 ISSN:2302-8149
e-ISSN:2540-9719

Hyperparameter Tuning pada Algoritma Klasifikasi dengan Grid Search

Hyperparameter Tuning on Classification Algorithm with Grid Search

[1]Wahyu Nugraha*, [2]Agung Sasongko
Sistem Informasi, Fakultas Teknik dan Informatika, Universitas Bina Sarana Informatika
Jl. Abdul Rahman Saleh No.18, Kec. Pontianak Tenggara, Kota Pontianak, Indonesia
*e-mail: wahyu.whn@bsi.ac.id

(***received*:**29 November 2021 ***revised*:**19 Februari 2022 ,***accepted*:** 3 Maret 2022)

Abstrak

Saat ini algoritma *machine learning* terus dikembangkan untuk bisa dilakukan optimasi dengan berbagai metode agar menghasilkan model dengan performa terbaik. Salah satu cara optimasi dengan menerapkan *tuning hyperparameter*. Pada *Supervised learning* atau klasifikasi sebagian besar algoritmanya memiliki *hyperparameter*. *Tuning hyperparameter* merupakan arsitektur dari *deep learning* untuk meningkatkan performa dari model prediksi. Metodologi *hyperparameter* yang populer diantaranya adalah *Grid Search*. *Grid Search* menggunakan *Cross Validation* memberikan kemudahan dalam menguji coba setiap parameter model tanpa harus melakukan validasi manual satu persatu. Pada penelitian ini akan menggunakan metode dalam optimasi *hyperparameter* yaitu *Grid Search*. Tujuan dari penelitian ini ingin mengetahui optimasi terbaik dari *hyperparameter* terhadap 7 algoritma klasifikasi *machine learning.* Validasi terhadap hasil eksperimen menggunakan metrik pengukuran *Mean Cross Validation*. Dari hasil eksperimen menunjukkan bahwa model *XGBoost* memperoleh nilai terbaik sedangkan *Decision tree* memiliki nilai terendah.

Kata kunci: Mesin Pembelajaran, *Hyperparameter*, *Grid Search*, Klasifikasi.

Abstract

Currently, machine learning algorithms continue to be developed to perform optimization with various methods to produce the best-performing model. In Supervised learning or classification, most of the algorithms have hyperparameters. Tuning hyperparameter is an architecture of deep learning to improve the performance of predictive models. One of the popular hyperparameter methodologies is Grid Search. Grid Search using Cross Validation provides convenience in testing each model parameter without having to do manual validation one by one. In this study, we will use a method in hyperparameter optimization, namely Grid Search. The purpose of this study is to find out the best optimization of hyperparameters against 7 machine learning classification algorithms. Validation of experimental results using the Mean Cross Validation. The experimental results show that the XGBoost model gets the best value while the Decision tree has the lowest value.

Keywords: *Machine Learning, Hyperparameter, Grid Search, Classification.*

1 Pendahuluan

Saat ini *machine learning* menjadi topik pembahasan terhangat karena dengan data sebuah model yang telah diprogram secara eksplisit mampu untuk belajar dan membangun model prediksi secara otomatis [1]. Algoritma *machine learning* menggunakan berbagai metode statistik, probabilistik dan optimisasi untuk dipelajari dari pengalaman masa lampau kemudian digunakan untuk mendeteksi pola yang berguna dari kumpulan data yang tidak terstruktur dan kompleks [2]. Algoritma tersebut secara luas diaplikasikan diberbagai bidang seperti kategorisasi teks otomatis, *network intrusion detection*, penyaringan *email* sampah, deteksi penipuan kartu kredit, deteksi perilaku membeli dari pelanggan, mengoptimalkan proses manufaktur, serta bisa digunakan untuk mendeteksi penyakit tertentu [3]. Sebagian besar aplikasi diimplementasikan menggunakan algoritma *machine learning supervised*

variants dari pada algoritma *machine learning unsupervised* maupun *semi-supervised* [3]. *Machine Learning* dibagi menjadi tiga pendekatan yaitu *supervised, unsupervised dan semi-supervised learning* [4]. *Supervised learning* juga disebut dengan klasifikasi (*classification*) atau *inductive learning* dimana telah digunakan hampir disetiap domain dan mengalami kesuksesan dalam penerapannya [5].

Proses *machine learning* khususnya klasifikasi pada umumnya meliputi uji coba berbagai model terhadap dataset setelah itu memilih model dengan performa terbaik. Model dengan hasil prediksi yang akurat bisa dilakukan dengan berbagai macam cara salah satunya dengan *hyperparameter tuning* yang merupakan arsitektur dari *deep learning* untuk meningkatkan performa dari model prediksi [6]. *Hyperparameter tuning* mempunyai peran yang sangat penting dalam mengoptimalkan kinerja dari algoritma *machine learning* [7]. Pemahaman mengenai bagaimana *hyperparameter* berinteraksi dengan performa dari model masih terus dikembangkan oleh para peneliti untuk menyempurnakan *fine-tune* pada *hyperparameter*. Salah satunya dilakukan dengan pendekatan untuk *tuning deep neural networks* yang menggunakan subset data untuk melatih model sebelumnya [8]. Selain itu ada juga yang menyajikan algoritma pemilihan *hyperparameter* yang menggabungkan pengetahuan yang dipelajari dari eksperimen sebelumnya menggunakan teknik optimasi dan peringkat berbasis pengganti [9]. *Random search* merupakan metode yang cukup populer sebagai alternatif dari *Grid Search* [7]. Metode ini diklaim lebih efisien dibandingkan *Grid Search*. Namun, tidak dapat diandalkan untuk *tuning hyperparameter deep belief networks* (DBNs) [10].

Pada sebagian besar algoritma *machine learning* memiliki *hyperparameter* dan kinerja prediktif sangat dipengaruhi oleh nilai *hyperparameter* yang digunakan untuk melatihnya. Contohnya *artificial neural network* membutuhkan penentuan jumlah *hidden layer*, *node* dan banyak lagi parameter lain yang terkait dengan proses *fitting* model [7]. Namun, *tuning hyperparameter* ini dapat memakan biaya komputasi yang tinggi, terutama pada kumpulan data yang lebih besar, sementara pengaturan yang disetel (*tuned*) tidak selalu secara signifikan lebih baik dari nilai *default* [11]. Belum ada konsensus yang jelas berkenaan dengan *tuning hyperparameter*. Metodologi yang populer diantaranya adalah *Grid Search* [10]. Walaupun metode ini terkadang tidak efisien bahkan tidak layak untuk digunakan [7]. Metode *Grid Search* masih menjadi solusi umum daripada harus mengubah satu *hyperparameter* pada satu waktu dan mengukur pengaruhnya terhadap kinerja model, justru hal tersebut akan menjadi tidak efisien dan tidak menjamin hasil yang optimal karena mengabaikan interaksi antara *hyperparameter* [7].

Pada penelitian ini peneliti akan menggunakan metode yang paling banyak digunakan dalam optimasi *hyperparameter* yaitu *Grid Search*. Tujuan dari penelitian ini ingin mengetahui optimasi terbaik dari *hyperparameter* terhadap beberapa algoritma *machine learning* menggunakan metode *Grid Search*. Metode ini akan melakukan uji coba combinasi dan validasi satu persatu kemudian memilih kombinasi yang menghasilkan performa model terbaik untuk prediksi. *Grid search cross validation* akan melakukan validasi dari setiap model kombinasi dan *hyperparameter* secara otomatis. Namun, kelemahan dari *grid search* ialah proses *tuning* yang memakan waktu ketika *hyperparameter* ditambahkan, karena jumlah kombinasi parameter meningkat secara eksponensial [6]. Uji coba eksperimental penelitian ini menggunakan *dataset* UCI *Pima Indian Diabetes* yang diperoleh dari *repositori University of California/Irvine* (UCI). *Dataset* ini merupakan salah satu *dataset* yang cukup banyak digunakan dalam penelitian *machine learning* karena diabetes menjadi masalah utama baik di negara maju maupun berkembang dan merupakan penyakit kronis yang tidak dapat disembuhkan [12]. Model pembelajaran menggunakan algoritma *supervised learning* seperti *Support Vector Machine* (SVM), *Random Forest, Logistic Regression, Gaussian Naive Bayes, Decision Tree, XGBoost, K-Nearest Neighbors* (k-NN). Nilai untuk menentukan *hyperparameter* terbaik dari *Grid Search* pada masing-masing model algoritma klasifikasi menggunakan metrik pengukuran *Mean Cross Validation*. Hasil eksperimen menunjukkan bahwa *hyperparameter* pada algoritma *XGBoost* menghasilkan *score* terbaik dari semua algoritma klasifikasi dengan nilai *score* 0,772.

2 Tinjauan Literatur

Penelitian yang dilakukan oleh Kunang [13] mengenai sistem deteksi intrusi (*Intrusion Detection System*) untuk mengurangi ancaman serangan pada jaringan. Penelitian ini memberikan solusi alternatif untuk model struktur *deep learning* melalui proses optimasi *hyperparameter* otomatis yang

 ISSN:2302-8149
e-ISSN:2540-9719

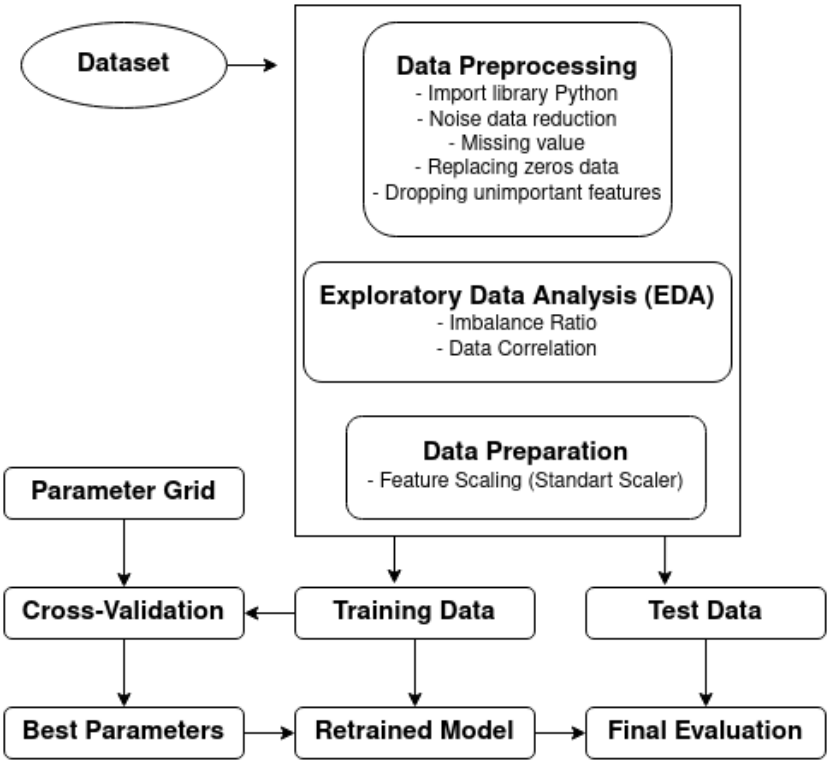
menggabungkan *grid search* dan *random search*. Proses optimasi *hyperparameter* otomatis membantu menentukan nilai *hyperparameter* dan konfigurasi *hyperparameter* kategoris terbaik untuk meningkatkan kinerja deteksi.

Penelitian yang dilakukan oleh Lujan-Moreno [7] menyatakan bahwa Sebagian besar algoritma *machine learning* memiliki *hyperparameter*. Metodologi yang paling populer namun memiliki kekurangan karena kurang efisien adalah *grid search*. Beberapa peneliti lain menggunakan cara lain yaitu mengubah satu *hyperparameter* pada satu waktu dan mengukur pengaruhnya terhadap kinerja model. Lujan-Moreno mengusulkan untuk menggunakan metodologi *Design of Experiments* (DOE) (desain faktorial) untuk penyaringan dan *Response Surface Methodology* (RSM) untuk *tuning hyperparameter*. Model disajikan menggunakan algoritma *random forest* menggunakan *dataset public*. Hasil yang diperoleh dari model tersebut adalah menghasilkan lebih sedikit *training runs* dan pemilihan parameter menjadi lebih baik.

Pada Penelitian ini peneliti mencoba melakukan optimasi model *machine learning* menggunakan *hyperparameter* dengan metode *grid search*. Walaupun pada beberapa kasus metode ini masih memiliki kekurangan. Namun, masih menjadi pilihan yang masih banyak digunakan dan cukup berhasil dalam mencari nilai *hyperparameter* dengan hasil yang cukup memuaskan. Model yang diuji menggunakan *hyperparamer* terdiri dari 7 model dengan algoritma klasifikasi yaitu *Support Vector Machine* (SVM), *Random Forest*, *Logistic Regression*, *Gaussian Naive Bayes*, *Decision Tree*, *XGBoost*, *K-Nearest Neighbors* (k-NN). *Dataset* menggunakan *dataset* UCI *Pima Indian Diabetes* yang diperoleh dari *repositori University of California/Irvine* (UCI). Sedangkan evaluasi hasil untuk model terbaik terhadap *hyperparameter* menggunakan metrik pengukuran *Mean Cross Validation*.

3 Metode Penelitian

Pada penelitian ini optimasi model dengan *hyperparameter* akan menggunakan metode *grid search*. Gambar 1 Menunjukkan tahapan eksperimen yang dilakukan yaitu dimulai dari *import dataset*, *data preprocessing*, *Exploratory Data Analysis* (EDA), *Data preparation* dan model *fitting* dengan *hyperparameter*.



GGambar 1: Model Tahapan Penelitian

 ISSN:2302-8149
e-ISSN:2540-9719

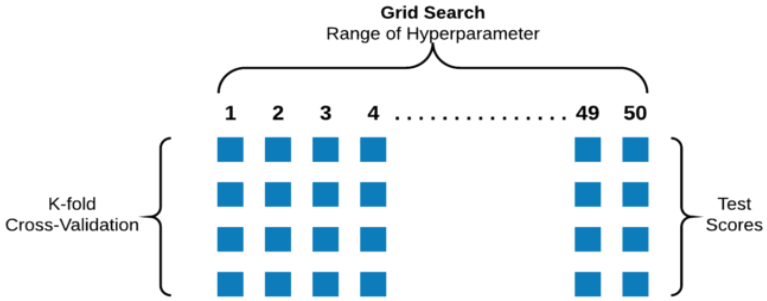
A. Pengumpulan *Data*

Dataset yang digunakan pada penelitian ini merupakan *dataset* yang berkaitan dengan penyakit diabetes. *Dataset* yang digunakan berasal dari *Kaggle dataset repository* (UCI *Pima Indians Diabetes Database*) yang diunduh dari *https://www.kaggle.com/uciml/pima-indians-diabetes-database*. *Dataset* ini memiliki 9 attribut dengan total sampel valid sebanyak 768 sampel. Nilai attribut *dataset* ini berasal dari semua wanita berusia minimal 21 tahun [14]. *Dataset* terdiri dari 8 variabel prediktor medis dan satu variabel target yaitu *Outcome*. Penelitian lain yang telah menggunakan dataset *Pima Indians* ini diantaranya dilakukan oleh Kumar dengan penelitian mengenai prediksi penyakit diabetes melitus menggunakan *Deep Neural Networks classifier* [15].

B. *Hyperparameter Tuning*

Hyperparameter tuning memiliki peran yang sangat penting dalam mengoptimalkan kinerja algoritma *machine learning* (ML) apa pun [7]. Nilai dari *hyperparameter* tidak dapat ditentukan dari data dan selalu kita ambil *as given* saat pendefinisian model dengan kata lain nilai *hyperparameter* harus ditetapkan sebelum sebuah model menjalani proses pembelajarannya. *Hyperparameter* merupakan variabel yang memengaruhi *output* dari sebuah model. Misalnya contoh pada model *k-nearest neighbours* kita mengenal *hyperparameter k* = (*number of nearest neighbors*), *k*-NN dengan nilai *k* = 1 dan *k* = 5 kemungkinan besar memberikan *output* yang berbeda meski diberikan input yang sama.

Penelitian ini menggunakan teknik *Grid Search* untuk proses menemukan nilai *hyperparameter* yang optimal pada model. *Grid Search Cross Validation* adalah istilah yang digunakan untuk merujuk teknik *Grid Search* dan *Cross-Validation* yaitu metode pemilihan kombinasi model dan *hyperparameter* dengan cara menguji coba satu persatu kombinasi dan melakukan validasi untuk setiap kombinasi [16]. Gambar 2 merupakan contoh ilustrasi proses *tuning hyperparameter* pada rentang 1-50 mengguna *Grid Search Cross Validation*.



Gambar 2. Ilustrasi *Grid Search Cross Validation*

4 Hasil dan Pembahasan

Eksperimen untuk mencapai hasil dari penelitian ini terdiri dari beberapa tahap diantaranya data *preprocessing*, *exploratory data analysis* (EDA), *data preparation*, model *fitting* (pengujian beberapa model terhadap *hyperparameter*) dan evaluasi hasil untuk menentukan *hyperparameter* terbaik dari seluruh model.

A. *Dataset*

Dataset diperoleh dari *Kaggle dataset repository* memiliki bentuk format CSV (*a comma-separated values*). *Dataset* terdiri dari beberapa variabel prediktor medis dan satu variabel target (*Outcome*). Variabel prediktor meliputi jumlah kehamilan yang dialami pasien, BMI, kadar insulin, usia, dan sebagainya. Berikut ini adalah contoh *overview* data primer diabetes yang dapat dilihat pada Tabel 1.

 ISSN:2302-8149
e-ISSN:2540-9719

Tabel 1. *Sample Dataset Diabetes*

No	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.63	50	1
1	1	85	66	29	0	26.6	0.35	31	0
2	8	183	64	0	0	23.3	0.67	32	1
3	1	89	66	23	94	28.1	0.17	21	0
4	0	137	40	35	168	43.1	2.29	33	1
5	5	116	74	0	0	25.6	0.2	30	0
6	3	78	50	32	88	31	0.25	26	1
7	10	115	0	0	0	35.3	0.13	29	0
8	2	197	70	45	543	30.5	0.16	53	1
9	8	125	96	0	0	0	0.23	54	1

B. *Data preprocessing*

Langkah pertama adalah mengimpor beberapa *library* yang diperlukan dalam pra-pemrosesan data. Eksperimen ini menggunakan *python3 language* sehingga *library* yang digunakan diantaranya *numpy*, *pandas*, *matplotlib* dan *seaborn*. Langkah selanjutnya adalah memuat *dataset* CSV *format* menggunakan *library pandas*. Setelah *dataset* dimuat, kita harus memeriksanya dan mencari *noise* dengan membuat *feature* matrix X dan *vector* observasi Y terhadap X. Langkah selanjutnya menemukan dan mengatasi nilai yang hilang (*missing value*) karena dapat menyebabkan masalah pada saat pelatihan. Gambar 3 menunjukkan detail statistik data seperti *percentile*, *mean*, std dan lainnya dari data *frame*. Dari Gambar 2 dapat dilihat bahwa beberapa variabel (*Glucose*, *Blood pressure*, *Skin thickness*, *Insulin* dan BMI) memiliki nilai *min* 0 padahal seharusnya tidak boleh 0. Sehingga nilai 0 harus digantikan dengan nilai statistik yang sesuai dengan distribusinya.

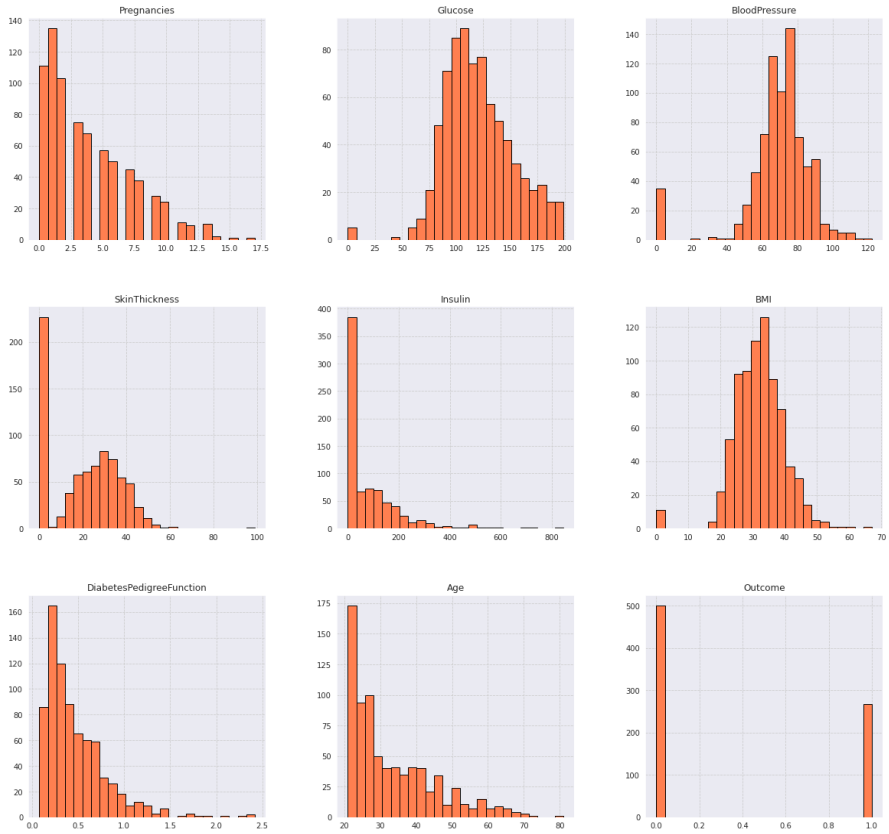
```
df.describe().transpose()
```

	count	mean	std	min	25%	50%	
Pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.
Glucose	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.
BloodPressure	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.
SkinThickness	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.
Insulin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.
BMI	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.
Age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.
Outcome	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.

Gambar 3. Detail Statistik *Data*

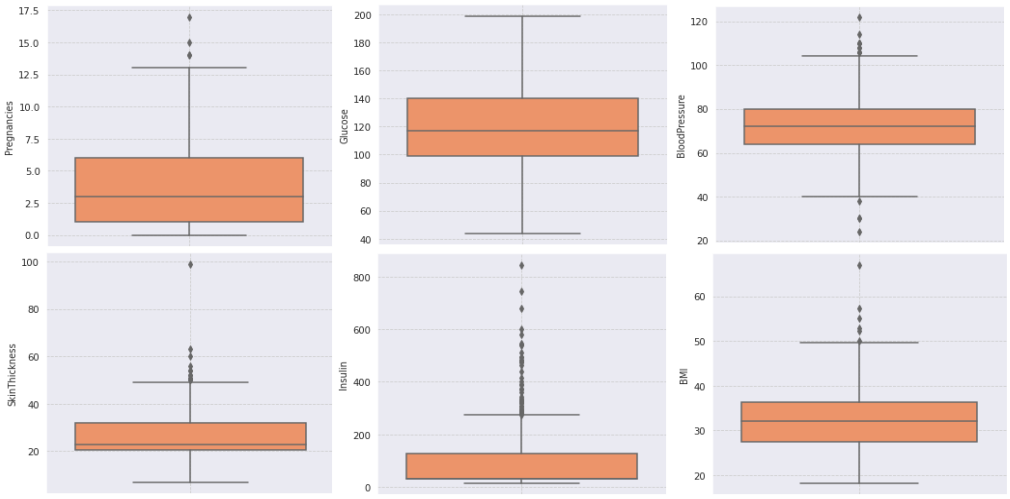
Pada Gambar 4 dapat dilihat distribusi data dengan jelas menggunakan *Histogram*. Dari Gambar 4 terlihat bahwa *bar* 0 pada *Glucose*, *Blood pressure*, *Skin thickness*, *Insulin* dan BM memiliki sifat berbeda dengan nilai rata-rata dari distribusinya. Jadi nilai 0 tersebut akan digantikan dengan nilai *mean* atau *median* distribusinya.

ISSN:2302-8149
e-ISSN:2540-9719



Gambar 4. ***Data Distributed in Histogram***

Features DiabetesPedigreeFunction memiliki *variance* yang sangat rendah karena hampir konstan dan memberikan sedikit informasi sehingga kolom tersebut akan dihapus (*dropping*) [17]. Langkah selanjutnya adalah mencari *outlier* yang ada di dalam data dan menghilangkannya. Gambar 5 memperlihatkan bahwa *Pregnancies*, *BloodPressure*, *SkinThickness*, *Insulin* dan BMI terdapat *outlier*.

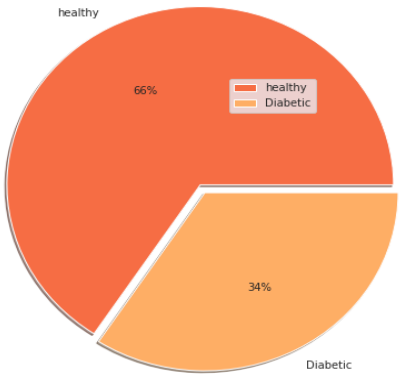


Gambar 5. ***Outlier Detection*** **Menggunakan** ***Boxplots***

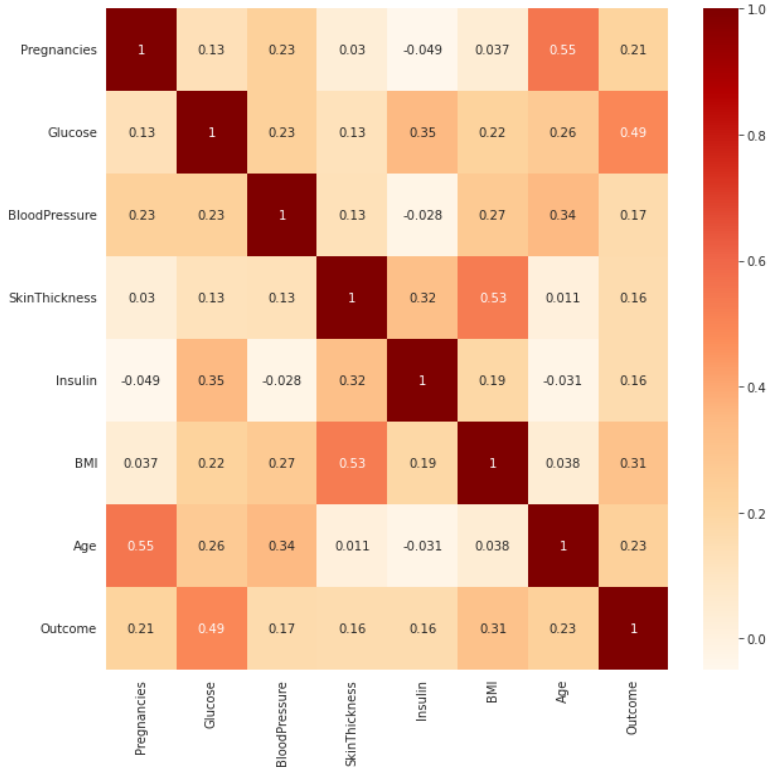
 ISSN:2302-8149
e-ISSN:2540-9719

C. *Exploratory Data Analysis* (EDA)

Analisis data eksplorasi pada tahap ini dilakukan untuk menemukan jawaban atas pertanyaan seperti rasio dari *imbalance class* dan melihat hubungan dua atribut data, *feature* yang bekaitan erat dengan penyakit diabetes. Gambar 6 Menggambarkan persentase *imbalance data* 66% *healthy* dan 34% mengalami diabetes. Sedangkan Gambar 7 menggambarkan hubungan korelasi antar variabel yang digambarkan dengan *seaborn correlation heatmap* menggunakan *python*.



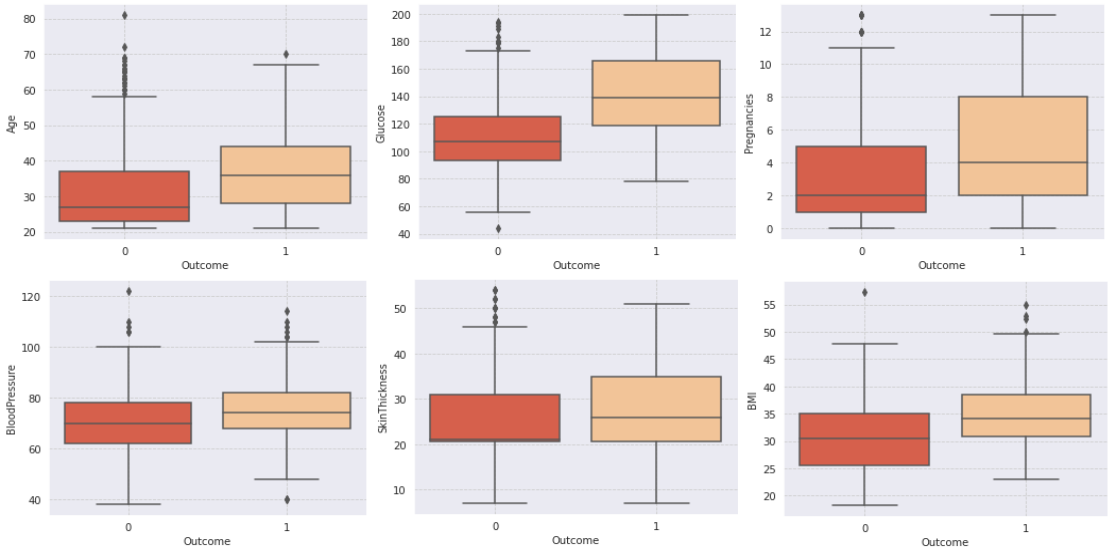
Gambar 6. *Imbalance Dataset*



Gambar 7. *Correlation Heatmap Dataset*

Dari Gambar 7 dapat dilihat korelasi setiap variabel terhadap target dimana *Glucose* menjadi variabel yang memiliki korelasi tertinggi terhadap *Outcome* dengan nilai 0,49 diikuti dengan BMI dengan nilai 0,31. Agar lebih jelas korelasi data akan digambarkan menggunakan *side-by-side box plots*. *Boxplots* sangat berguna ketika ingin membandingkan distribusi variabel kuantitatif untuk tingkat beberapa variabel kualitatif [18].

 ISSN:2302-8149
e-ISSN:2540-9719



Gambar 8. *Correlation Data* Menggunakan S*ide-by-side Box Plots*

Pada Gambar 8 dapat disimpulkan seperti yang telah diamati sebelumnya pada Gambar 7 bahwa *Glucose* tinggi dapat menyebabkan diabetes, usia/*age* yang lebih tua lebih mungkin terkena diabetes, wanita dengan *high pregnancy* sebagian besar adalah penderita diabetes, seseorang dengan *thick tricep* dapat menjadi penderita diabetes dan BMI tinggi menyebabkan diabetes

D. *Data Preparation*

Nilai dari data mentah (*raw data*) sangat bervariasi dan dapat mengakibatkan pelatihan model menjadi bias atau mungkin pada akhirnya dapat meningkatkan biaya komputasi. Jadi, sangat penting untuk menormalkan data tersebut. Salah satu caranya menggunakan *feature scaling* yaitu teknik yang digunakan untuk membawa nilai data dalam rentang yang lebih pendek. *Feature scaling* dalam pembelajaran mesin adalah salah satu langkah paling penting sebelum membuat model pembelajaran mesin [19]. Eksperimen ini peneliti menggunakan *Standard Scaler* yang berasal dari *library sklearn* yang ada di *python*.

E. *Model Fitting*

Pengujian terhadap model klasifikasi pada eksperimen ini akan menggunakan 7 algoritma klasifikasi yaitu *Support Vector Machine* (SVM), *Random Forest*, *Logistic Regression*, *Gaussian Naive Bayes*, *Decision Tree*, *XGBoost*, *K-Nearest Neighbors* (k-NN). Dari seluruh model yang dibangun akan dicari *hyperparameter* terbaik menggunakan metode *Grid Search* dengan *Cross Validation*. Untuk menentukan model terbaik maka pengukuran terhadap model algoritma klasifikasi menggunakan metrik pengukuran *Mean Cross Validation*. Dari hasil eksperimen dapat dilihat pada Gambar 9 dimana dari hasil eksekusi best *hyperparameter* telah didapatkan.

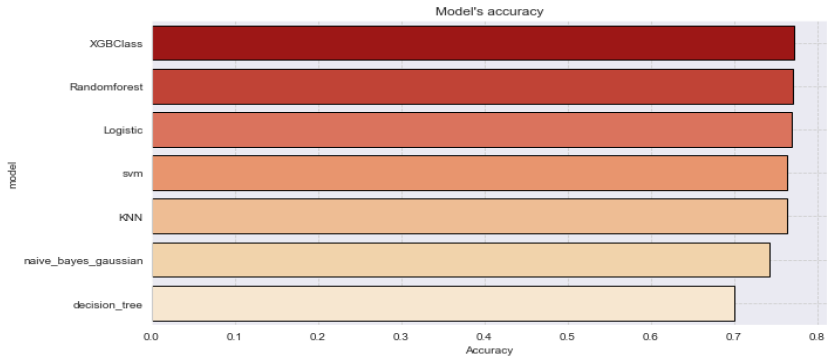
 ISSN:2302-8149
e-ISSN:2540-9719

	model	best_score	best_params
0	svm	0.763373	{'C': 1, 'kernel': 'linear'}
1	Randomforest	0.771190	{'criterion': 'entropy', 'n_estimators': 100}
2	Logistic	0.769961	{'C': 1}
3	naive_bayes_gaussian	0.743529	{}
4	decision_tree	0.701255	{'criterion': 'gini'}
5	XGBClass	0.772706	{'alpha': 0, 'eta': 0.1, 'lambda': 1, 'n_estim...
6	KNN	0.763294	{'metric': 'manhattan', 'n_neighbors': 13, 'we...

Gambar 9. Hasil Eksekusi Model

	model	best_score
5	XGBClass	0.772706
1	Randomforest	0.771190
2	Logistic	0.769961
0	svm	0.763373
6	KNN	0.763294
3	naive_bayes_gaussian	0.743529
4	decision_tree	0.701255

Gambar 10. *Best Score Model*



Gambar 11. *Best Score* Dalam Bentuk Grafis

Gambar 10 menunjukkan bahwa dari model yang telah diujikan model *XGBoost* memperoleh nilai *mean cross validation* sebesar 0,772 diikuti oleh *Random Forest* sebesar 0,771. Selanjutnya model *Logistic Regression* dengan nilai 0,769 kemudian model *Support Vector Machine* (SVM) dan *K-Nearest Neighbors* (k-NN) memperoleh nilai hampir sama sebesar 0,763. Model dengan urutan terbawah adalah *Gaussian Naive Bayes* dengan nilai sebesar 0,743 dan paling terakhir adalah model *Decision Tree* sebesar 0,701. Jika disajikan dalam bentuk grafis seperti Gambar 11 maka, dapat disimpukan bahwa tidak ada perbedaan yang signifikan antara model yang diuji. Model yang terlihat cukup signifikan hanya pada 2 model dengan nilai terendah yaitu *Gaussian Naive Bayes* dan *Decision Tree*.

 ISSN:2302-8149
e-ISSN:2540-9719

5 Kesimpulan

Berdasarkan penelitian yang dilakukan menggunakan *dataset* UCI *Pima Indian Diabetes* dapat disimpulkan bahwa optimasi dengan *tuning hyperparameter* menggunakan *Grid Search* terhadap model *machine learning* membuat proses pemilihan model menjadi lebih mudah. *Grid Search* menggunakan *Cross Validation* memberikan kemudahan dalam menguji coba setiap parameter model tanpa harus melakukan validasi manual satu persatu. Jika disandingkan dengan pemahaman dan intuisi yang baik akan memberikan hasil prediksi yang akurat dan optimal. Hasil eksperimen menunjukkan bahwa model *XGBoost* memperoleh nilai terbaik yaitu sebesar 0,772 sedangkan *Decision tree* memiliki nilai terendah yaitu 0,701. Untuk Penelitian selanjutnya seperti yang kita ketahui bahwa *grid search* memiliki kelemahan dalam proses *tuning* yang memakan waktu ketika *hyperparameter* ditambahkan, karena jumlah kombinasi parameter meningkat secara eksponensial [6]. Salah satu cara untuk mengatasinya adalah dengan menggabungkan atau mengkombinasikan *grid search* dengan *random search*.

Referensi

[1] P. C. Sen, M. Hajra, and M. Ghosh, "Supervised Classification Algorithms in Machine Learning: A Survey and Review," in *Advances in Intelligent Systems and Computing*, 2020, vol. 937, pp. 99–111, doi: 10.1007/978-981-13-7403-6_11.

[2] T. M. Mitchell, *Machine Learning*. Boston: WCB/McGraw-Hill, 1997.

[3] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," *BMC Med. Inform. Decis. Mak.*, vol. 19, no. 1, pp. 1–16, 2019, doi: 10.1186/s12911-019-1004-8.

[4] R. Saravanan and P. Sujatha, "Algorithms : A Perspective of Supervised Learning Approaches in Data Classification," *Proc. Second Int. Conf. Intell. Comput. Control Syst. (ICICCS 2018)*, pp. 945–949, 2018.

[5] B. Liu, *Web data mining: exploring hyperlinks, contents, and usage data*, Second. Berlin: Springer, 2011.

[6] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[7] G. A. Lujan-Moreno, P. R. Howard, O. G. Rojas, and D. C. Montgomery, "Design of experiments and response surface methodology to tune machine learning hyperparameters, with a random forest case-study," *Expert Syst. Appl.*, vol. 109, pp. 195–205, 2018, doi: 10.1016/j.eswa.2018.05.024.

[8] J. P. Lalor, H. Wu, and H. Yu, "CIFT: Crowd-Informed Fine-Tuning to Improve Machine Learning Ability," *arXiv Comput. Lang.*, vol. 6, no. February, 2017, [Online]. Available: http://arxiv.org/abs/1702.08563.

[9] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag, "Collaborative hyperparameter tuning," *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, no. 2, pp. 199–207, 2013.

[10] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[11] R. G. Mantovani, A. L. D. Rossi, E. Alcobaça, J. Vanschoren, and A. C. P. L. F. de Carvalho, "A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers," *Inf. Sci. (Ny).*, vol. 501, pp. 193–221, 2019, doi: 10.1016/j.ins.2019.06.005.

[12] F. Alaa Khaleel and A. M. Al-Bakry, "Diagnosis of diabetes using machine learning algorithms," *Mater. Today Proc.*, no. xxxx, 2021, doi: 10.1016/j.matpr.2021.07.196.

[13] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, "Attack classification of an intrusion detection system using deep learning and hyperparameter optimization," *J. Inf. Secur. Appl.*, vol. 58, no. March, p. 102804, 2021, doi: 10.1016/j.jisa.2021.102804.

[14] Y. Hayashi and S. Yukita, "Rule extraction using Recursive-Rule extraction algorithm with J48graft combined with sampling selection techniques for the diagnosis of type 2 diabetes mellitus in the Pima Indian dataset," *Informatics Med. Unlocked*, vol. 2, pp. 92–104, 2016, doi: 10.1016/j.imu.2016.02.001.

ISSN:2302-8149
e-ISSN:2540-9719

[15] B. P. Manoj Kumar, S. R. Perumal, and N. R. K, "Type 2: Diabetes mellitus prediction using Deep Neural Networks classifier," *Int. J. Cogn. Comput. Eng.*, vol. 1, pp. 55–61, 2020, doi: 10.1016/j.ijcce.2020.10.002.

[16] S. Patel, "Fundamental concepts for Model Selection and Model Evaluation — Part2," *Medium*, 2020. https://medium.com/analytics-vidhya/fundamental-concepts-for-model-selection-and-model-evaluation-part2-e72b384f8ab6 (accessed Nov. 26, 2021).

[17] B. Arup, "T103: Filter method-Feature selection techniques in machine learning," *upskillpoint*, 2020. https://www.upskillpoint.com/machine learning/2020/03/11/feature-selection-using-filter-method/ (accessed Nov. 27, 2021).

[18] P. Kampstra, "Beanplot: A Boxplot Alternative for Visual Comparison of Distributions," *J. Stat. Softw.*, vol. 28, no. November, pp. 1–9, 2008, [Online]. Available: papers3://publication/uuid/692988CE-7E10-498E-96EC-E7A0CE3620A3.

[19] B. Roy, "All about Feature Scaling," *towardsdatascience*, 2020. https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35 (accessed Nov. 27, 2021).