b>Tahapan</b> | <b>Keterangan</b>                                       |
|----------------|---------------------------------------------------------|
| <b>(1)</b>     | <b>(2)</b>                                              |
| Input          | Data training hasil TF-IDF dan SMOTE                    |
| Proses         | Pelatihan model XGBoost                                 |
| Output         | Prediksi sentimen positif dan negatif                   |
| Evaluasi       | Accuracy, precision, recall, F1-score, confusion matrix |

### 3.11 Klasifikasi Menggunakan Random Forest

Random Forest merupakan algoritma ensemble learning yang membangun banyak decision tree untuk menghasilkan prediksi akhir. Setiap tree memberikan

prediksi, kemudian hasil akhir ditentukan berdasarkan voting. Dalam penelitian ini, Random Forest digunakan untuk mengklasifikasikan sentimen ulasan aplikasi JobStreet menggunakan data training yang telah melalui proses TF-IDF dan SMOTE.

Implementasinya menggunakan Scikit-learn versi 1.5.0 dengan parameter bawaan ( $n\_estimators=100$ ,  $max\_depth=None$ ,  $min\_samples\_split=2$ ,  $min\_samples\_leaf=1$ ), karena fokus penelitian adalah perbandingan performa algoritma, bukan optimasi parameter. Hasil yang diperoleh merepresentasikan performa dasar Random Forest pada data ulasan JobStreet.

Pelatihan dilakukan dengan `RandomForestClassifier()` dan metode `.fit()` pada data yang telah diproses TF-IDF dan SMOTE. Setelah itu, model diuji dengan data uji. Evaluasi performa menggunakan *confusion matrix* dan *classification report* untuk mengukur akurasi, presisi, *recall*, dan F1-score

Dokumentasi Scikit-learn menjelaskan bahwa Random Forest merupakan meta estimator yang membangun sejumlah decision tree pada berbagai sub-sampel dataset untuk meningkatkan akurasi prediksi dan mengurangi overfitting[13].

**Tabel 3. 8 Tahapan Klasifikasi Menggunakan Random Forest**

| <b>Tahapan<br/>(1)</b> | <b>Keterangan<br/>(2)</b>                               |
|------------------------|---------------------------------------------------------|
| Input                  | Data training hasil TF-IDF dan SMOTE                    |
| Proses                 | Pelatihan model Random Forest                           |
| Output                 | Prediksi sentimen positif dan negatif                   |
| Evaluasi               | Accuracy, precision, recall, F1-score, confusion matrix |

Dokumentasi Scikit-learn menjelaskan bahwa Random Forest merupakan meta estimator yang membangun sejumlah decision tree pada berbagai sub-sampel dataset untuk meningkatkan akurasi prediksi dan mengurangi overfitting[13]

### 3.12 Evaluasi Model

Evaluasi model dilakukan untuk mengetahui performa algoritma XGBoost dan Random Forest dalam mengklasifikasikan sentimen ulasan aplikasi JobStreet. Proses evaluasi menggunakan data testing yang tidak melalui SMOTE agar hasil pengujian tetap mencerminkan kondisi data asli.

**Tabel 3. 9 Metrik Evaluasi Model**

| <b>Metrik</b>    | <b>Keterangan</b>                                                |
|------------------|------------------------------------------------------------------|
| <b>(1)</b>       | <b>(2)</b>                                                       |
| Accuracy         | Mengukur jumlah prediksi benar dari seluruh data                 |
| Precision        | Mengukur ketepatan model dalam memprediksi suatu kelas           |
| Recall           | Mengukur kemampuan model menemukan seluruh data pada suatu kelas |
| F1-score         | Mengukur keseimbangan antara precision dan recall                |
| Confusion matrix | Menampilkan jumlah prediksi benar dan salah pada setiap kelas    |

Scikit-learn menyediakan fungsi `classification_report` untuk menampilkan precision, recall, F1-score, dan support, serta fungsi `confusion_matrix` untuk mengevaluasi hasil klasifikasi berdasarkan label aktual dan label prediksi[15]. Selain itu, digunakan juga confusion matrix untuk melihat hasil klasifikasi secara lebih detail.

**Tabel 3. 10 Rencana Perbandingan Hasil Evaluasi XGBoost dan Random Forest**

| <b>Algoritma</b>      | <b>Accuracy</b> | <b>Precision</b> | <b>Recall</b> | <b>F1-score</b> |
|-----------------------|-----------------|------------------|---------------|-----------------|
| <b>(1)</b>            | <b>(2)</b>      | <b>(3)</b>       | <b>(4)</b>    | <b>(5)</b>      |
| XGBoost + SMOTE       | ...             | ...              | ...           | ...             |
| Random Forest + SMOTE | ...             | ...              | ...           | ...             |

### 3.13 Perbandingan Hasil Algoritma

Setelah model XGBoost dan Random Forest diuji, hasil evaluasi kedua algoritma dibandingkan untuk menentukan algoritma dengan performa terbaik dalam klasifikasi sentimen ulasan aplikasi JobStreet.

Perbandingan dilakukan berdasarkan metrik accuracy, precision, recall, F1-score, dan confusion matrix. Apabila terdapat perbedaan hasil antar metrik, maka F1-score dijadikan pertimbangan utama karena mampu menunjukkan keseimbangan antara precision dan recall, terutama pada data tidak seimbang.

Penelitian ini mengimplementasikan dua algoritma klasifikasi, yaitu XGBoost dan Random Forest, masing-masing menggunakan library Python dengan parameter bawaan. XGBoost versi 2.0.3 dikonfigurasi dengan `n_estimators=100`, `learning_rate=0.3`, `max_depth=6`, dan `subsample=1.0`, sedangkan Random Forest dari Scikit-learn versi 1.5.0 menggunakan `n_estimators=100`, `max_depth=None`, `min_samples_split=2`, dan `min_samples_leaf=1`. Karena tujuan penelitian adalah membandingkan performa algoritma pasca-SMOTE, bukan mengoptimalkan model, maka parameter default pada kedua algoritma digunakan sebagai tolok ukur awal yang objektif.

### 3.14 Perangkat dan Tools Penelitian

Penelitian ini dapat dilakukan menggunakan perangkat keras dan perangkat lunak pada tabel berikut:

**Tabel 3. 11 Perangkat Keras**

| <b>Komponen</b> | <b>Spesifikasi</b> |
|-----------------|--------------------|
| <b>(1)</b>      | <b>(2)</b>         |
| Laptop/PC       | HP                 |
| RAM             | 4 GB               |
| Penyimpanan     | 128 GB             |
| Sistem Operasi  | Windows            |

**Tabel 3. 12 Perangkat Lunak**

| <b>Tools</b> | <b>Fungsi</b>            |
|--------------|--------------------------|
| <b>(1)</b>   | <b>(2)</b>               |
| Python       | Bahasa pemrograman utama |

|                                 |                                                   |
|---------------------------------|---------------------------------------------------|
| Google Colab / Jupyter Notebook | Lingkungan pengolahan data                        |
| Pandas                          | Pengolahan dataset                                |
| NumPy                           | Operasi numerik                                   |
| Scikit-learn                    | TF-IDF, split data, evaluasi model, Random Forest |
| Imbalanced-learn                | Penerapan SMOTE                                   |
| XGBoost                         | Klasifikasi menggunakan XGBoost                   |
| Matplotlib / Seaborn            | Visualisasi data dan hasil evaluasi               |