LAPORAN TUGAS BESAR 02

IF3170 Inteligensi Artifisial

Implementasi Algoritma Pembelajaran Mesin

Disusun Oleh:

Enrique Yanuar	13522077
Ivan Hendrawan Tan	13522111
Derwin Rustanly	13522115
Mesach Harmasendro	13522117

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2024

DAFTAR ISI

BAB I
DESKRIPSI PERMASALAHAN

Pembelajaran mesin merupakan salah satu cabang dari kecerdasan buatan yang membuat sistem dapat belajar dari data dan membuat prediksi atau keputusan tanpa perlu diprogram secara eksplisit. Dataset yang digunakan untuk membuat model sistem ini adalah dataset UNSW-NB15.

Dataset UNSW-NB15 adalah sekumpulan data lalu lintas jaringan yang terdiri dari berbagai jenis serangan siber dan juga aktivitas normalnya. Algoritma yang perlu diterapkan untuk mengimplementasikan pembelajaran mesin yaitu k-Nearest Neighbor (KNN), Gaussian Naive-Bayes, dan Iterative Dichotomiser 3 (ID3) pada dataset UNSW-NB15. Implementasi KNN, Gaussian Naive-Bayes, dan ID3 dibuat dari scratch dan dibandingkan dengan algoritma dari scikit-learn. Di akhir, model harus bisa di-save dan di-load dengan implementasinya dapat menggunakan .txt, .pkl, dll.

BAB II
PEMBAHASAN

1. K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) adalah algoritma supervised learning yang dapat digunakan untuk klasifikasi. Prinsip dasar KNN sangat intuitif: untuk memprediksi kelas atau nilai suatu sampel baru, algoritma mencari sejumlah k tetangga terdekat di ruang fitur. Pada kasus klasifikasi, hasil prediksi ditentukan oleh mayoritas kelas dari para tetangga terdekat tersebut.

Berbeda dengan beberapa algoritma lain yang membentuk model eksplisit selama tahap pelatihan, KNN tidak membangun model matematika yang kompleks. Pada tahap pelatihan (fit), KNN hanya menyimpan data latih (X_train, y_train) dan tidak melakukan proses komputasi berat. Proses penentuan kelas atau nilai prediksi sepenuhnya terjadi pada tahap prediksi (predict), ketika algoritma menghitung jarak antara sampel uji dan setiap sampel latih. Jarak ini dapat dihitung dengan berbagai metrik, seperti euclidean, manhattan, atau minkowski, dengan penyesuaian nilai eksponen p pada metrik minkowski. Setelah seluruh jarak dihitung, k sampel terdekat dipilih, dan kelas atau nilai prediksi ditentukan sesuai dengan mayoritas kelas atau rata-rata nilai k tetangga tersebut.

Pada awalnya, pendekatan implementasi KNN dilakukan secara sekuensial, di mana setiap sampel uji dibandingkan dengan seluruh data latih satu per satu. Hal ini dapat menjadi sangat lambat, terutama ketika jumlah data latih berukuran besar. Untuk mengatasi masalah ini, di terapkanlah konsep paralelisasi (multi-threading) pada tahap prediksi. Dengan demikian, perhitungan jarak untuk beberapa sampel uji dapat dijalankan secara bersamaan, mempercepat waktu komputasi secara signifikan dan membuat KNN menjadi lebih efisien. Fasilitas ini sering diimplementasikan dalam bentuk atribut kelas, seperti jumlah thread (n_jobs), sehingga pengguna dapat menyesuaikan jumlah thread yang digunakan sesuai dengan sumber daya komputasi yang tersedia.

2. Gaussian Naive-Bayes

Gaussian Naive Bayes adalah salah satu varian sederhana dari algoritma klasifikasi Naive Bayes yang mengasumsikan bahwa setiap fitur mengikuti distribusi Gaussian (normal) dan bahwa setiap fitur dianggap bersyarat independen dari fitur lainnya. Proses pembentukan model ini terbagi menjadi dua tahap utama, yaitu tahap pelatihan (training) dan tahap prediksi.

Pada tahap pelatihan, data yang tersedia terlebih dahulu dibagi berdasarkan kelas, kemudian untuk masing-masing kelas dihitung nilai rata-rata (mean) dan varians (variance) dari setiap fitur. Nilai mean dan varians tersebut digunakan untuk memodelkan distribusi probabilitas fitur yang diharapkan mengikuti pola distribusi Gaussian. Selain itu, pada tahap ini juga dihitung prior probability, yaitu peluang dasar kemunculan setiap kelas dalam data latih. Informasi mengenai mean, variance, dan prior inilah yang membentuk dasar model probabilistik yang akan digunakan untuk prediksi.

Selanjutnya, pada tahap prediksi, setiap sampel baru yang akan diklasifikasikan diukur seberapa besar kemungkinan (likelihood) kemunculannya pada distribusi masing-masing kelas. Likelihood ini dihitung dengan menggunakan fungsi kepadatan probabilitas Gaussian yang telah dibangun sebelumnya berdasarkan parameter mean dan varians. Nilai likelihood yang dihasilkan kemudian digabungkan dengan prior probability menggunakan Teorema Bayes, sehingga diperoleh posterior probability untuk setiap kelas. Kelas dengan posterior probability tertinggi akan dipilih sebagai prediksi akhir.

Dengan pendekatan ini, Naive Bayes Gaussian mampu memprediksi kelas suatu sampel baru secara efisien. Meskipun algoritma ini didasarkan pada asumsi kemerdekaan bersyarat antar fitur yang tidak selalu sesuai dengan kenyataan, dalam praktiknya pendekatan ini seringkali memberikan hasil yang cukup memuaskan, terutama pada masalah klasifikasi dengan jumlah fitur yang banyak dan keterbatasan data atau sumber daya komputasi.

3. Iterative Dichotomiser 3 (ID3)

ID3 (Iterative Dichotomiser 3) merupakan salah satu algoritma awal yang digunakan untuk membangun pohon keputusan (decision tree) dari data berlabel. Algoritma ini memilih atribut yang paling informatif berdasarkan information gain, dimana semakin tinggi nilai information gain, semakin baik atribut tersebut dalam memisahkan data menjadi subset yang lebih homogen. Prosesnya dimulai dengan menghitung entropi awal pada seluruh data, lalu secara berurutan menghitung information gain untuk setiap atribut. Atribut dengan information gain tertinggi dipilih sebagai node pemisah, dan data dipilah menjadi subset berdasarkan nilai atribut tersebut. Langkah ini diulang secara rekursif pada subset-subset berikutnya, hingga seluruh data pada sebuah node termasuk ke dalam satu kelas yang sama atau tidak ada atribut lain yang dapat memisahkan data lebih lanjut.

Pada tahap train (pelatihan), ID3 membangun pohon keputusan dengan melakukan semua perhitungan entropi, information gain, dan pemilihan atribut secara berulang. Hasil akhir dari tahap ini adalah sebuah pohon keputusan lengkap yang dapat digunakan untuk memprediksi kelas sampel baru. Pada implementasi awal, ID3 dijalankan secara sekuensial, sehingga setiap perhitungan dilakukan satu per satu. Pendekatan ini efektif untuk dataset kecil, tetapi menjadi sangat lambat ketika jumlah data dan fitur sangat besar.

Untuk mengatasi masalah skala, ID3 dapat diimplementasikan secara paralel. Dengan memanfaatkan komputasi paralel, perhitungan information gain untuk tiap atribut dapat dilakukan bersamaan, sehingga proses pemilihan atribut terbaik pada setiap node menjadi lebih cepat dan efisien. Meskipun demikian, prinsip dasar ID3 tidak berubah: proses tetap berfokus pada pemilihan atribut yang paling meningkatkan informasi dalam upaya memecah data menjadi subset yang lebih homogen.

Pada tahap predict (prediksi), pohon keputusan yang sudah terbentuk digunakan untuk menentukan kelas sampel baru. Proses ini tidak memerlukan perhitungan entropy dan information gain. Cukup dengan menelusuri pohon keputusan dari akar ke daun, memilih cabang yang sesuai berdasarkan nilai atribut sampel tersebut, hingga mencapai node daun yang menentukan kelas prediksi akhir. Dengan cara ini, ID3 dapat digunakan dengan efisien untuk melakukan klasifikasi

pada data baru, setelah sebelumnya melakukan tahap train yang mungkin telah dioptimalkan dengan komputasi paralel.

4. Tahapan Cleaning
    a. Penghapusan Kolom id dan label:

       Pada tahap awal pembersihan data (data cleaning), dilakukan penghapusan kolom yang tidak relevan, yaitu kolom id dan label. Kolom-kolom ini umumnya bukan fitur yang berkontribusi langsung terhadap proses prediksi, sehingga dihilangkan untuk menghindari noise.
    b. Outlier Clamping (Outlier Capping) dengan OutlierCapper:

       Outlier seringkali dapat mendistorsi distribusi data serta model yang dihasilkan. Transformer OutlierCapper berfungsi untuk membatasi nilai ekstrim pada tingkat yang masih masuk akal dengan menggunakan median, persentil tertentu (misalnya persentil 95%), atau ambang batas yang ditentukan. Jika suatu nilai melebihi threshold yang ditentukan (misal, 10 kali median), nilai tersebut akan digantikan oleh nilai cap yang telah ditetapkan. Hal ini bertujuan menstabilkan distribusi dan mengurangi pengaruh outlier ekstrim tanpa harus menghapus data tersebut.
    c. Remove Duplicates (Duplikasi Data):

       Data duplikat dihapus untuk mengurangi bias dan redundansi informasi. Meski langkah ini tidak meningkatkan akurasi model secara signifikan, membersihkan duplikasi membantu menjaga integritas dataset.

5. Tahapan Preprocessing
    a. Feature Scaling & Data Normalization:
        i. Robust Scaling dengan RobustScalerTransformer atau RobustScalerWithImputer:

           Scaling data diperlukan agar fitur numerik yang memiliki rentang nilai berbeda tidak mendominasi model. Robust Scaler menggunakan median dan IQR, yang lebih tahan terhadap outlier.

           RobustScalerWithImputer selain melakukan scaling, juga mengimputasi nilai yang hilang terlebih dahulu menggunakan strategi yang ditentukan (default median), sebelum mengaplikasikan robust

scaling. Hal ini memastikan data yang hilang tidak mempengaruhi hasil scaling.

b. Log Transformer (LogTransformer):

Fitur dengan distribusi yang sangat menceng dapat dilog-transform agar distribusi lebih mendekati normal. Jika suatu kolom numerik memiliki jumlah nilai unik yang tinggi, log-transform membantu mengurangi skewness dan dapat memudahkan model dalam belajar pola data.

c. Feature Encoding:

i. OneHotEncoderTransformer:

Fitur kategorik dengan nominal values banyak perlu diubah menjadi representasi numerik. OneHotEncoderTransformer mengubah setiap kategori menjadi kolom biner. Drop parameter drop='first' digunakan untuk menghindari perangkap dummy. handle_unknown='ignore' mengatasi kategori yang muncul pada data inferensi tetapi tidak ada pada data latih.

ii. ReduceLabelTransformer:

Jika terdapat kategori yang terlalu banyak (misalnya >5), maka kategori yang jarang muncul dapat digabung ke dalam satu kategori umum (misalnya "-"). Hal ini menyederhanakan ruang fitur dan mencegah sparsity yang berlebihan.

iii. LabelEncoderTransformer (untuk Y):

Untuk variabel target kategorikal, LabelEncoderTransformer akan mengubah label kategori menjadi nilai numerik. Ini penting agar model dapat memproses target dengan benar, terutama pada kasus klasifikasi.

d. Handling Imbalanced Data:

i. ImbalancedDataHandler:

Ketidakseimbangan kelas dapat memperburuk performa model. ImbalancedDataHandler dapat melakukan oversampling pada kelas minoritas atau undersampling pada kelas mayoritas. Meskipun menurut percobaan tidak meningkatkan akurasi secara signifikan, tahapan ini tetap layak dicoba untuk meningkatkan keseimbangan dan kestabilan model.

ii. SMOTE

Class SMOTEUnderSampler merupakan transformer yang

menggabungkan teknik SMOTE (Synthetic Minority Oversampling Technique) dengan teknik undersampling untuk menyeimbangkan distribusi kelas pada data. Ketika data klasifikasi memiliki distribusi kelas yang tidak seimbang, model cenderung condong (bias) ke kelas mayoritas. Teknik penyeimbangan data ini berguna untuk memperbaiki performa model, terutama pada metrik yang peka terhadap kelas minoritas (misalnya recall pada kelas jarang).

e. Dimensionality Reduction (PCA) dengan PCATransformer:

Principal Component Analysis (PCA) digunakan untuk mereduksi dimensi data tanpa kehilangan informasi penting secara signifikan. PCATransformer memproyeksikan data ke komponen utama yang merangkum variabilitas tertinggi dalam data. Fitur seperti explained_variance_ratio_ dan singular_values_ dapat digunakan untuk menganalisis pentingnya masing-masing komponen. Meskipun upaya ini tidak meningkatkan akurasi, PCA dapat bermanfaat dalam hal interpretabilitas, komputasi, dan pengurangan noise.

6. Tahapan Preprocessing

Berdasarkan ketiga algoritma yang telah diimplementasikan dari scratch dan dibandingkan dengan library dari scikit-learn, urutan algoritma pembelajaran mesin dari yang terbaik ke yang terburuk adalah ID3, KNN, lalu Gaussian Naive-Bayes.

ID3 yang memiliki hasil F1 score terbaik memiliki accuracy dan recall sebesar 77,99%, precision sebesar 78,18%, F1 score sebesar 55,86%, dan waktu total untuk menjalankan algoritma scratch ini sebesar 1777 detik. Sedangkan hasil ID3 dari library scikit-learn memiliki tingkat accuracy dan recall sebesar 79,27%, precision sebesar 79,04%, F1 score sebesar 57,05%, dan waktu total untuk menjalankan library ini adalah sebesar 2,51 detik.

KNN yang berada di urutan kedua hasil dari scratch memiliki accuracy sebesar 77,56%, recall sebesar 77,56%, precision sebesar 76,84%, F1 score sebesar 48,41%, dan total waktu dari mulai menjalankan train dan predict hingga selesai sebesar 172,19 detik. Hasil dari library scikit-learn memiliki tingkat accuracy sebesar

75,95%, recall sebesar 75,95%, precision sebesar 77,08%, F1 score sebesar 48,21%, dan total waktu dari mulai hingga selesai sebesar 3,65 detik.

Gaussian Naive-Bayes dari scratch memiliki tingkat accuracy dan recall sebesar 47,62%, precision sebesar 68,71%, F1 score sebesar 25,20%, dan lama waktu keseluruhan untuk menjalankan sebesar 2,44 detik. Sedangkan dengan algoritma GaussianNB dari library scikit-learn memiliki tingkat accuracy dan recall sebesar 49,12%, precision sebesar 68,88%, F1 score sebesar 26,31%, dan waktu total yang digunakan sebesar 0,17 detik.

Berdasarkan perbandingan hasil F1 score dengan total waktu, algoritma dengan hasil F1 score terbaik adalah ID3 dengan F1 score sebesar 55,86%, lalu KNN dengan F1 score sebesar 48,41, dan Gaussian Naive-Bayes sebesar 25,2%. Meskipun memiliki F1 score yang terbaik, proses ID3 memakan waktu yang sangat lama dibanding dengan kedua algoritma lainnya yang memakan waktu hingga 30 menit, berbeda dengan KNN yang hanya memakan waktu hampir 3 menit dengan selisih F1 score dengan ID3 sebesar 7,45%. Algoritma tercepat dipegang oleh Gaussian Naive-Bayes, namun algoritma ini memiliki F1 score terburuk.

BAB III
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan implementasi dan evaluasi algoritma KNN, Gaussian Naive-Bayes, dan ID3 pada dataset UNSW-NB15, ditemukan bahwa algoritma ID3 memiliki performa terbaik dengan F1 score tertinggi. Keunggulan ID3 terletak pada penggunaan algoritma pohon keputusan yang mampu menangkap hubungan non-linear antara fitur dengan pendekatan berbasis informasi gain. Hal ini membuat ID3 lebih adaptif dalam menghadapi dataset kompleks dengan tipe data yang beragam, serta menangani nilai yang hilang secara efisien.

Sebaliknya, performa Naive Bayes berada di peringkat terendah karena asumsi independensinya yang terlalu kuat, sehingga kurang mampu menangani hubungan kompleks antara fitur dan cenderung bias terhadap kelas mayoritas. Adapun KNN, meskipun memiliki akurasi yang baik, sensitivitasnya terhadap skala dan distribusi data menempatkannya di bawah ID3.

Dalam hal penanganan data yang hilang, metode imputasi terbukti lebih efektif dibandingkan penghapusan data karena mampu mempertahankan pola-pola penting dalam dataset tanpa mengurangi representativitas data. Selain itu, penerapan teknik scaling seperti robust scaler dan transformasi log terbukti meningkatkan performa model, terutama pada KNN dan ID3 dengan mengurangi pengaruh outlier dan memperbaiki distribusi data.

Secara keseluruhan, penggunaan teknik preprocessing yang tepat, seperti imputasi, scaling, dan normalisasi, memberikan kontribusi signifikan terhadap peningkatan performa model pembelajaran mesin.

B. Saran

1. Peningkatan Teknik Preprocessing

Untuk memperbaiki performa model, penelitian lanjutan dapat mengeksplorasi metode preprocessing yang lebih canggih yang lebih sesuai dengan distribusi data.

2. **Optimasi Hyperparameter yang Lebih Mendalam**

   Teknik optimasi seperti grid search atau bayesian optimization dapat digunakan untuk menemukan konfigurasi hyperparameter terbaik pada setiap algoritma guna memaksimalkan performa.

3. **Meningkatkan Fleksibilitas Model dari Scratch**

   Model KNN dan Naive Bayes yang diimplementasikan dari scratch dapat ditingkatkan dengan kemampuan menerima parameter konfigurasi. Fitur ini mempermudah hyperparameter tuning tanpa perlu mengubah kode sumber, sehingga eksperimen dapat dilakukan lebih efisien dan terstruktur.

PEMBAGIAN TUGAS

Nama / NIM	Tugas
Enrique Yanuar / 13522077	● Tuning Data ● Optimize algo
Ivan Hendrawan Tan / 13522111	● Tuning Data ● Optimize algo
Derwin Rustanly / 13522115	● Algoritma ● Optimize tuning
Mesach Harmasendro / 13522117	● Algoritma ● Optimize tuning

REFERENSI

Data Preparation. Diakses pada 15 Desember 2024 dari https://cdn-edunex.itb.ac.id/64464-Artificial-Intelligence-Parent-Class/297373-Data-Preparation/1730818790714_IF-3170-Data-Preparation.pdf

k-Nearest Neighbor. Diakses pada 15 Desember 2024 dari https://cdn-edunex.itb.ac.id/53145-Artificial-Intelligence-Parallel-Class/210071-Supervised-Learning/90133-Supervised-Learning/1699250331293_IF3170_Materi09_Seg01_AI-kNN.pdf

Naive Bayes. Diakses pada 15 Desember 2024 dari https://cdn-edunex.itb.ac.id/53145-Artificial-Intelligence-Parallel-Class/210071-Supervised-Learning/90133-Supervised-Learning/1699250380758_IF3170_Materi09_Seg02_AI-NaiveBayes.pdf

Modul: Decision Tree Learning (DTL). Diakses pada 15 Desember 2024 dari https://cdn-edunex.itb.ac.id/64464-Artificial-Intelligence-Parent-Class/298936-Modeling-Decision-Tree-Learning-DTL/120485-Modul-Introduction-to-DTL/1731401412073_IF3170_SupervisedLearning_DTL.pdf