Nama/ NIM : Rafli Limandijaya/1103210243

Hasil summary metrics

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Bagging	0.999448	0.920	0.732	0.816	0.904
KNN	0.999436	0.943	0.704	0.806	0.904
Decision Tree	0.999095	0.734	0.718	0.726	0.859
Logistic Regression	0.999095	0.842	0.563	0.675	0.964
SVM	0.999084	0.864	0.535	0.661	0.962
Boosting	0.998625	0.745	0.268	0.394	0.401

Confusion Matrix(KNN sebagai contoh saja karena yang lainnya mirip-mirip bisa dilihat di colab notebook)

	Predicted 0	Predicted 1
Actual 0 (Normal)	84970	6
Actual 1 (Fraud)	42	100

Sebelum memulai analisis, dataset ini sangat imbalance dan akurasi tingginya bisa menipu karena misal :

- Total data: 84970 + 6 + 42 + 100 = 85000 + 148 = 85118

- Fraud cases (Actual 1): 42 + 100 = 142 , itu cuma ~0.16% dari total data.

Misal model asal nebak semua 0, akurasi sudah 99.8%. Oleh karena itu Accuracy jadi metrik yang kurang baik buat kasus fraud detection seperti ini.

Yang harus diperhatikan lebih penting adalah:

- Recall untuk Class 1 , mengacu ke pertanyaan : Apakah banyak fraud yang berhasil dideteksi.
- F1-Score, yaitu balance antara Precision dan Recall.

Bagian 1

1. Analisis dan Pemilihan Model Terbaik

Pada tugas klasifikasi ini, dataset memiliki distribusi label yang sangat imbalanced, di mana kelas normal (Class 0) mendominasi hampir seluruh data. Hal ini menyebabkan akurasi model menjadi sangat tinggi, meskipun model belum tentu baik dalam mendeteksi kelas minoritas (Class 1 – fraud).

Karena itu, akurasi tidak menjadi metrik utama yang dipertimbangkan. Sebagai gantinya, Recall, Precision, F1-Score, dan ROC AUC digunakan untuk evaluasi performa model.

Berdasarkan hasil evaluasi:

Model	**F1-Score**	**Recall**	**ROC AUC**
Bagging	0.8157	0.7324	0.9046
KNN	0.8065	0.7042	0.9048
Decision Tree	0.7260	0.7183	0.8589
Logistic Regression	0.6751	0.5634	0.9637
SVM	0.6609	0.5352	0.9620
Boosting	0.3938	0.2676	0.4013

Dari tabel tersebut dapat disimpulkan bahwa:

- Bagging memiliki F1-Score tertinggi (0.8157) dan Recall yang cukup tinggi (0.7324), sehingga menjadi model terbaik untuk deteksi fraud pada dataset ini.
- KNN juga menunjukkan performa yang hampir setara.
- Logistic Regression dan SVM memiliki ROC AUC tinggi, namun Recall rendah, yang artinya banyak kasus fraud tidak terdeteksi.

Oleh karena itu, Bagging dipilih sebagai model terbaik untuk tugas klasifikasi ini.

2. Penjelasan setiap model

A. Logistic Regression
Logistic Regression membangun model klasifikasi berdasarkan fungsi sigmoid, mengestimasi probabilitas sebuah data masuk ke dalam kelas tertentu.

Kelebihannya sederhana, cepat dilatih, dan interpretatif. Namun, pada dataset yang sangat imbalanced seperti ini, Logistic Regression cenderung bias ke kelas mayoritas dan memiliki Recall yang lebih rendah.

B. Decision Tree
   Decision Tree membuat pohon keputusan dengan mempartisi fitur menggunakan kriteria tertentu (seperti Gini Index atau Entropy). Decision Tree fleksibel menangani data kategorikal dan numerikal, tapi rentan overfitting tanpa pruning, terutama pada dataset yang imbalanced.
C. K-Nearest Neighbors (KNN)
   KNN mengklasifikasikan data berdasarkan mayoritas tetangga terdekat. Sangat bergantung pada distribusi data dan scaling. Pada kasus ini, KNN bekerja cukup baik karena outlier minoritas (fraud) bisa ditemukan berdasarkan jarak, walau butuh tuning k yang tepat.
D. Bagging
   Bagging (Bootstrap Aggregating) membuat banyak model (seperti banyak decision tree) dari resampling data berbeda lalu menggabungkan hasil prediksinya. Ini mengurangi varians dan meningkatkan generalisasi. Bagging cocok untuk kasus imbalanced karena banyak sampel minoritas bisa termuat dalam bootstrap samples.
E. Boosting
   Boosting membangun model secara bertahap, fokus memperbaiki kesalahan prediksi sebelumnya. Cocok untuk dataset kompleks tapi berisiko overfitting di data minoritas jika tuning kurang tepat. Pada kasus ini, Boosting underfit, mungkin karena terlalu mengabaikan minoritas.
F. Support Vector Machine (SVM)
   SVM mencari hyperplane optimal untuk memisahkan kelas. Dengan kernel linear, SVM sederhana tapi kurang fleksibel pada decision boundary kompleks. Scaling sangat penting agar SVM perform baik. Pada kasus ini, SVM punya ROC tinggi tapi Recall rendah.

Bagian 2

Jawaban dari pertanyaan analisis 1-5

1. Penyebab dan Solusi AUC tinggi, Presisi rendah

a. Faktor penyebab:

-AUC-ROC mengukur kemampuan model membedakan kelas positif dan negatif pada semua threshold.

-Presisi fokus pada seberapa banyak prediksi positif yang benar.

-Ketidaksesuaian terjadi jika model memang bisa membedakan fraud vs non-fraud, tapi threshold default terlalu rendah yang menyebabkan banyak false positives.

b. Strategi tuning hyperparameter:

-Adjust threshold dari default 0.5 ke nilai lebih tinggi untuk meningkatkan Precision (sedikit kurangi Recall, tapi AUC tetap tinggi).

-Menggunakan class weight seimbang (class_weight='balanced') pada model.

-Tuning regularization (seperti C di SVM, Logistic Regression) untuk kontrol margin yang lebih ketat.

c. Kepentingan Recall di fraud detection

-False Negative (fraud yang lolos) jauh lebih mahal (biaya finansial, reputasi).

-Recall tinggi memastikan lebih banyak fraud terdeteksi, meski ada false positives.

-Hubungan dengan cost:

- False negative -> keuangan bocor.
- False positive -> sedikit lebih banyak verifikasi manual, tapi lebih aman.

2. Fitur high-cardinality dan Data Leakage

a. Dampak 1000 nilai unik:

-Estimasi koefisien jadi tidak stabil -> terlalu banyak parameter -> overfitting.

-Presisi turun karena noise berlebih dari nilai unik yang jarang muncul.

b. Penyebab target encoding riskan leakage

-Target encoding menggunakan rata-rata label untuk setiap kategori.

-Jika dihitung sebelum split, model "mengintip" label test data.

c. Alternatif encoding yang lebih aman:

-Leave-one-out encoding (LOO): saat encode satu data, tidak memakai labelnya sendiri.

-Frequency encoding: mengganti kategori dengan frekuensi kemunculannya.

-Embedding (kalau pakai model kompleks, seperti neural net).

3. Normalisasi Min-Max, SVM, dan Margin Minoritas

a. Dampak normalisasi Min-Max:

-Mengubah semua fitur ke rentang [0,1].

-SVM sangat tergantung pada ukuran fitur, normalisasi membuat hyperplane lebih optimal.

-Presisi naik karena decision boundary lebih rapat.

-Recall turun karena margin mengecil, minoritas sulit masuk kelas positif.

b. Mengapa scaling bisa efek terbalik di Gradient Boosting?

-Gradient Boosting berbasis tree, dan pohon tidak sensitif terhadap skala.

-Scaling malah bisa "meratakan" perbedaan kecil yang berguna untuk splitting di boosting.

4. Feature Interaction menyebabkan AUC naik

a. Mekanisme matematis:

-Perkalian dua fitur menciptakan fitur baru non-linear.

-Decision boundary bisa berubah dari linear ke non-linear, sehingga data fraud yang sebelumnya tumpang tindih bisa terpisah lebih baik.

b. Penyebab chi-square gagal

-Chi-square hanya deteksi hubungan linear antar fitur dengan label.

-Interaction nonlinear lewat perkalian tidak terdeteksi oleh uji chi-square.

c. Alternatif domain knowledge:

-Memakai correlation plots khusus untuk kombinasi fitur.

-PCA, t-SNE, atau UMAP untuk melihat struktur non-linear.

-Wawancara expert atau pakai business logic (contoh: jumlah transaksi × jam transaksi).

5. Oversampling sebelum split rawan Data Leakage

a.Alasan leakage terjadi

-Oversampling membuat data sintetis (contoh SMOTE) menggunakan seluruh dataset. Saat split, training dan testing bisa punya data mirip (overlapping). Akibatnya, validasi kelihatan bagus (AUC 0.95) padahal testing buruk (AUC 0.65).

b. Kenapa temporal split lebih aman di fraud detection?

-Fraud berubah seiring waktu.

-Melatih di masa lalu, menguji di masa depan itu layaknya simulasi dunia nyata.

c.Kenapa stratified sampling memperparah?

-Stratified sampling acak berdasarkan label.

-Kalau oversampling dulu, strata jadi tidak natural,itu membuat testing tidak mewakili kasus nyata.

d. Desain preprocessing yang benar:

-Split dulu dataset (train-test).

-Oversampling hanya di training set.

-Validasi pakai cross-validation di training.

-Evaluasi akhir hanya pakai test set tanpa sentuhan oversampling.