Komparasi Decision Tree dan Random Forest untuk Klasifikasi Kapal pada Citra Satelit di Wilayah Maritim

Nama Kelompok : The Return of Sultan

Anggota Kelompok

1. Sultan Muzahidin (1806169591)
2. Fauhan Handay Pugar (1806255355)
3. Rif'at Ahdi Ramadhani (1806269543)

Abstrak

Indonesia adalah negara kepulauan terbesar didunia yaitu dengan jumlah 17.499 pulau dari sabang sampai merauke. Sebagai negara kepulauan, Indonesia memiliki wilayah laut lebih luas dari daratan sehingga Indonesia disebut sebagai negara maritim. Salah satu permasalahan yang dihadapi di negara maritim seperti Indonesia adalah praktik *illegal fishing*. Untuk mencegah permalasahan *illegal fishing* tersebut diperlukan adanya pengawasan yang intensif. Namun, tantangan dalam pengawasan di wilayah maritim yang luas memerlukan sumber daya yang besar. Oleh karena itu diperlukan adanya sebuah sistem yang dapat membantu proses pengawasan wilayah laut di Indonesia. Metode *machine learning* mampu memberikan solusi dalam permasalahan pengawasan di wilayah laut Indonesia dengan cara mengidentifikasi sebaran kapal yang berada di wilayah laut Indonesia. Metode *machine learning* yang diterapkan untuk mengatasi permasalahan tersebut adalah metode *decision tree* dengan menerapkan *gradient boosting* dan metode *random forest* menggunakan data augmentasi dengan 10-fold. Pengujian metode yang diterapkan menggunakan metrik *precision*, *recall*, dan *f1-score*. Dalam eksperimen yang dilakukan, diperoleh metode *decision tree* dengan menerapkan *gradient boosting* memperoleh hasil terbaik dengan nilai *precision* sebesar 0,96, *recall* sebesar 0,96, dan *f1-score* sebesar 0,96. Berdasarkan eksperimen yang dilakukan, metode yang diterapkan mampu mengidentifikasi kapal dengan hasil yang baik.

Permasalahan

Indonesia adalah negara kepulauan terbesar didunia dimana indonesia memiliki 17.499 pulau dari sabang sampai merauke. Sebagai negara kepulauan, Indonesia memiliki wilayah laut lebih luas dari daratan karena itu Indonesia di sebut sebagai negara maritim. Total luas wilayah Indonesia adalah 7,81 juta km^2 yang terdiri dari 3,25 juta km^2 lautan, 2,01 juta km^2 daratan dan 2,55 juta km^2 Zona Ekonomi Eksklusif (ZEE) [1]. Wilayah laut Indonesia memiliki potensi sumber kekayaan sangat besar yaitu sebagai pemasok ikan terbesar didunia. Potensi ekonomi sumber daya kelautan dan perikanan yang dapat dimanfaatkan untuk mendorong pertumbuhan ekonomi diperkirakan mencapai USD 82 miliar per tahun [2]. Dengan banyaknya potensi yang dimiliki Indonesia memerlukan perlindungan dan pengelolaan sumber daya perairan yang baik.

Salah satu permasalahan yang dihadapi negara maritim seperti Indonesia adalah praktik *illegal fishing*. *Illegal fishing* merupakan aktivitas pencurian ikan yang dilakukan oleh kapal asing yang melewati wilayah yurisdiksi suatu negara secara ilegal. Praktik ini jelas telah sangat merugikan negara setiap tahunnya. Menurut Menteri Kelautan dan Perikanan, Susi Pudjiastuti, kerugian negara telah mencapai Rp 240 triliun [3]. Selain itu, praktik *illegal fishing* juga menyebabkan kerugian lainnya, yakni kerusakan ekosistem laut.

Mengacu pada Undang-Undang Nomor 31 tahun 2004 dan Undang-Undang Nomor 45 Tahun 2009, Pemerintah Indonesia telah melakukan kebijakan penanganan terhadap praktik *illegal fishing* dengan cara menerapkan kebijakan penenggelaman kapal yang melakukan tindak pidana tersebut. Hingga tahun 2018, sebanyak 488 kapal illegal fishing telah ditenggelamkan dalam penerapan kebijakan tersebut [4]. Untuk menerapkan kebijakan tersebut, tentunya diperlukan pengawasan secara intensif. Namun pengawasan secara intensif masih memiliki tantangan besar dalam hal usaha dan biaya yang diperlukan.

Peran *machine learning* diharapkan mampu memberikan solusi yang efisien dalam masalah pengawasan wilayah laut di Indonesia. Klasifikasi berperan untuk mengetahui pemetaan sebaran jumlah kapal di area tertentu. Identifikasi ini diperlukan agar pengawasan menjadi lebih terarah dan menjadi pondasi yang membantu dalam mengindentifikasi kapal yang terlibat dalam aktifitas *illegal fishing*. Oleh karena itu, pada penilitian ini kami membuat kerangka sistem machine learning untuk mengklasifikasikan objek kapal dan bukan kapal yang berada di suatu wilayah tertentu.

Dataset dan Feature

Sebagai representasi dari wilayah maritim kami menggunakan dataset yang diperoleh dari *Planet satellite imagery* yang berada pada area San Fransisco Bay dan San Pedro Bay sekitar di California. Dataset terdiri dari 4000 citra chip RGB, dimana setiap citra berukuran 80x80 piksel. Citra chip yang didapat dari visual frame *PlanetScope* secara penuh yang ter-ortoreksi ke ukuran piksel dengan jarak 3 meter. Data citra yang digunakan adalah png dimana penamaan gambarnya memiliki format khusus yaitu {label}_{scene_id}_{longitude}_{latitude}.png. Dataset ini juga menyediakan format teks JSON dengan nama shipsnet.json yang terdiri dari data, label, scene_ids dan daftar lokasi. Label terdiri nilai 1 dan 0, angka 1 merepresentasikan kelas "kapal" dan "bukan kapal". Kelas "kapal" (1000 citra) terdiri dari banyak bentuk kapal dengan berbagai ukuran dan bentuk. Sedangkan kelas "bukan kapal" (3000 citra) dimana sepertiganya adalah *random sampling* dari fitur tutupan lahan (*landcover features*) seperti air, vegetasi , tanah kosong, bangunan, dll. Sepertiga berikutnya adalah "kapal parsial" yang hanya berisi sebagaian kapal sehingga tidak memenuhi bagian kapal secara penuh. Sepertiga terakhir adalah gambar yang salah label oleh pembelajaran mesin, biasanya disebabkan oleh piksel yang cerah atau lain-lainnya. *Scene_id* adalah pengenal unik dari *PlanetScope* untuk setiap visual dari citra chip yang di ekstrak. *Longitude_latitude* merupakan koordinat dari citra pada titik tengah gambar dimana setiap nilainya dipisah dengan underscore. Setiap nilai piksel dari citra RGB berukuran 80x80 disimpan dalam bentuk list yang terdiri dari nilai integer 19200. Data pertama terdiri dari 6400 nilai pada channel R, selanjutnya 6400 nilai pada channel G, dan terakhir 6400 nilai pada channel B. Dataset tersedia pada situs Kaggle – *Ship in Satellite Imagery* [5].

Citra yang digunakan diubah menjadi *grayscale* karena fitur yang diperlukan tidak memerlukan fitur warna secara utuh. Pada data citra satelit yang diperoleh memiliki karakteristik warna yang tidak konsisten sehingga fitur warna tidak dapat merepresentasikan objek yang akan diidentifikasi sebagai kapal. Fitur yang digunakan dalam penelitian ini adalah menggunakan fitur *Histogram of Oriented Gradients* (HOG). Fitur HOG yang diekstrak berdasarkan nilai parameter orientation sebesar 16, pixel per cell sebesar 16x16, dan cell per block sebesar 4x4. Fitur ini digunakan sebegai representasi bentuk kapal yang akan diidentifikasi.

Metode

Decision tree menggunakan *gradient boosting*

Gradient Boosting Decision Tree (GBDT) merupakan metode pengembangan dari *decision tree* yang menerapkan strategi *ensemble boosting* berbasis gradient [6]. Secara umum, proses *fitting* dilakukan secara bertahap sampai dapat meminimalkan nilai loss dari nilai residual (selisih dari nilai pengamatan sebenarnya y dan nilai prediksi ŷ) yang ada. *Cost function* dari metode GBDT dapat dituliskan dalam persamaan (2) sebagai berikut:

$$L = \sum_{i}^{N} L\big(y_i, F(x_i)\big) + \sum_{i}^{N} \Omega(f_k)$$

di mana L merupakan *training loss* terhadap sampel yang dapat berupa nilai *absolute error*, *mean squared error*, dsb. Ω sebagai fungsi regularisasi yang memberikan *penalty* terhadap kompleksitas f_k.

XGBoost merupakan metode pengembangan dari model GDBT yang diterapkan dengan memperhatikan aspek kecepatan komputasi dan performa model. Jika dibandingkan dengan model GDBT, XGBoost memiliki beberapak keunggulan antara lain:

1. XGboost menerapkan term regularisasi ($\sum_{k=1}^{K} \Omega f_k$) pada cost function untuk mengatur tingkat kompleksitas model dan membantu model terhindar dari over-fitting
2. Sebagaimana Random forests, XGBoost dapat mengurangi over-fitting dan besarnya komputasi yang diperlukan dengan menerapkan column sampling.

Metode dasar GBDT hanya menggunakan optimasi dari turunan pertama sedangkan pada XGBoost diterapkan ekspansi Taylor orde kedua pada cost function, dengan menggunakan turunan pertama derivative (G_j) dan turunan kedua derivative (H_j).

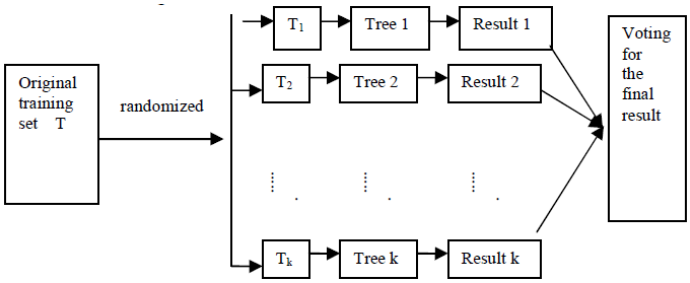
Random Forest

Random forest adalah kombinasi dari algoritma *machine learning.* Dimana kita mengkombinasikan serangkaian *tree classifier*, pada setiap *tree* dilakukan *voting* untuk mendapatkan kelas paling populer. Kemudian hasil dari kombinasi tersebut dilakukan *sorting*. *Random forest* memiliki akurasi yang tinggi, tahan terhadap noise dan juga tidak pernah mendapatkan *overfitting*.

Pada Breiman's RF model, setiap *tree* dilakukan *training* menggunakan *random variable*, dimana variabel acak pada *tree* di notasikan dengan Θ_k, antara dua variabel acak saling *independent and identically distributed*. Hasil klasifikasi h(x, Θ_k) dimana x adalah masukan sebuah vektor. Setelah k dijalankan kita akan mendapatkan urutan klasifikasi { $h_1(x), h_2(x),\ldots h_k(x)$}, dan kemudian setelah mendapatkan model sistem yang lebih dari satu klasifikasi, hasil akhirnya dilakukan majority vote, fungsi dari pemilihannya adalah sebagai berikut :

$$H(x) = \arg\max_{Y} \sum_{i=1}^{k} I\left(h_i(x) = Y\right) \tag{2}$$

H(x) adalah kombinasi klasifikasi model, h_i adalah *single decision tree model*, Y adalah *output variable*, I adalah *indicator function*. Gambar jelasnya sebagai berikut [7]:



Gambar 1 Skema Random Forest

Hasil Eksperimen

Eksperimen dilakukan dengan berbagai skenario pengujian, meliputi:

1. Decision Tree
2. Decision Tree (5-fold)
3. Decision Tree (10-fold)
4. Decision Tree (Tunning = Random)
5. Decision Tree (Tunning = Grid Search)
6. Decision Tree dengan data augmentasi
7. Decision Tree dengan data augmentasi (5-fold)
8. Decision Tree dengan data augmentasi (10-fold)
9. Decision Tree dengan data augmentasi (Tunning = Random)

10. Decision Tree dengan data augmentasi (Tunning = Grid search)
11. Decision Tree menggunakan Gradient Boosting
12. Decision Tree menggunakan Gradient Boosting (5-fold)
13. Decision Tree menggunakan Gradient Boosting (10-fold)
14. Decision Tree menggunakan Gradient Boosting data augmentasi
15. Decision Tree menggunakan Gradient Boosting (5-fold) data augmentasi
16. Decision Tree menggunakan Gradient Boosting (10-fold) data augmentasi
17. Random Forest
18. Random Forest (5-fold)
19. Random Forest (10-fold)
20. Random Forest (Tunning = Random)
21. Random Forest (Tunning = Grid Search)
22. Random Forest dengan data augmentasi
23. Random Forest dengan data augmentasi (5-fold)
24. Random Forest dengan data augmentasi (10-fold)

Sedangkan, *hyperparameter* untuk setiap eksperimen yang diuji dapat dilihat pada lampiran.

Dari ke-24 skenario diatas, hasil terbaik diperoleh dengan menggunakan metode *decision tree* menggunakan *gradient boosting* dan metode *random forest* menggunakan data augmentasi dengan 10-fold. Pada tabel dibawah adalah hasil terbaik dengan menggunakan metode *decision tree* menggunakan *gradient boosting* dengan 5-fold dan metode *random forest* menggunakan data augmentasi dengan 10-fold.

Tabel 1. Hasil ekperimen metode decision tree menggunakan gradient boosting dengan dan metode Random Forest menggunakan data augmentasi (10-fold)

Decision tree menggunakan *gradient boosting*					*Random Forest* dengan data augmentasi - 10 fold			
	precision	recall	f1-score	support	precision	recall	f1-score	support
0	0.98	0.98	0.98	604	0.96	0.99	0.98	4219
1	0.95	0.93	0.94	196	0.98	0.87	0.92	1381
micro avg	0.97	0.97	0.97	800	0.96	0.96	0.96	5600
macro avg	0.96	0.96	0.96	800	0.97	0.93	0.95	5600
weighted avg	0.97	0.97	0.97	800	0.96	0.96	0.96	5600

Dalam pengujian citra satelit untuk melakukan klasifikasi antara kapal dan bukan kapal diperoleh hasil deteksi menggunakan metode *decision tree* dengan *gradient boosting*. Hasil eksperimen lebih lengkap dapat dilihat pada lampiran.

Secara umum, metode ensemble memiliki keunggulan yaitu mampu meminimalkan bias dan varian dengan lebih baik jika dibandingkan metode decision tree biasa. Metode decision tree pada umumnya memiliki sifat varian tinggi. Selain itu decision tree juga memiliki kelemahan dalam menangani data berdimensi tinggi. Hal inilah yang menjadi dasar pengembangan decision tree dengan menggunakan teknik ensemble seperti bagging atau boosting. Meskipun metode bagging seperti random forest dapat mengatasi data dengan dimensi yang besar, namun metode ini masih memiliki kekurangan dalam melakukan generalisasi fitur dari data citra yang beragam sehingga kesulitan ketika

melakukan training data yang mengandung noise. Berbeda dengan strategi yang diterapkan pada metode random forest, metode gradient boosting menerapkan regularisasi sehingga dapat memperoleh model yang dapat memperoleh hasil yang lebih baik karena mampu menangani berbagai kondisi data citra.

Kontribusi

Pada final report ini kami mengimplementasikan algoritma *decision tree*, *random forest* dan *decision tree* menggunakan *gradient boostin*g dengan *stratified k-fold* dengan parameter pada tabel yang berada di lampiran. Masing-masing algoritma diukur perfomansinya menggunakan akurasi, presisi, recall, dan F1-score. Adapun pekerjaan spesifik yang dilakukan setiap anggota adalah sebagai berikut:

1. Sultan Muzahidin : Decision tree, decision tree (5-fold), decision tree (10-fold), decision tree (tuning = random), decision tree (tuning = grid search), decison tree data augmentasi, decision tree menggunakan data augmentasi dan 5-fold, decision tree menggunakan data augmetnasi dan 10-fold dengan evaluasi perfoma, final report.
2. Fauhan Handay Pugar : decision tree menggunakan data augmentasi (tuning=random), decision tree menggunakan data augmentasi (tuning=grid search), decision tree menggunakan gradient boosting, decision tree menggunakan gradient boosting dan 5-fold, decison tree menggunakan gradient boosting dan 10-fold, decision tree menggunakan gradient boosting dan data augmentasi, decision tree menggunakan gradient boosting dan data augmentasi (5-fold), decision tree menggunakan gradient boosting dan data augmentasi (10-fold) dengan evaluasi perfoma, final report, Banner.
3. Rif'at Ahdi Ramadhani : Random forest, random forest (5-fold), random forest (10-fold), random forest (tuning=random), random forest (tuning=grid search), random forest menggunakan data augmentasi, random forest menggunakan data augmentasi (5-fold), random forest menggunakan data augmentasi (10-fold) dengan evaluasi perfoma, final report.

Referensi

[1] B. P. S. INDONESIA, "STATISTIK SUMBER DAYA LAUT DAN PESISIR." 2018.

[2] S. K. R. INDONESIA, "Potensi Besar Perikanan Tangkap Indonesia." 2016.

[3] Detik, "Menteri Susi: Kerugian Akibat Illegal Fishing Rp 240 Triliun." 2014.

[4] Katadata, "Cek Data: Benarkah 488 Kapal Illegal Fishing Sudah Ditenggelamkan?" 2019.

[5] Kaggle, "Ships in Satellite Imagery," 2018. [Online]. Available: https://www.kaggle.com/rhammell/ships-in-satellite-imagery.

[6] Y. Xi, X. Zhuang, X. Wang, R. N. B, and G. Z. B, *Web Information Systems and Applications*, vol. 11242. Springer International Publishing, 2018.

[7] Y. Liu, Y. Wang, and J. Zhang, "New Machine Learning Algorithm: Random Forest," 2012, pp. 246–252.

Lampiran

Tabel

Tabel 2. Lini waktu penelitian

Rencana	Februari 2019	Maret 2019					April					Mei	
	5	1	2	3	4	5	1	2	3	4	5	1	2
Proposal													
Analisis Dataset													
Pre-processing													
Ekstraksi Fitur													
Progress 1													
Decision Tree													
Random Forest													
Progress 2													
Optimasi dan Evaluasi													
Pembuatan Laporan Akhir dan Poster													
Pengumpulan													

Keterangan :

Warna abu-abu: Aktifitas yang telah dilakukan

Warna Hijau: Akftitas yang akan dilakukan

Gambar hasil eksperimen

Berikut adalah hasil klasifikasi antara kapal dan bukan kapal pada citra satelit oleh *Planet satellite imagery* dan satelit Google maps di wilayah laut Indonesia

Decision tree menggunakan gradient boosting	Random forest menggunakan data augmentasi dan 10-fold

Gambar 2. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra pelabuhan makassar

Gambar 3. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra pelabuhan makassar

Gambar 4. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra lb-1

Gambar 5. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra lb-1

Gambar 6. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra lb-3

Gambar 7. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra lb-3

Gambar 8. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra sfbay-1

Gambar 9. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra sfbay-1

Gambar 10. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra pelabuhan trisakti

Gambar 11. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra pelabuhan trisakti

Gambar 12. Hasil deteksi dengan decision tree menggunakan gradient boosting pada citra pelabuhan sunda-kelapa jakarta utara

Gambar 13. Hasil deteksi dengan random forest menggunakan data augmentasi pada citra pelabuhan sunda-kelapa jakarta utara

Tabel Hasil Eksperimen

Tabel 3. Tabel kumpulan hyperparameter untuk metode decision tree, random forest, dan decision tree menggunakan gradient boosting

No.	Decision Tree	Random Forest	Decision Tree menggunakan gradient boosting
1	max_depth : None	n_estimator : 50	learning_rate : 0.3
	max_features : None	max_depth : None	min_split_loss : 0
	min_sample_leaf : 1	max_features : Auto	max_depth : 6
	min_sample_split : 2	min_sample_leaf : 1	min_child_weight : 1
	class_weight : None	min_sample_split : 2	
		class_weight : None	
2	Kfold : 5	Kfold : 5	Kfold : 5
	max_depth : None	n_estimator : 1000	learning_rate : 0.3
	max_features : None	max_depth : None	min_split_loss : 0
	min_sample_leaf : 1	max_features : Auto	max_depth : 6
	min_sample_split : 2	min_sample_leaf : 1	min_child_weight : 1
	class_weight : None	min_sample_split : 2	
		class_weight : None	
3	Kfold : 10	Kfold : 10	Kfold : 10
	max_depth : None	n_estimator : 1000	learning_rate : 0.3
	max_features : None	max_depth : None	min_split_loss : 0

	min_sample_leaf : 1	max_features : Auto	max_depth : 6
	min_sample_split : 2	min_sample_leaf : 1	min_child_weight : 1
	class_weight : None	min_sample_split : 2	
		class_weight : None	
4	Random Search	Random Search	
	n_estimator : [100, 2000]	n_estimator : [100, 2000]	
	max_depth : [10, 20, 30, 40, 50]	max_depth : [10, 20, 30, 40, 50]	
	max_features : [auto, sqrt]	max_features : [auto, sqrt]	
	min_sample_leaf : [0.1, 1.0]	min_sample_leaf : [0.1, 1.0]	
	min_sample_split : [0.1, 0.5]	min_sample_split : [0.1, 0.5]	
	class_weight : [balanced]	class_weight : [balanced]	
	n_iter : 100	n_iter : 100	
5	Grid Search	Grid Search	
	max_depth : [10, 20, 30, 40, 50]	max_depth : [10, 20, 30, 40, 50]	
	max_features : [auto, sqrt]	max_features : [auto, sqrt]	
	min_sample_leaf : [0.1, 1.0]	min_sample_leaf : [0.1, 1.0]	
	min_sample_split : [0.1, 0.5]	min_sample_split : [0.1, 0.5]	

Tabel 4. Hasil eksperimen decision tree

Decision Tree				
	precision	recall	f1-score	support
0	0.91	0.9	0.9	610
1	0.69	0.71	0.7	190
micro avg	0.85	0.85	0.85	800
macro avg	0.8	0.8	0.8	800
weighted avg	0.86	0.85	0.86	800

Tabel 5. Hasil eksperimen decision tree (5-fold)

Decision Tree 5 fold				
	precision	recall	f1-score	support
0	0.89	0.91	0.9	601
1	0.7	0.65	0.67	199
micro avg	0.84	0.84	0.84	800
macro avg	0.79	0.78	0.78	800
weighted avg	0.84	0.84	0.84	800

Tabel 6. Hasil eksperimen decision tree (10-fold)

Decision Tree 10-fold				
	precision	recall	f1-score	support
0	0.89	0.88	0.88	589
1	0.68	0.69	0.68	211
micro avg	0.83	0.83	0.83	800
macro avg	0.78	0.79	0.78	800
weighted avg	0.83	0.83	0.83	800

Tabel 7. Hasil eksperimen decision tree menggunakan random search tuning

Decision Tree - Random Search Tuning				
	precision	recall	f1-score	support
0	0.88	0.92	0.9	596
1	0.74	0.64	0.69	204
micro avg	0.85	0.85	0.85	800
macro avg	0.81	0.78	0.8	800
weighted avg	0.85	0.85	0.85	800

Tabel 8. Hasil eksperiment decision tree menggunakan grid search tuning

Decision Tree Grid Search Tuning				
	precision	recall	f1-score	support
0	0.92	0.83	0.87	596
1	0.62	0.78	0.69	204
micro avg	0.82	0.82	0.82	800
macro avg	0.77	0.81	0.78	800
weighted avg	0.84	0.82	0.83	800

Tabel 9. Hasil eksperimen decision tree menggunakan data augmentasi

Decision Tree Augmentasi Data				
	precision	recall	f1-score	support
0	0.93	0.93	0.93	4210
1	0.78	0.79	0.79	1390
micro avg	0.89	0.89	0.89	5600
macro avg	0.86	0.86	0.86	5600
weighted avg	0.89	0.89	0.89	5600

Tabel 10. Hasil eksperimen decision tree menggunakan data augmentasi dan 5-fold

Decision Tree Augmentasi Data 5 fold				
	precision	recall	f1-score	support
0	0.93	0.92	0.92	4146
1	0.78	0.8	0.79	1454
micro avg	0.89	0.89	0.89	5600
macro avg	0.85	0.86	0.86	5600
weighted avg	0.89	0.89	0.89	5600

Tabel 11. Hasil eksperimen decision tree menggunakan data augmentasi dan 10-fold

Decision Tree Augmentasi Data 10 fold

	precision	recall	f1-score	support
0	0.93	0.92	0.93	4168
1	0.78	0.81	0.79	1432
micro avg	0.89	0.89	0.89	5600
macro avg	0.86	0.86	0.86	5600
weighted avg	0.89	0.89	0.89	5600

Tabel 12. Hasil eksperiment menggunakan decision tree menggunakan data augmetnasi dan random tuning

Decision Tree Augmentasi Data Random Tunning				
	precision	recall	f1-score	support
0	0.91	0.69	0.79	4174
1	0.47	0.79	0.59	1426
micro avg	0.72	0.72	0.72	5600
macro avg	0.69	0.74	0.69	5600
weighted avg	0.79	0.72	0.74	5600

Tabel 13. Hasil eksperimen decision tree menggunakan data augmentasi dan grid search tuning

Decision Tree Augmentasi Data dan Grid Search Tunning				
	precision	recall	f1-score	support
0	0.91	0.7	0.79	4174
1	0.48	0.79	0.59	1426
micro avg	0.73	0.73	0.73	5600
macro avg	0.69	0.75	0.69	5600
weighted avg	0.8	0.73	0.74	5600

Tabel 14. Hasil eksperimen decision tree menggunakan gradient boosting

Decision tree menggunakan gradient boosting				
	precision	recall	f1-score	support
0	0.98	0.98	0.98	604
1	0.95	0.93	0.94	196
micro avg	0.97	0.97	0.97	800
macro avg	0.96	0.96	0.96	800
weighted avg	0.97	0.97	0.97	800

Tabel 15. Hasil eksperiment decision tree menggunakan gradient boosting dan 5-fold

Decision tree menggunakan gradient boosting – 5-fold				
	precision	recall	f1-score	support
0	0.98	0.99	0.99	600
1	0.97	0.94	0.95	200
micro avg	0.98	0.98	0.98	800

macro avg	0.97	0.96	0.97	800
weighted avg	0.98	0.98	0.98	800

Tabel 16. Hasil eksperimen decision tree menggunakan gradient boosting dan 10-fold

Decision tree menggunakan gradient boosting – 10-fold				
	precision	recall	f1-score	support
0	0.98	1	0.99	300
1	0.99	0.95	0.97	100
micro avg	0.98	0.98	0.98	400
macro avg	0.99	0.97	0.98	400
weighted avg	0.99	0.98	0.98	400

Tabel 17. Hasil eksperimen decision tree menggunakan gradient boosting dan data augmentasi

Decision tree menggunakan gradient boosting dan data augmentasi				
	precision	recall	f1-score	support
0	0.95	0.99	0.97	4250
1	0.95	0.82	0.88	1350
micro avg	0.95	0.95	0.95	5600
macro avg	0.95	0.91	0.92	5600
weighted avg	0.95	0.95	0.95	5600

Tabel 18. Hasil eksperiment decision tree menggunakan gradient boosting, data augmentasi dan 5-fold

Decision tree menggunakan gradient boosting, data augmentasi dan 5 fold				
	precision	recall	f1-score	support
0	0.95	0.99	0.97	4200
1	0.95	0.84	0.89	1400
micro avg	0.95	0.95	0.95	5600
macro avg	0.95	0.91	0.93	5600
weighted avg	0.95	0.95	0.95	5600

Tabel 19. Hasil eksperiment decision tree menggunakan gradient boosting, data augmentasi dan 10-fold

Decision tree menggunakan gradient boosting, data augmentasi dan 10-fold				
	precision	recall	f1-score	support
0	0.95	0.99	0.97	2100
1	0.96	0.84	0.89	700
micro avg	0.95	0.95	0.95	2800
macro avg	0.95	0.91	0.93	2800
weighted avg	0.95	0.95	0.95	2800

Tabel 20. Hasil eksperimen random forest

Random Forest				
	precision	recall	f1-score	support
0	0.9	0.91	0.91	605
1	0.71	0.7	0.7	195
micro avg	0.86	0.86	0.86	800
macro avg	0.81	0.8	0.81	800
weighted avg	0.86	0.86	0.86	800

Tabel 21. Hasil eksperimen random forest menggunakan 5-fold

Random Forest - 5-fold				
	precision	recall	f1-score	support
0	0.93	0.97	0.95	606
1	0.88	0.77	0.82	194
micro avg	0.92	0.92	0.92	800
macro avg	0.9	0.87	0.88	800
weighted avg	0.92	0.92	0.92	800

Tabel 22. Hasil eksperimen random forest menggunakan 10-fold

Random Forest - 10-fold				
	precision	recall	f1-score	support
0	0.93	0.96	0.95	615
1	0.86	0.77	0.81	185
micro avg	0.92	0.92	0.92	800
macro avg	0.9	0.87	0.88	800
weighted avg	0.92	0.92	0.92	800

Tabel 23. Hasil eksperimen menggunakan random tuning

Random Forest menggunakan random tuning				
	precision	recall	f1-score	support
0	0.94	0.79	0.86	596
1	0.58	0.84	0.68	204
micro avg	0.8	0.8	0.8	800
macro avg	0.76	0.82	0.77	800
weighted avg	0.84	0.8	0.81	800

Tabel 24. Hasil eksperimen menggunakan grid search tuning

Random Forest menggunakan grid search tuning				
	precision	recall	f1-score	support
0	0.93	0.79	0.86	596
1	0.58	0.84	0.69	204

micro avg	0.81	0.81	0.81	800
macro avg	0.76	0.82	0.77	800
weighted avg	0.84	0.81	0.81	800

Tabel 25. Hasil eksperiment random forest menggunakan data augmentasi

Random Forest menggunakan data augmentasi				
	precision	recall	f1-score	support
0	0.96	0.99	0.97	4204
1	0.98	0.86	0.92	1396
micro avg	0.96	0.96	0.96	5600
macro avg	0.97	0.93	0.94	5600
weighted avg	0.96	0.96	0.96	5600

Tabel 26. Hasil eksperimen random forest menggunakan data augmentasi dan 5 fold

Random Forest menggunakan data augmentasi dan 5 fold				
	precision	recall	f1-score	support
0	0.96	0.99	0.97	4184
1	0.98	0.86	0.92	1416
micro avg	0.96	0.96	0.96	5600
macro avg	0.97	0.93	0.95	5600
weighted avg	0.96	0.96	0.96	5600

Tabel 27. Hasil eksperiment random rorest menggunakan data augmentasi dan 10 fold

Random Forest menggunakan data augmentasi dan 10 fold				
	precision	recall	f1-score	support
0	0.96	0.99	0.98	4219
1	0.98	0.87	0.92	1381
micro avg	0.96	0.96	0.96	5600
macro avg	0.97	0.93	0.95	5600
weighted avg	0.96	0.96	0.96	5600