Jurnal Kecerdasan Buatan dan Teknologi Informasi
Vol. 4, No. 1, January 2025, hlm. 01-12
e-ISSN: 2964-2922, p-ISSN: 2963-6191
DOI : https://doi.org/10.69916/jkbti.v4i1.160

IMPLEMENTASI ALGORITMA *DECISION TREE* PADA PENILAIAN KINERJA KARYAWAN MENGGUNAKAN KERANGKA *OBJECTIVE AND KEY RESULT*

Andrian Falah Kalyana[*1], Dini Hamidin[2], Saepudin Nirwan[3]

[1,2,3]Program Diploma III Teknik Informatika, Sekolah Vokasi, Universitas Logistik dan Bisnis Internasional, Bandung, Indonesia
Email: [1]andrianedisi1@gmail.com , [2]dinihamidin@ulbi.ac.id , [3]saepudin@ulbi.ac.id

(Diterima : 19 Agustus 2024, Direvisi : 6 September 2024, Disetujui : 16 Desember 2024)

Abstrak

OKR (Objectives and Key Results) merupakan kerangka kerja yang digunakan untuk menetapkan dan memantau tujuan serta hasil kunci yang ingin dicapai oleh suatu organisasi. Penilaian dilakukan secara berkala untuk mengukur kemajuan dan efektivitas OKR dalam mencapai tujuan yang telah ditentukan. Dalam penelitian ini, data sintetis digunakan untuk memprediksi kinerja karyawan. Data tersebut dihasilkan menggunakan metode faker dari *library Python*, dengan mengacu pada data asli. *Machine learning*, sebagai bagian dari kecerdasan buatan, memungkinkan sistem untuk belajar secara otomatis dan meningkatkan performanya berdasarkan pengalaman tanpa perlu pemrograman eksplisit. Algoritma *Decision Tree* diterapkan dalam penelitian ini untuk melakukan klasifikasi dan regresi. Tujuan penelitian ini adalah memprediksi kinerja karyawan dengan label "cukup," "baik," "memuaskan," dan "sangat memuaskan." Hasil penelitian menunjukkan bahwa algoritma *Decision Tree* mampu membangun pohon keputusan yang efektif dalam memprediksi kategori kinerja karyawan dengan akurasi mencapai 95%. Selain itu, nilai *Macro Average* untuk *precision* dan *recall* masing-masing adalah 0,75, sedangkan F1-Score mencapai 0,77. Adapun nilai *Weighted Average* untuk *F1-Score* juga sebesar 0,77. Kesimpulannya, algoritma *Decision Tree* terbukti efektif dalam evaluasi kinerja karyawan.

Kata kunci: *objective and key result(okr), machine learning, Decision Tree*

IMPLEMENTATION OF THE DECISION TREE ALGORITHM IN EMPLOYEE PERFORMANCE ASSESSMENT USING THE OBJECTIVE AND KEY RESULT FRAMEWORK

Abstract

OKR (Objectives and Key Results) is a framework used to set and monitor goals and key outcomes desired by an organization. Assessments are conducted periodically to evaluate progress and measure the effectiveness of OKRs in achieving the established goals. In this study, synthetic data is used to predict employee performance. The data is generated using the faker method from the Python library, referencing original data as a basis. Machine learning, as part of artificial intelligence, allows systems to learn automatically and improve their capabilities based on experience without the need for explicit programming. The Decision Tree algorithm is applied in this study for classification and regression tasks. The study aims to predict employee performance with labels such as "adequate," "good," "satisfactory," and "very satisfactory." The results indicate that the Decision Tree algorithm can effectively build a Decision Tree for predicting employee performance categories with an accuracy of 95%. Additionally, the Macro Average for precision and recall are both 0.75, with an F1-Score of 0.77. The Weighted Average for the F1-Score is also 0.77. In conclusion, the Decision Tree algorithm proves to be effective in evaluating employee performance.

Keywords: *objective and key results (okr), machine learning, decision tree algorithm*

1. PENDAHULUAN

Memantau dan meningkatkan kinerja serta produktivitas organisasi semakin penting untuk mencapai tujuan strategis yang telah ditetapkan [1]. Salah satu metode yang efektif dalam mengelola dan mengevaluasi kinerja adalah dengan menggunakan *Objective dan Key Result*. OKR adalah kerangka kerja yang digunakan untuk mendefinisikan dan melacak tujuan *(objective)* serta hasil kunci *(key result)* yang diinginkan oleh organisasi [2]. Penilaian ini

dilakukan secara berkala untuk mengevaluasi kemajan dan mengukur seberapa efektif OKR dalam mencapai tujuan [3]. Penggunaan OKR dalam penilaian kinerja karyawan menawarkan beberapa keunggulan. Pertama, OKR membantu organisasi menetapkan tujuan yang jelas dan terukur, yang dapat menjadi panduan bagi setiap individu dalam mencapai target mereka. Kedua, OKR meningkatkan transparansi dan akuntabilitas, karena setiap anggota tim dapat melihat peran mereka dalam pencapaian tujuan organisasi. Ketiga, OKR memfasilitasi komunikasi yang lebih efektif di seluruh organisasi, memungkinkan penyesuaian strategi secara dinamis untuk merespons perubahan dalam lingkungan bisnis.

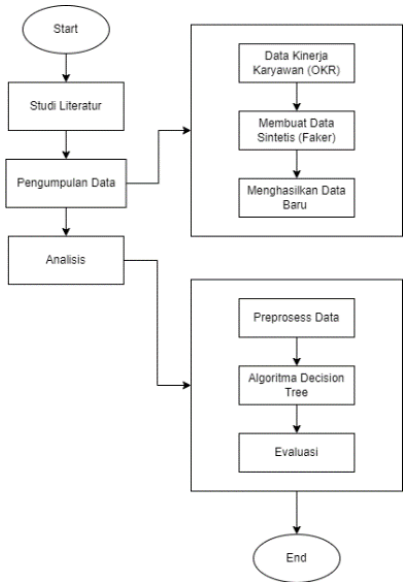
Dalam konteks penelitian ini, OKR digunakan untuk mengukur dan mengevaluasi kinerja karyawan disuatu perusahaan. Data kinerja yang dikumpulkan melalui OKR dapat memberikan wawasan penting mengenai pencapaian individu maupun tim dalam organisasi [4]. Namun, untuk mendapatkan pemahaman yang lebih mendalam dan akurat diperlukan analisis data lebih komprehensif. *Machine learning* adalah Aplikasi bagian dari kecerdasan buatan yang membuat sistem memiliki kemampuan belajar secara otomatis dan meningkatkan kemampuan berdasarkan pengalaman tanpa diprogram secara eksplisit [5]. Tujuan dari *Machine learning* yakni untuk menemukan dan mengaplikasikan pola-pola di dalam data [6]. *Machine learning* dapat digunakan untuk mengalisis data OKR, membuat prediksi dan rekomendasi yang dapat membantu pengambilan keputusan strategis. Pembersihan dan praproses data, eksplorasi data, pelatihan dan evaluasi model adalah beberapa langkah penting dalam proses ini [7]. Dalam pengolahan data OKR kinerja karyawan, metode *Decision Tree* digunakan sebagai salah satu teknik *Machine learning*. Metode *Decision Tree* ini merupakan teknik klasifikasi yang efektif dalam proses tersebut [8]. Dengan menggunakan teknik *Machine learning*, hasil analisis ini diharapkan dapat memberikan gambaran yang lebih jelas dan mendalam tentang faktor-faktor yang mempengaruhi kinerja karyawan. *Decision Tree* adalah metode dalam *Machine learning* yang digunakan untuk klasifikasi dan regresi. Metode ini memodelkan keputusan dan hasil strukkturnya menyerupai pohon. Simpul akar (*internal node*) mewakili fitur dalam dataset, simpul ranting (*branch node*) mewakili aturan keputusan, dan setiap simpul daun (*leaf node*) mewakili hasil atau output [9].

Penelitian ini diharapkan perusahaan dapat meningkatkan produktivitas dan efektivitas kerja, serta mencapai tujuan strategis mereka dengan lebih optimal dengan penggunaan *Machine learning* dan algoritma *Decision Tree* dalam analisis data kinerja kerja. Dengan demikian, teknologi *Machine learning* dapat diintegrasikan dengan kerangka kerja OKR untuk membuat sistem pengelolaan kinerja karyawan yang lebih efisien.

2. METODE PENELITIAN

Metodologi untuk pembuatan data sintetis menggunakan *faker*. *Faker* sangat berguna untuk kebutuhan pengujian, simulasi, dan pengembangan perangkat lunak, terutama ketika data asli tidak tersedia atau tidak digunakan karena alasan privasi atau keamanan .

Metodologi yang digunakan untuk menganalisis data adalah dengan penerapan *Machine learning* menggunakan algoritma *Decision Tree.* Algoritma *Decision Tree* ini bertujuan untuk memprediksi kinerja karyawan, dengan haril output yang dikategorikan sebagai cukup, baik, memuaskan, dan sangat memuaskan [10]. Metodologi yang digunakan untuk pengembangan website yaitu menggunakan metode waterfall dengan tahapan, requirements analiysis, design, implementasi dan, testing. Tahapan penelitian sebagai berikut.



Gambar 1. Tahapan Penelitian

2.1 Analisis

Data sintetis yang telah dibuat dianalisis menggunakan *Machine learning* dengan mengimplementasikan algoritma *Decision Tree* untuk memprediksi kinerja karyawan.

2.2 Preprocedding Data

Data yang telah diperoleh perlu diolah terlebih dahulu sebelum digunakan untuk prediksi. Pengolahan data ini bertujuan untuk mempersiapkan data agar dapat digunakan secara efektif dalam proses prediksi.

2.3 *Exploratory Data Analysis (EDA)*

Exploratory Data Analysis (EDA) adalah pendekatan untuk menganalisis data dengan cara visualisasi, statistik, dan teknik eksplorasi lainnya untuk memahami struktur, pola, dan hubungan dalam dataset sebelum menerapkan model analitik atau *Machine learning*.

2.4 *Feature Engineering*

Feature Engineering adalah proses dalam *Machine learning* dan data science yang melibatkan pembuatan, transformasi, dan pemilihan fitur (variabel) dari data mentah untuk meningkatkan kinerja model prediksi. Tujuannya adalah untuk membuat fitur yang lebih informatif, relevan, dan berguna bagi algoritma *Machine learning*.

2.5 Normalisasi Data

Proses skala ulang fitur dalam dataset sehingga semua fitur memiliki rentang yang serupa, biasanya dalam rentang 0 hingga 1 atau -1 hingga 1. Proses ini membantu algoritma *Machine learning* berkonvergensi lebih cepat dan meningkatkan akurasi model. Contoh seperti ditunjukkan pada rumus formula(1)

$$x' = \frac{x - x\_min}{x\_\max - x\_min} \quad (1)$$

Keterangan:
X = nilai asli dari fitur.
Xmin = nilai minimum dari fitur dalam dataset.
Xmax = nilai maksimum dari fitur dalam dataset.
X′ = nilai yang dinormalisasi.

2.6 *Encode Data*

Proses konversi data ke dalam format yang dapat dipahami oleh algoritma *Machine learning*, terutama ketika data tersebut tidak berbentuk numerik.

2.7 *Modeling*

Proses pembagian dataset menjadi set pelatihan *(training set)* dan set pengujian *(test set)* untuk evaluasi model. Pembagian ini membantu dalam melatih model pada data yang sudah dikenal dan menguji kinerjanya pada data yang belum pernah dilihat sebelumnya.

2.8 Implementasi Algoritma *Decision Tree*

Tahap di mana model *Decision Tree* dibangun dan digunakan untuk melakukan prediksi berdasarkan dataset yang tersedia. Langkah-langkah implementasi ini melibatkan beberapa fase, mulai dari preprocessing data hingga evaluasi model.

2.9 Rumus Gini Index

Rumus dari algoritma gini index bertujuan untuk mengukur kemurnian atau homogenitas daris sebuah node. Contoh seperti ditunjukkan pada rumus formula (2)

$$Gini(S) = 1 - \sum_{i=1}^{n} Pi2 \quad (2)$$

Keterangan:
S = set data atau node yang sedang dianalisis.
n = jumlah kelas yang ada dalam data.
Pi = proporsi dari elemen dalam kelas i di dalam set data S.

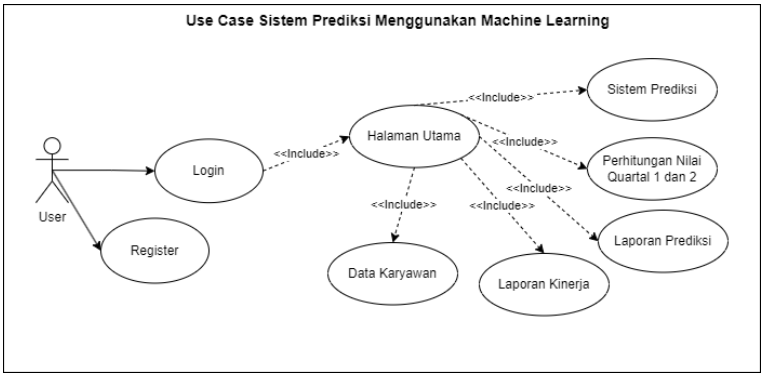
2.10 Evaluasi Model

Dengan memvisualisasikan hasil prediksi menggunakan pohon keputusan dan confusion matri, peneliti dapat memperoleh gambaran yang jelas tentang pola dan distribusi kelompok-kelompok tersebut. Selain itu

memvisualisasikan kinerja karyawan pada setiap predikat memungkinkan peneliti untuk mengetahui nilai rata-rata kinerja setiap karyawannya

2.11 *Use Case*

Use Case diagram menggambarkan fungsional yang di harapkan dari sebuah sistem. Tujuan dari *Use Case* adalah untuk menunjukkan interaksi antara aktor dengan sistem. Aktor adalah suatu entitas manusia yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.



Gambar 1. *Use Case* Sistem Prediksi Penilaian Kinerja Karyawan

3. HASIL DAN PEMBAHASAN

Pada bagian ini dijelaskan terkait dengan pengolahan data yang di lakukan pada data kinerja karyawan menggunakan *Machine learning*.

a. *Imporiting Library*

Pada tahap ini mengimport semua *library* yang dibutuhkan untuk memprediksi penilaian kinerja karayawan.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import tensorflow as tf
import os
from random import randrange
import scipy as sp
from scipy import stats
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score, StratifiedKFold
from sklearn.tree import DecisionTreeClassifier, plot_tree, export_graphviz
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, classification_report
from sklearn import tree
import graphviz
import pickle
import warnings
```

Gambar 3. *Importing Library*

b. Membaca Data

Data yang dipakai data sintetis kinerja karyawan dengan format .CSV

```
# Membaca data pada excel
data = pd.read_csv('data_sintetis2.csv')
data
```

Gambar 4. Membaca Data

c. *Data Processing*

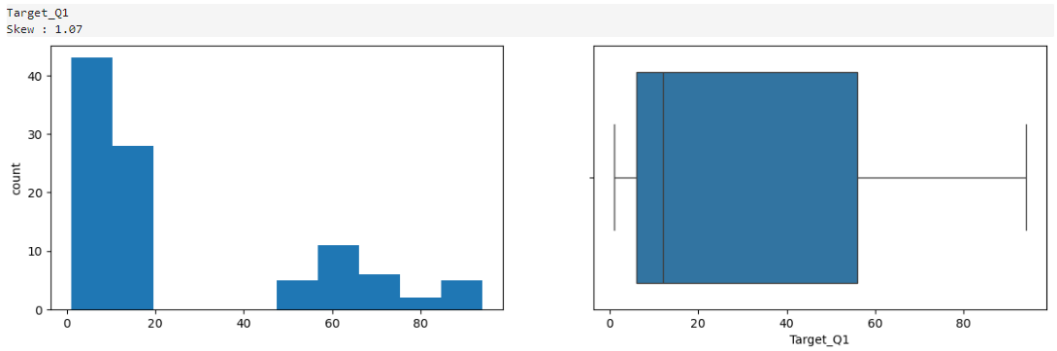
Data Processing bertujuan untuk memastikan bahwa data yang digunakan dalam model adalah akurat, konsisten dan bebas dari kesalahan. Proses ini mencakup berbagai langkah untuk mengatasi maslah umum yang ditumukan dalam data mentah.

```
data = data.drop_duplicates()
data
```

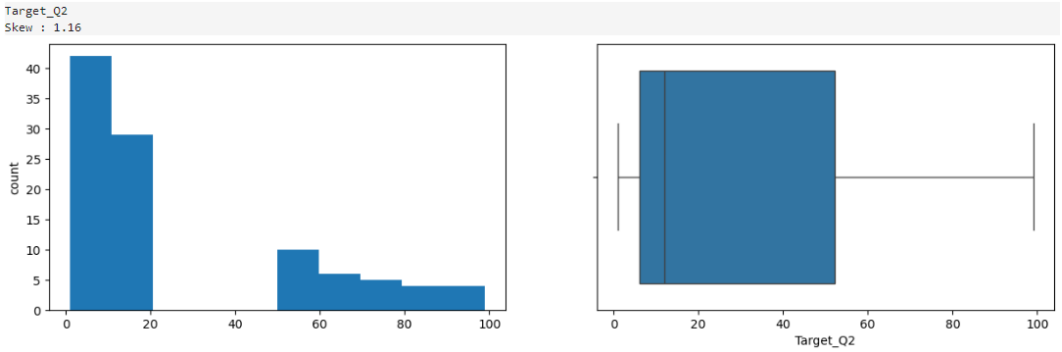
Gambar 5. *Data Processing*

d. *Exploratory Data Analysis (EDA)*

Exploratory Data Analysis (EDA) adalah pendekatan untuk menganalisis data dengan cara visualisasi, statistik, dan teknik eksplorasi lainnya untuk memahami struktur, pola, dan hubungan dalam dataset sebelum menerapkan model analitik atau *Machine learning*. Distribusi target q1 memiliki nilai 1.07, dan target q2 memiliki nilai 1.16, memiliki skewness posistif yang signifikan, yang mengidentifikasikan bahwa data cenderung berkumpul di nilai-nilai yang lebih rendah dengan beberapa outlier tinggi.

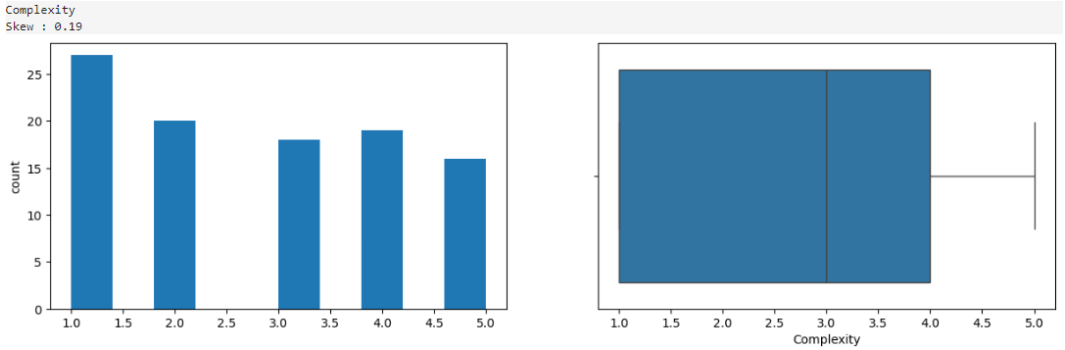


Gambar 6. Diagram Target Q1

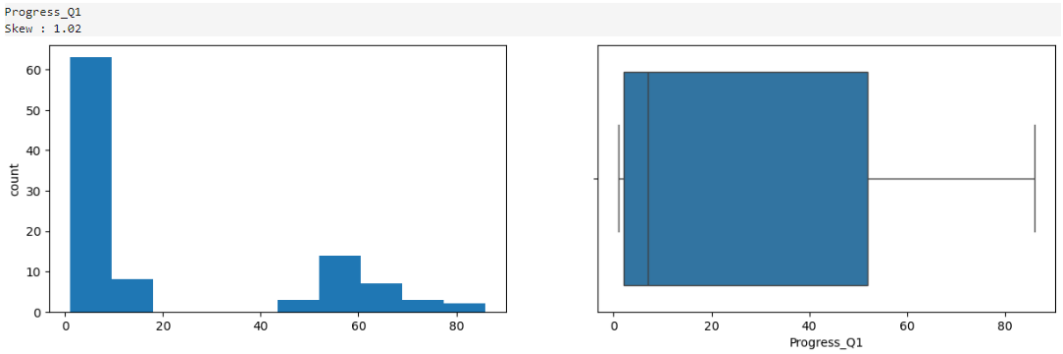


Gambar 7. Diagram Target Q2

Complexcity memiliki nilai 0.19, memiliki distribusi yang hampir simetris data yang relatif merata dan target q2 memiliki nilai 1.16, yang mengidentifikasikan bahwa data cenderung berkumpul di nilai-nilai yang lebih rendah dengan beberapa outlier tinggi.

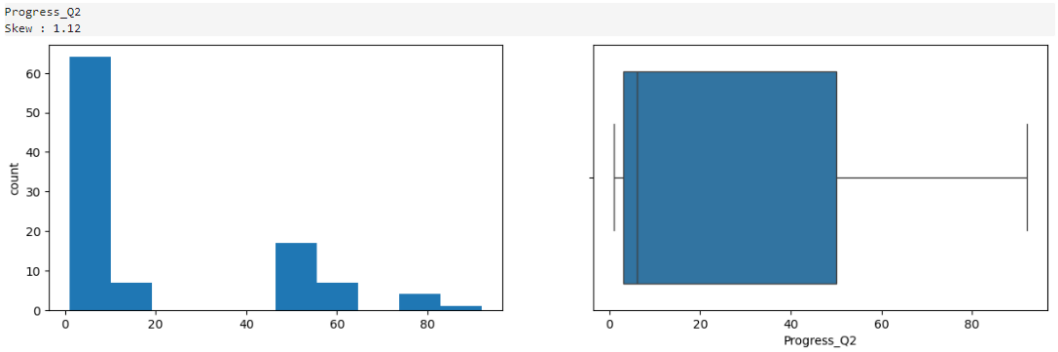


Gambar 8. Diagram *Complexcity*

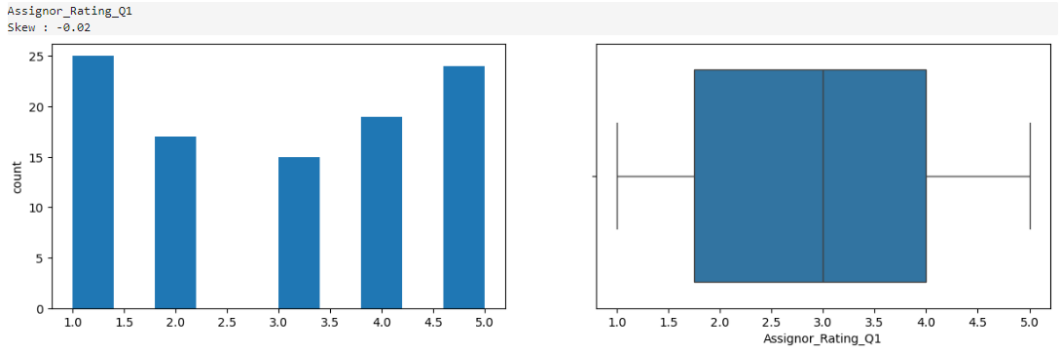


Gambar 9. Diagram Progress Q1

Progress Q2 memiliki nilai 1.12, yang mengidentifikasikan bahwa data cenderung berkumpul di nilai-nilai yang lebih rendah dengan beberapa outlier tinggi. Assignor rating q1 memiliki nilai -0.02, data cenderung pada nilai yang lebih tinggi dengan beberapa outlier rendah.

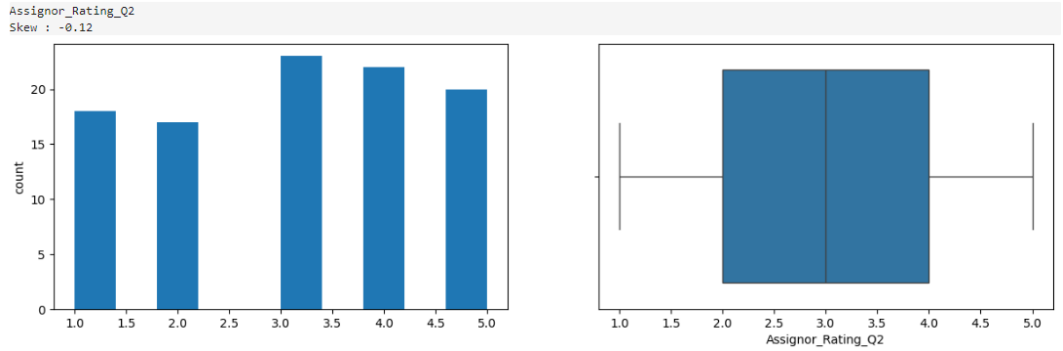


Gambar 10. Diagram Progress Q2

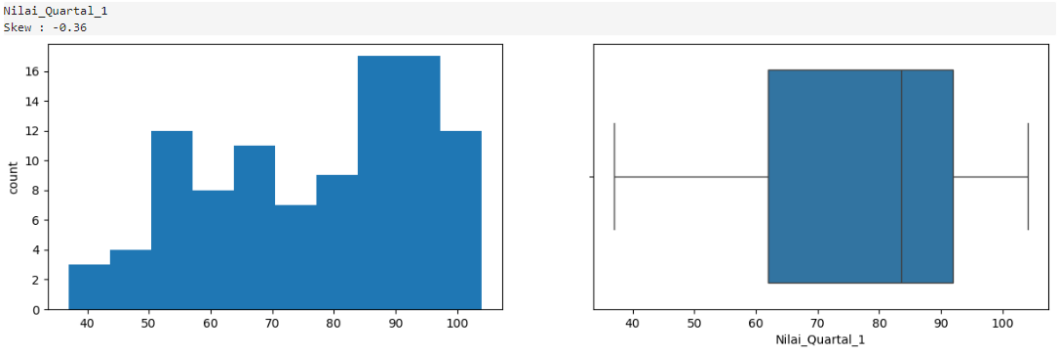


Gambar 11. Diagram *Assignor* Rating Q1

Assignor rating q1 memiliki nilai -0.12 dan nilai quartal 1 memiliki nilai -0.36, data cenderung pada nilai yang lebih tinggi dengan beberapa outlier rendah.

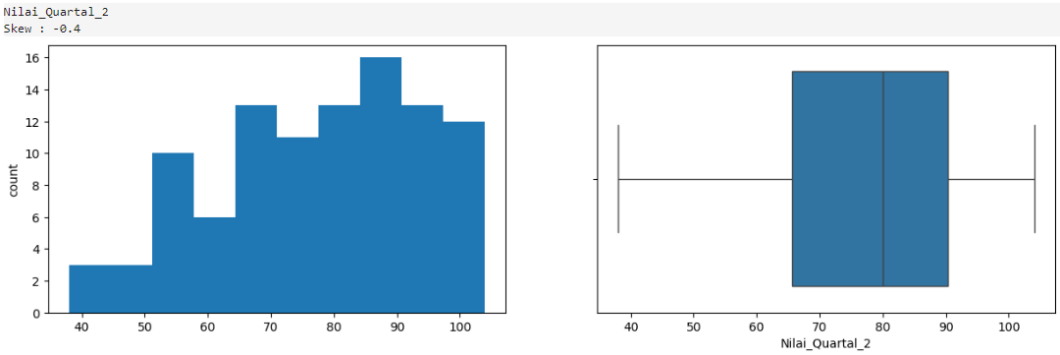


Gambar 12. Diagram *Assignor* Rating Q2

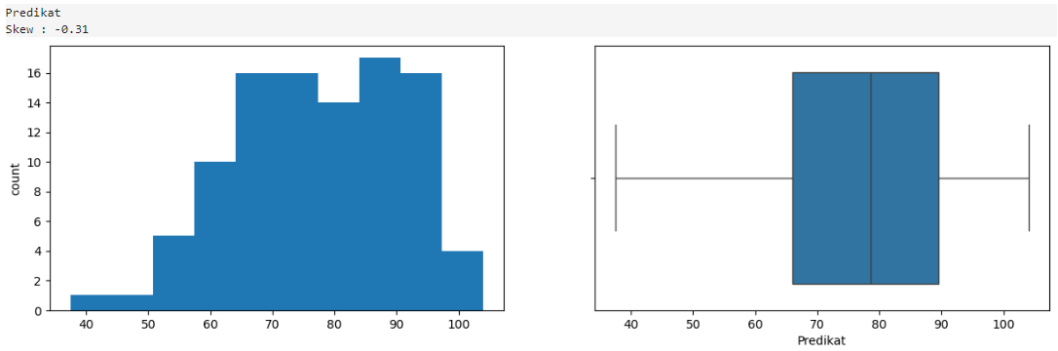


Gambar 13. Diagram Nilai Quartal 1

Nilai quartal 2 memiliki nilai -0.4 dan predikat -0.31, data cenderung pada nilai yang lebih tinggi dengan beberapa outlier rendah.



Gambar 14. Diagram Nilai Quartal 2



Gambar 15. Diagram Predikat

e. *Feature Engineering*

Proses dalam *Machine learning* dan data science yang melibatkan pembuatan, transformasi dan pemilihan fitur (variable) dari data mentah guna meningkatkan kinerja model prefiktif.

```
X = data.drop(['Kategori_Predikat', 'Predikat', 'NIP', 'Nama', 'Jabatan', 'Unit', 'Tanggal'], axis=1)
y = data['Kategori_Predikat']
```

```
X = data[['Objective_1', 'Key_Result_1', 'Objective_2', 'Key_Result_2', 'Target_Q1', 'Target_Q2', 'Compl
X
```

Gambar 16. *Feature Engineering*

f. Normaslisasi Data

Gambar 17 adalah proses normalisasi data. Proses mengubah data ke dalam skala tertentu biasanya antara 0 dan 1, tanpa mengubah distribusi atau hubungan antara nilai-nilai dalam dataset.

```
from sklearn.preprocessing import StandardScaler

def normalize_data(data, columns_to_normalize):
    scaler = StandardScaler()
    data[columns_to_normalize] = scaler.fit_transform(data[columns_to_normalize])
    return data

# Kolom-kolom numerik yang akan dinormalisasi
columns_numeric = ['Target_Q1', 'Target_Q2', 'Complexity', 'Progress_Q1', 'Progress_Q2', 'Assignor_Rating_Q1',

# Memanggil fungsi normalize_data hanya untuk kolom numerik
data_normalized = normalize_data(data, columns_numeric)

# Menampilkan DataFrame yang sudah dinormalisasi
print("Mean values after normalization:")
print(data_normalized[columns_numeric].mean())

print("\nStandard deviation values after normalization:")
print(data_normalized[columns_numeric].std())
```

Gambar 17. Data Yang Telah Dinormalisasikan

g. *Data Encoded*

Gambar dibawah ini adalah proses data *encoded*. Proses konversi data ke dalam format yang dapat dipahami oleh algoritma *Machine learning*, terutama ketika data tersebut tidak berbentuk numerik.

```
# Misalkan X adalah DataFrame awal
# ... (kode untuk mendefinisikan DataFrame X)

# Copy DataFrame agar tidak merubah DataFrame asli
data_encoded = X.copy()

# Inisialisasi LabelEncoder
label_encoder = LabelEncoder()

# Lakukan label encoding pada kolom
data_encoded['Objective_1'] = label_encoder.fit_transform(X['Objective_1'])
data_encoded['Key_Result_1'] = label_encoder.fit_transform(X['Key_Result_1'])

# Lakukan label encoding pada kolom
data_encoded['Objective_2'] = label_encoder.fit_transform(X['Objective_2'])
data_encoded['Key_Result_2'] = label_encoder.fit_transform(X['Key_Result_2'])

# Tampilkan DataFrame hasil
print(data_encoded)
```

Gambar 18. *Data Encoded*

h. *Modelling*

Gambar dibawah ini adalah proses modeling. Proses merancang dan melatih model matematis untuk memprediksi atau mengklasifikasikan data berdasarkan pola yang ditemukan dalam dataset.

```
# Pilih fitur dan target
features = ['Objective_1', 'Key_Result_1', 'Key_Result_2', 'Objective_2', 'Target_Q1', 'Target_Q2',
X_features = data_encoded[features]
y = data_encoded['Kategori_Predikat']

# Split data menjadi train dan test set
X_train, X_test, y_train, y_test = train_test_split(X_features, y, test_size=0.2, random_state=0)

# Tampilkan ukuran dari train dan test set
print("Train set size:", X_train.shape[0])
print("Test set size:", X_test.shape[0])
```

Gambar 19. *Modelling*

i. Model Training Algoritma *Decision Tree*

Gambar dibawah ini hasil dari traning model menggunakan algoritma *Decision Tree*, yang menghasilkan pohon keputusan, pohon keputusan ini mencari hasil akhir. Berikut adalah codenya.

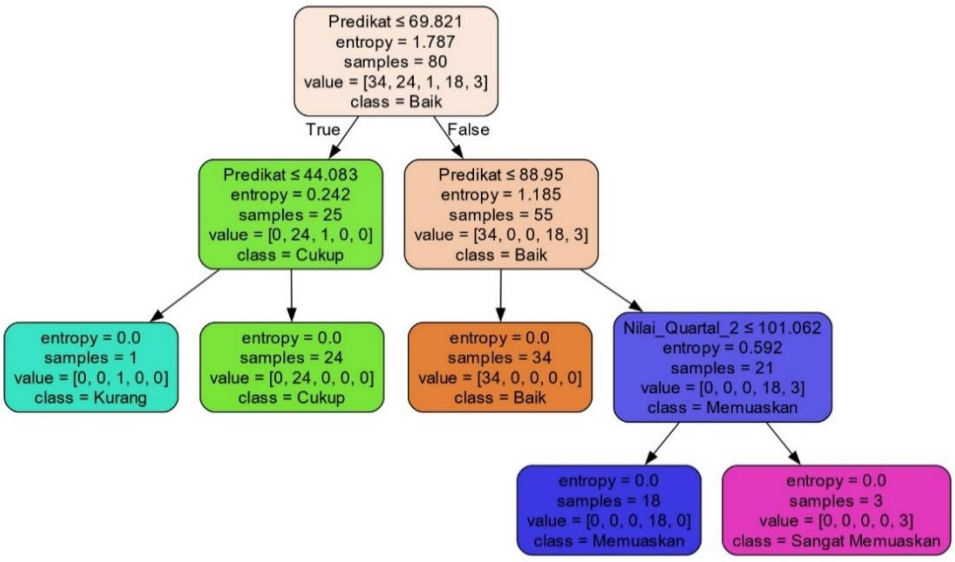
```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier(criterion="entropy", max_depth=3)
# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)
#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

```
# Verifikasi path Graphviz
os.environ["PATH"] += os.pathsep + 'C:/Program Files/Graphviz/bin'

# Visualisasikan pohon keputusan
dot_data = export_graphviz(clf, out_file=None,
                           feature_names=X_test.columns,
                           class_names=clf.classes_,
                           filled=True, rounded=True,
                           special_characters=True)

graph = graphviz.Source(dot_data)
graph.render("pohonkeputusan")  # Menyimpan hasil ke file
graph
```

Gambar 20. Kode Pembuatan Pohon Keputusan

Berikut keterangan untuk gambar dibawah ini

1. Predikat ≤ 69.821: Sebagian besar data diklasifikasikan sebagai 'Baik' jika nilai predikat di bawah 69.821.
2. Predikat ≤ 44.083: Dalam subset dengan Predikat ≤ 69.821, jika predikat juga kurang dari atau sama dengan 44.083, mayoritas sampel akan diklasifikasikan sebagai 'Cukup'.
3. Predikat > 88.95: Jika nilai predikat lebih besar dari 88.95, sebagian besar data diklasifikasikan sebagai 'Baik', tetapi klasifikasi lebih lanjut bergantung pada nilai Nilai_Quartal_2.



Gambar 21. Pohon Keputusan

j. Evaluasi Model

Precision, Recall, dan F1 Score

1. Precision: Mengukur berapa banyak dari prediksi yang benar-benar relevan. Rumusnya adalah Precision = True Positives / (True Positives + False Positives).
2. Recall: Mengukur berapa banyak dari kasus yang relevan yang berhasil ditemukan oleh model. Rumusnya adalah Recall = True Positives / (True Positives + False Negatives).
3. F1 Score: Merupakan harmonic mean dari precision dan recall, memberikan gambaran keseimbangan antara keduanya. Rumusnya adalah F1 = 2 * (Precision * Recall) / (Precision + Recall).
4. Precision: 1.0 (Mencapai 100% untuk presisi, artinya semua prediksi yang dibuat model benar-benar relevan).
5. Recall: 0.95 (Artinya, model berhasil menemukan 95% dari semua kasus yang relevan).
6. F1 Score: 0.97 (Gabungan dari precision dan recall yang sangat tinggi).

Accuracy

Accuracy on test set: 0.95 (95% dari prediksi model pada data uji adalah benar). Ini dihitung dengan accuracy = *(True Positives + True Negatives) / Total Samples*.

Classification Report

Classification report memberikan rincian performa model pada masing-masing kelas yang diprediksi. Kolom yang dijelaskan dalam *classification report* adalah sebagai berikut:

1. *Precision*: Tingkat ketepatan model dalam mengklasifikasikan setiap kelas.
2. *Recall*: Kemampuan model dalam mendeteksi setiap kelas.
3. *F1-Score*: Keseimbangan antara precision dan recall untuk setiap kelas.
4. *Support*: Jumlah contoh sebenarnya untuk setiap kelas dalam dataset uji.

Pada laporan ini, kita melihat performa pada empat kelas: 'Baik', 'Cukup', 'Memuaskan', dan 'Sangat Memuaskan'.

Overall Metrics:

1. *Accuracy*: 95%
2. *Macro Average*:
3. *Average dari precision, recall,* dan *F1-Score* dihitung untuk setiap kelas dan diambil rata-rata (0.75 untuk *precision* dan *recall*, 0.77 untuk *F1-Score*).
4. *Weighted Average:*
5. Rata-rata tertimbang dari metrik di atas berdasarkan *support* dari setiap kelas (0.77 untuk *F1-Score*).

k. *Confussion Matrix*

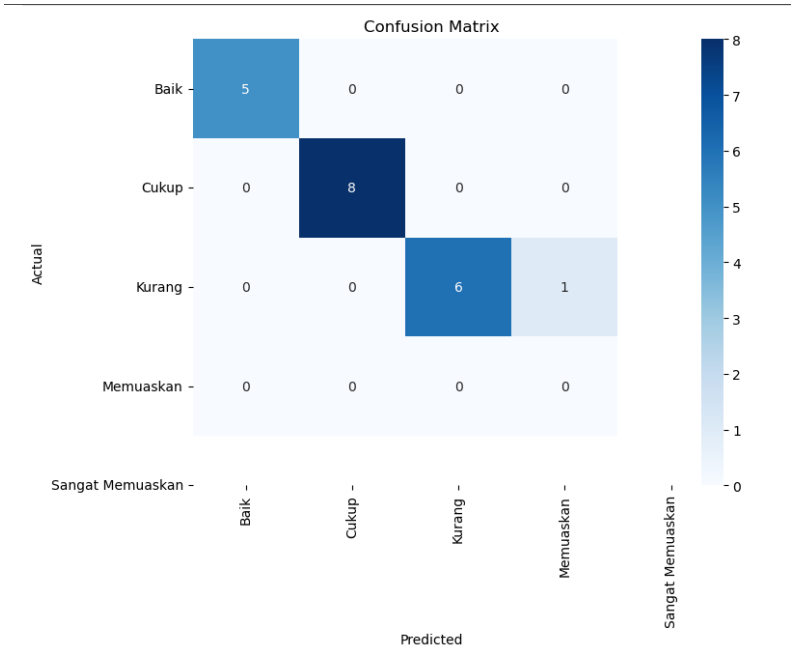
Karena data yang digunakan sedikit berikut hasil analisis menggunakan confusion mastrix, Akurasi Tinggi pada 'Baik', 'Cukup', dan ‘Kurang’: Model ini memiliki performa yang sangat baik dalam mengklasifikasikan kelas 'Baik', 'Cukup', dan ‘Kurang’, yang ditunjukkan oleh nilai diagonal yang tinggi (5, 8, dan 6) tanpa ada kesalahan (nilai di luar diagonal utama).

l. Struktur *Confussion Matrix*:

1. Baris *(Actual)*: Label sebenarnya dari data uji.
2. Kolom *(Predicted)*: Prediksi yang dibuat oleh model.
3. Diagonal utama: Prediksi yang benar untuk setiap kelas.
4. Di luar diagonal: Kesalahan prediksi.

Interpretasi:

1. Baik: 5 prediksi benar, 0 salah.
2. Cukup: 8 prediksi benar, 0 salah.
3. Kurang: 6 prediksi benar, 1 salah.
4. Memuaskan: Tidak ada prediksi benar.
5. Sangat Memuaskan: Tidak ada prediksi.



Gambar 22. *Confussion Matriks* Penilaian Kinerja Karyawan

4. KESIMPULAN

Penelitian ini mengimplementasikan metode pembuatan data sintetis dan algoritma pohon keputusan (*Decision Tree*) untuk memprediksi kinerja karyawan. Hasil evaluasi model menunjukkan akurasi sebesar 95%, yang berarti 95% dari prediksi model pada data uji sesuai dengan hasil yang benar. Selain itu, evaluasi juga mencakup penghitungan *macro average* dan *weighted average*, dengan nilai *precision* dan *recall* masing-masing mencapai 0.75, serta *F1-Score* sebesar 0.77 pada kedua kategori tersebut. Analisis confusion matrix mengindikasikan bahwa model ini unggul dalam mengklasifikasikan kelas 'Baik', 'Cukup', dan 'Kurang', dengan prediksi akurat sebanyak 5, 8, dan 6 kali. Namun, terdapat tantangan dalam mengklasifikasikan kelas 'Memuaskan' dan 'Sangat Memuaskan'. Penggunaan *library Python Faker* berhasil menghasilkan data sintetis yang sangat menyerupai data asli tanpa mengungkapkan informasi sensitif. Algoritma *Decision Tree* juga mampu membentuk model yang efektif dalam menggambarkan proses pengambilan keputusan berbasis data pelatihan, menghasilkan prediksi yang akurat, dan relevan untuk penilaian kinerja karyawan.

DAFTAR PUSTAKA

[1] P. Ogheneogaga Irikefe, “Effect of Objectives and Key Results (OKR) on Organisational Performance in the Hospitality Industry,” *IJRP*, vol. 91, no. 1, Dec. 2021, doi: 10.47119/IJRP1009111220212596.

[2] V. Stray, J. H. Gundelsby, R. Ulfsnes, and N. Brede Moe, “How agile teams make Objectives and Key Results (OKRs) work,” in *Proceedings of the International Conference on Software and System Processes and International Conference on Global Software Engineering*, Pittsburgh PA USA: ACM, May 2022, pp. 104–109. doi: 10.1145/3529320.3529332.

[3] J. L. Butler, T. Zimmermann, and C. Bird, “Objectives and Key Results in Software Teams: Challenges, Opportunities and Impact on Development,” in *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice*, Lisbon Portugal: ACM, Apr. 2024, pp. 358–368. doi: 10.1145/3639477.3639747.

[4] V. Stray, N. B. Moe, H. Vedal, and M. Berntzen, “Using Objectives and Key Results (OKRs) and Slack: A Case Study of Coordination in Large-Scale Distributed Agile”.

[5] A. S. Diantika and Y. Firmanto, “IMPLEMENTASI *MACHINE LEARNING* PADA APLIKASI PENJUALAN PRODUK DIGITAL (STUDI PADA GRABKIOS)”.

[6] D. K. M.Sc, *Pengenalan Machine learning dengan Python*. Elex Media Komputindo, 2022.

[7] A. Rizal, “Tahapan Desain dan Implementasi Model *Machine learning* untuk Sistem Tertanam,” *Ultima Computing : Jurnal Sistem Komputer*, vol. 12, no. 2, pp. 79–85, Nov. 2020, doi: 10.31937/sk.v12i2.1782.

[8] I. Sutoyo, “IMPLEMENTASI ALGORITMA *DECISION TREE* UNTUK KLASIFIKASI DATA PESERTA DIDIK,” *Jurnal Pilar Nusa Mandiri*, vol. 14, no. 2, Art. no. 2, 2018, doi: 10.33480/pilar.v14i2.70.

[9] A. H. Nasrullah, "IMPLEMENTASI ALGORITMA *DECISION TREE* UNTUK KLASIFIKASI PRODUK LARIS :," *Jurnal Ilmiah Ilmu Komputer Fakultas Ilmu Komputer Universitas Al Asyariah Mandar*, vol. 7, no. 2, Art. no. 2, Sep. 2021, doi: 10.35329/jiik.v7i2.203.

[10] B. Imran, Zaeniah, Sriasih, S. Erniwati, and Salman, "Data Mining Using a Support Vector Machine , Decision Tree , Logistic Regression and Random Forest for," *J. Infokum*, vol. 10, no. 2, pp. 792–802, 2022.