Perbandingan Algoritma *Decision Tree* Dan *Support Vector Machine* Dalam Prediksi Kualitas Udara

Muhammad Fahmi[a1], IKG Suhartana[a2]

[a]Informatika, Universitas Udayana
Jimbaran, Bali, Indonesia
[1]1808561084@student.unud.ac.id
[2]ikg.suhartana@unud.ac.id

Abstrak

Indonesia memiliki banyak permasalahan yang sangat penting untuk diatasi salah satunya pencemaran udara. Pencemaran udara tergolong permasalahan yang sedang dihadapi seluruh dunia salah satunya Indoneisa. Indonesia termasuk dalam urutan 20 besar dunia dalam pencemaran udara yang diukur berdasarkan *AQI (Air Quality Index)* dunia. Hal tersebut dikarenakan di Indonesia masih banyak aktivitas masyarakat yang menghasilkan polutan dan gas berbahaya seperti SO2, NO2, CO, PM10, PM2.5, dan O3. Polutan dan gas berbahaya sangat berbahaya bagi kesehatan karena dapat menimbulkan gangguan kesehatan seperti sesak nafas, iritasi kulit, dan penyakit lainnya. Oleh karena itu, dibutuhkan metode yang dapat mengklasifikasikan kualitas udara menjadi kelas tertentu sehingga masyarakat maupun pemerintah dapat mengetahui seberapa baik dan tercemarnya udara di sekitar mereka. Dalam penelitian ini dilakukan perbandingan metode klasifikasi antara SVM (*support vector machine)* dengan metode *Decision Tree*. Hasil dari penelitian ini metode *SVM* mendapatkan hasil yaitu akurasi sebesar 91%, *precision* sebesar *87%, recall* sebesar *61%,* dan *f1-score* sebesar *72%, sedangkan* metode *Decision Tree* menghasilkan tingkat akurasi sebesar 97%, dengan presisi sebesar 90%, recall sebesar 85%, dan f1-score sebesar 88%.

Keywords: ISPU*, Decision Tree, Air Quality, Support Vector Machine,* Data Mining

1. Introduction

Udara merupakan komponen utama di bumi yang dibutuhkan oleh makhluk hidup untuk bernapas. Udara yang dapat digunakan untuk bernapas yaitu udara yang bersih. Udara yang bersih kini semakin sulit untuk dicari karena kualitas udara yang semakin menurun. Penurunan kualitas udara ini disebabkan karena meningkatnya penggunaan transportasi serta banyaknya industri yang menghasilkan polutan seperti PM10 dan PM2.5 serta banyak nya penggunaan gas buang seperti karbon dioksida (CO2), karbon monoksida (CO), nitrogen monoksida (NO), dan ozon (O3). Pada bulan Juni kemarin, menurut databoks katadata, negara Indonesia terutama ibukota Jakarta menduduki peringkat kedua (2) kualitas udara terburuk dunia setelah Afrika Selatan dengan *Air Quality Index* (AQI) sebesar 166[1].

Menurut Peraturan Menteri Lingkungan Hidup Dan Kehutanan Republik Indonesia Nomor P.14/Menlhk/Setjen/Kum.1/7/2020 Tentang Indeks Standar Pencemar Udara, kualitas udara dapat dihitung dan menghasilkan tabel nilai berupa Indeks Standar Pencemaran Udara atau ISPU. Nilai ISPU tersebut dapat digunakan sebagai dasar untuk menentukan kualitas udara pada suatu daerah. Penentuan nilai kualitas udara dapat dikelompokkan menjadi beberapa kategori yaitu BAIK, SEDANG, TIDAK SEHAT, SANGAT TIDAK SEHAT, dan BERBAHAYA[2].

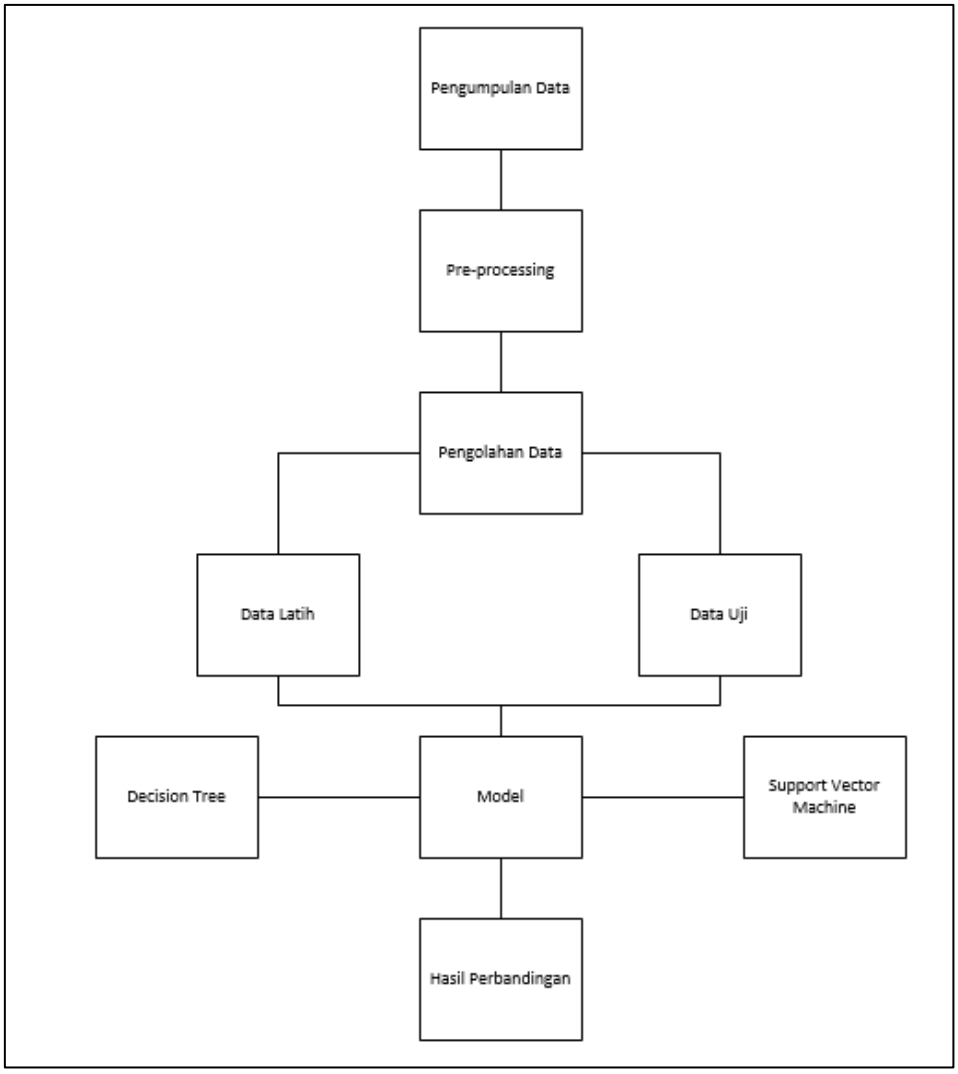
Pada penelitian sebelumnya yang dilakukan oleh [3] yang meneliti tentang penerapan algoritma *support vector machine* dalam menentukan kualitas udara. Hasil yang didapatkan yaitu *recall* sebesar 84.13%, *precision* sebesar 86.74%, dan *f1-score* sebesar 85.30%. Penelitian lainnya yang dilakukan oleh [4] yang meneliti tentang pebandingan performa antara algoritma *decision tree* dan algoritma *support vector machine.* Hasil yang diperoleh yaitu algoritma *decision tree* menghasilkan nilai *recall* sebesar 99,73%*, precision* sebesar 99,02%*,* dan *accuracy* sebesar

99,40%. Pada algoritma *support vector machine* dihasilkan *recall* sebesar 88,89%, *precision* sebesar 95,82%, dan *accuracy* sebesar 94,93%.

Dari penelitian – penelitian tersebut, penulis menggunakan algoritma *decision tree* dan algoritma *support vector machine* untuk menguji serta membandingkan performa dari kedua algoritma tersebut dalam mengklasifikasikan kualitas udara.

2. Reseach Methods

Tahapan-tahapan dalam penelitian ini dapat dilihat pada gambar berikut.



Gambar 1. Tahapan Penelitian

Berdasarkan gambar 1, tahap - tahap dalam penelitian ini terdiri atas pengumpulan data, pre-processing, pengolahan data, pembagian data menjadi data latih dan uji, memasukkan data ke model, dilanjutkan dengan melihat hasil perbandingan algoritma.

A. Pengumpulan Data

Data diambil dari open dataset yang terdapat di open data Jakarta yang dapat diakses melalui link berikut https://data.jakarta.go.id/. Dataset ini merupakan dataset kualitas udara DKI Jakarta pada bulan Januari sampai Desember 2021. Dataset ini terdiri atas 1825 baris dengan 10 atribut dan 1 kelas. Dataset yang diperoleh memiliki ekstensi .csv

B. Pre-processing Data

Pada proses ini data diseleksi untuk memilah atribut data yang tidak diperlukan, memilah baris atau kolom yang masih kosong hingga mengisi *record* data yang masih kosong dengan nilai yang baru. Kemudian data yang diolah dapat digunakan untuk melatih dan menguji model klasifikasi.

C. Pengolahan Data

Pengolahan data dilakukan dengan membagi dataset yang telah melewati tahap sebelumnya menjadi 2 (dua) yakni data uji serta data latih lalu diterapkan ke dalam algoritma *decision tree* dan *support vector machine*. Berikut cara kerja masing – masing algoritma.

1. *Decision Tree*
   Algoritma *Decision Tree* merupakan algoritma yang digunakan untuk mengklasifikasi dan memprediksi dari suatu kumpulan data atau dataset dengan membentuk sebuah pohon. Model *decision tree* menggunakan perhitungan *entropy* untuk menentukan atribut yang akan diletakkan pada level 0 (root) maupun pada level – level berikutnya. Kelebihan Decision Tree yaitu Daerah pengambilan keputusan yan sebelumnya kompleks dan sangat global, dapat diubah menjadi lebih simpel dan spesifik, Eliminasi perhitungan-perhitungan yang tidak diperlukan, karena Ketika menggunakan metode pohon keputusan maka sampel diuji hanya berdasarkan criteria atau kelas tertentu, Fleksibel untuk memilih fitur dari node internal yang berbeda, fitur yang terpilih akan membedakan suatu kriteriadibandingkan kriteria yang lain dalam node yang sama.

2. *Support Vector Machine*
   Support Vector Machine (SVM) yaitu sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi – fungsi linier dalam sebuah fitur yang berdimensi tinggi dan dilatih dengan menggunakan algoritma pembelajaran yang didasarkan pada teori optimasi[4]. Akurasi yang dihasilkan oleh model sangat bergantung dengan penentuan *hyperparameter* dan fungsi *kernel* yang digunakan. SVM dapat dibedakan menjadi 2 yaitu SVM linear dan SVM non-linear. SVM linear digunakan untuk mengolah data yang dapat dipisahkan secara linear sedangkan SVM non-linear digunakan untuk data yang tidak bisa dipisahkan secara linear dan biasanya menggunakan kernel untuk memisahkannya.

3. Result and Discussion

3.1. Persiapan Data

Dataset pada penelitian ini diperoleh dari *website Jakarta Open Data* yang disusun oleh Dinas Lingkungan Hidup Provinsi DKI Jakarta dalam bentuk *comma separated value* (CSV). Pada dataset terdapat 1825 baris dengan atribut tanggal, wilayah, stasiun, PM10, CO2, CO, O3, NO2, Maximum, Critical. Berikut tabel contoh dataset ISPU.

Tabel 1. Tabel Dataset ISPU

tanggal	pm10	pm25	so2	co	o3	no2	max	critical	categori	location
1/1/2021	43	---	58	29	35	65	65	O3	SEDANG	DKI2

1/2/2021	58	---	86	38	64	80	86	PM25	SEDANG	DKI3
1/3/2021	64	---	93	25	62	86	93	PM25	SEDANG	DKI3
1/4/2021	50	---	67	24	31	77	77	O3	SEDANG	DKI2
1/5/2021	59	---	89	24	35	77	89	PM25	SEDANG	DKI3
1/6/2021	73	---	81	29	66	85	85	O3	SEDANG	DKI2
1/7/2021	36	---	52	22	55	72	72	O3	SEDANG	DKI2
1/8/2021	38	---	68	26	51	71	71	O3	SEDANG	DKI2
1/9/2021	60	---	77	34	42	80	80	O3	SEDANG	DKI2
1/10/2021	24	---	39	16	38	59	59	O3	SEDANG	DKI2

3.2. Pre-Processing

Dataset yang digunakan dalam penelitian ini masih memiliki beberapa nilai yang hilang atau *missing value* serta baris data yang tidak memiliki nilai atau *null*. Adapun beberapa atribut yang tidak digunakan karena dianggap tidak memiliki pengaruh terhadap pengolahan data selanjutnya. Dengan kata lain, penentuan kualitas udara ditentukan berdasarkan nilai ukuran dari parameter-parameter seperti PM10, PM25, SO2, CO, O3, NO2. Berikut ini contoh data yang perlu melewati tahap *pre-processing*.

Tabel 2. Dataset dengan *missing value* dan *null*

	tanggal	stasiun	pm10	pm25	so2	co	o3	no2	max	critical	categori
124	2021-01-01	DKI5 (Kebon Jeruk) Jakarta Barat	37	NaN	20	12	25	4	37	PM10	BAIK
125	2021-01-02	DKI5 (Kebon Jeruk) Jakarta Barat	33	NaN	18	20	64	6	64	CO	SEDANG
126	2021-01-03	DKI5 (Kebon Jeruk) Jakarta Barat	42	NaN	15	10	62	3	62	CO	SEDANG
127	2021-01-04	DKI5 (Kebon Jeruk) Jakarta Barat	27	NaN	14	7	31	4	31	CO	BAIK
128	2021-01-05	DKI5 (Kebon Jeruk) Jakarta Barat	41	NaN	15	9	28	---	41	PM10	BAIK
129	2021-01-06	DKI5 (Kebon Jeruk) Jakarta Barat	33	NaN	12	19	46	---	46	CO	BAIK
130	2021-01-07	DKI5 (Kebon Jeruk) Jakarta Barat	27	NaN	13	11	36	---	36	CO	BAIK
131	2021-01-08	DKI5 (Kebon Jeruk) Jakarta Barat	36	NaN	18	19	28	---	36	PM10	BAIK
132	2021-01-09	DKI5 (Kebon Jeruk) Jakarta Barat	46	NaN	17	26	27	12	46	PM10	BAIK
133	2021-01-10	DKI5 (Kebon Jeruk) Jakarta Barat	19	NaN	20	8	34	5	34	CO	BAIK
134	2021-01-11	DKI5 (Kebon Jeruk) Jakarta Barat	33	NaN	21	10	34	9	34	CO	BAIK
135	2021-01-12	DKI5 (Kebon Jeruk) Jakarta Barat	25	NaN	22	6	38	6	38	CO	BAIK
136	2021-01-13	DKI5 (Kebon Jeruk) Jakarta Barat	25	NaN	20	5	27	9	27	CO	BAIK
137	2021-01-14	DKI5 (Kebon Jeruk) Jakarta Barat	32	NaN	17	8	36	7	36	CO	BAIK

Sebelum memasuki proses seleksi atribut pada dataset terlebih dahulu dipilih. Atribut yang dipilih yaitu PM10, PM25, SO2, CO, O3, NO2, dan categori. Berikut potongan kode dalam bahasa python untuk menyeleksi atribut yang tidak diperlukan pada dataset serta hasil *output*-nya.

```
[ ]  1 df_cleaned = df_cleaned.drop(columns=['tanggal', 'stasiun', 'max', 'critical'])
     2 df_cleaned
```

Gambar 2. Kode pemrograman untuk menyeleksi atribut yang tidak diperlukan

	pm10	pm25	so2	co	o3	no2	categori
0	38	53	29	6	31	13	SEDANG
1	27	46	27	7	47	7	BAIK
2	44	58	25	7	40	13	SEDANG
3	30	48	24	4	32	7	BAIK
4	38	53	24	6	31	9	SEDANG
...	...	...	...	...	...	...	...
1820	54	76	36	14	21	47	SEDANG
1821	44	68	20	11	21	33	SEDANG
1822	34	54	28	8	25	29	SEDANG
1823	53	75	25	15	23	44	SEDANG
1824	60	87	28	19	30	53	SEDANG

1825 rows × 7 columns

Gambar 3. Hasil seleksi atribut

Setelah menghilangkan atribut yang tidak diperlukan, dilanjutkan dengan proses *data cleansing* untuk mengisi atribut yang memiliki nilai kosong atau bernilai *null*. *Cleansing* data juga merupakan suatu proses analisis kualitas dari suatu data dengan cara mengubah, mengoreksi, atau menghapus data-data yang salah, tidak lengkap, tidak akurat, atau memiliki format yang salah dalam basis data guna menghasilkan data berkualitas tinggi. *Data cleansing* juga biasa disebut dengan data cleaning atau data scrubbing [4]. Pada proses *data cleansing* terdapat beberapa tahap antara lain mengubah seluruh data yang kosong ke dalam bentuk *null*. Hal ini dilakukan untuk memudahkan proses pengisian data yang kosong dengan data yang baru. Data baru dapat berupa *mean* atau *median*. Untuk data baru hanya dimasukkan ke dalam baris data atribut, sedangkan untuk baris data kategori yang bernilai kosong atau null akan di *drop* atau dihapus. Berikut potongan kode untuk *data cleansing* dalam python serta data yang sudah dilakukan *cleansing.*

```
# dalam dataset ini terdapat value kosong seperti "---", maka terlebih dahulu kita ganti dengan null
df_cleaned = df_cleaned.replace('---', np.nan)
df_cleaned = df_cleaned.replace('nan', np.nan)
df_cleaned = df_cleaned.replace('TIDAK ADA DATA', np.nan)
```

Gambar 4. Kode python untuk mengganti nilai kosong dengan *null*

```
#numerik
df_cleaned = round(df_cleaned.fillna(df_cleaned.median(numeric_only=True)))

#kategori
df_cleaned = df_cleaned.dropna(axis=0)
```

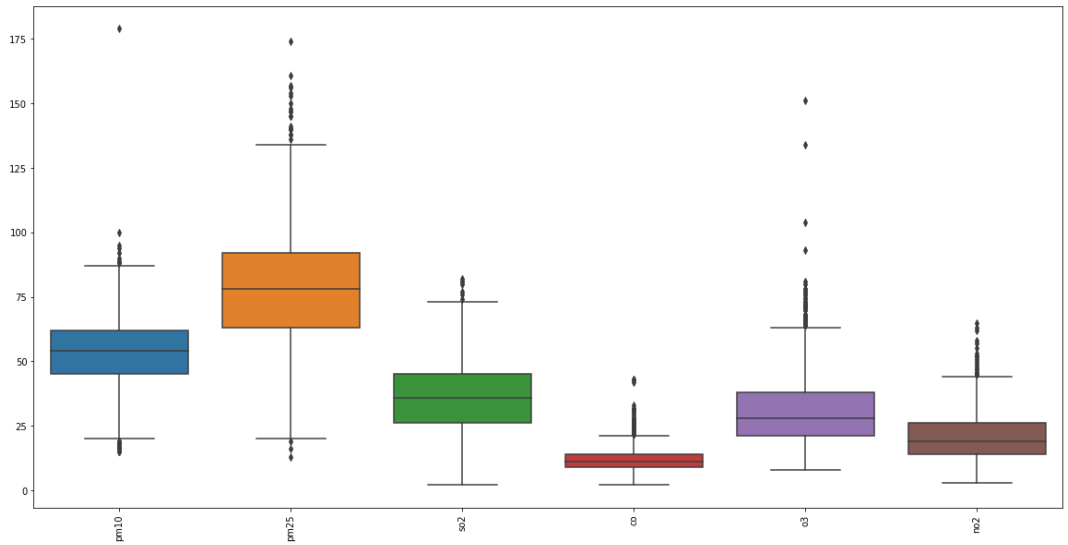
Gambar 5. Kode python untuk mengisi *null* dengan *median*

Tabel 3. Data yang telah di *cleansing*

pm10	pm25	so2	co	o3	no2	categori
37	53	20	12	25	4	BAIK
33	46	18	20	64	6	SEDANG
42	58	15	10	62	3	SEDANG
27	53	14	7	31	4	BAIK
41	47	15	9	28	7	BAIK
33	54	12	19	46	7	BAIK
27	61	13	11	36	7	BAIK
36	24	18	19	28	7	BAIK
46	25	17	26	27	12	BAIK
19	53	20	8	34	5	BAIK
33	73	21	10	34	9	BAIK
25	52	22	6	38	6	BAIK
25	62	20	5	27	9	BAIK

Setelah proses *cleansing*, dilanjutkan dengan proses menghilangkan *outliers* pada dataset. *Outliers* adalah data yang muncul dengan karakteristik yang sangat jauh berbeda dengan data lainnya dan muncul sebagai nilai ekstrim baik untuk sebuah variabel tunggal atau variabel kombinasi[5]. Pada penelitian ini proses menghilangkan *outliers* menggunakan *z-score* dan untuk deteksi *outliers* menggunakan *boxplot*. Berikut tahapan dalam menghilangkan *outliers*.

a. Tampilkan dataset sebelum penghapusan nilai *outliers* dengan *boxplot*



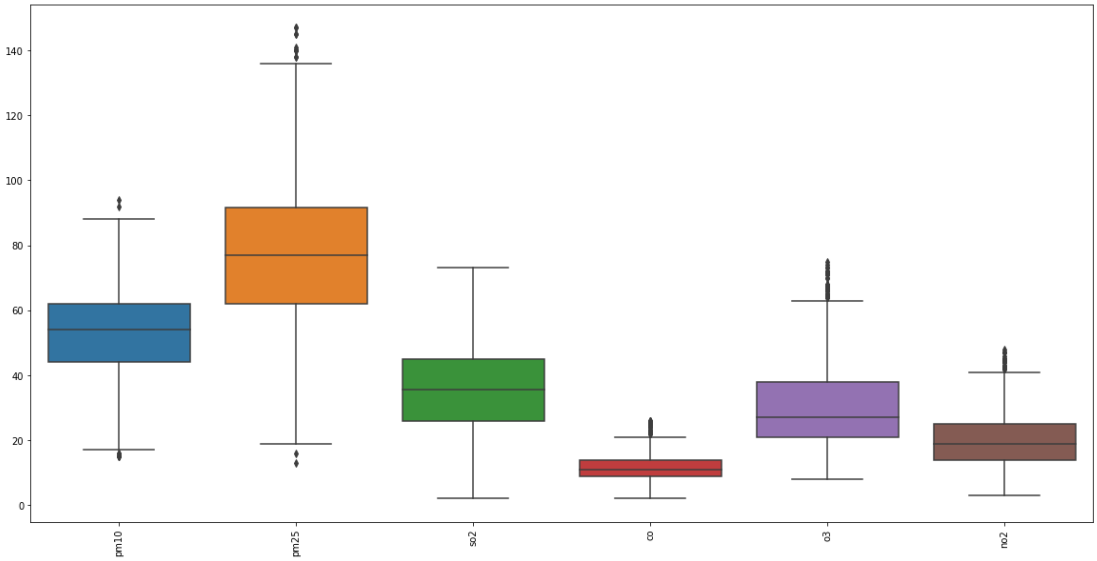
Gambar 6. *Boxplot* nilai *outliers*

b. Seleksi nilai *outliers* dengan *z-score* dan tampilkan dataset setelah penghapusan

```
features = df_cleaned.drop(['categori'], axis=1) # dropping target

z_scores = np.abs(stats.zscore(features, nan_policy='omit'))
outliers_threshold = 3
mask = (z_scores < outliers_threshold).all(axis=1)
df_cleaned = df_cleaned[mask]
print(df_cleaned)
```

Gambar 7. Kode *z-score* untuk seleksi nilai *outlier*



Gambar 8. *Boxplot* setelah seleksi *outliers*

3.3. Pengolahan Data

Pada tahap selanjutnya data akan dibagi menjadi 2 kelompok yaitu data training dan data testing. Data yang digunakan sebagai data training sebanyak 60% dan data data testing sebanyak 40%. Namun sebelum itu, data atribut kategori terlebih dahulu di transformasikan ke dalam tipe data numerik agar memiliki nilai yang sama dengan atribut lainnya. Berikut potongan kode untuk mengubah tipe data atribut kategori ke dalam tipe data numerik.

```
label_encoder = LabelEncoder()
for i in ['categori']:
    df_cleaned[i] = label_encoder.fit_transform(df_cleaned[i])
```

Gambar 9. Kode untuk mengubah tipe data kategori ke numerik

Kemudian dilanjutkan dengan proses normalisasi atau *data scaling* yaitu proses agar membuat beberapa data memiliki rentang nilai yang sama. Pada penelitian ini menggunakan MinMaxScaler untuk proses *data scaling*. Berikut tampilan dalam bentuk kode python.

```
features = df_cleaned.drop(['categori'], axis=1) # dropping target

min_max_scaler = MinMaxScaler()
df_transformed = df_cleaned.copy()
for feature in features:
    df_transformed[[feature]] = min_max_scaler.fit_transform(df_cleaned[[feature]])
```

Gambar 10. *Data Scaling* dengan MinMaxScaler

Setelah normalisasi, data dipisah menjadi 2 bagian seperti yang telah disebutkan yaitu 60% menjadi data latih dan 40% menjadi data uji. Berikut potongan kode untuk membagi dataset.

Pisahkan dataset sebagai variabel respons dan variabel fitur

```
[37]  1 X = df_transformed.iloc[:, 0:6].values #atribut
      2 y = df_transformed.iloc[:, 6].values #kategori
```

Train and Test splitting of data

```
[38]  1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.6, random_state = 42)
```

Gambar 11. Proses pembagian dataset menjadi data latih dan data uji

3.4. Proses Klasifikasi

Kemudian proses dilanjutkan ke proses klasifikasi, dalam proses klasifikasi memerlukan beberapa library antara lain:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.svm import SVC
```

Gambar 12. Import library untuk proses klasifikasi

Kemudian dilanjutkan dengan memanggil algoritma pertama yaitu SVC (Support Vector Classifier).

```
svc = SVC()
svc.fit(X_train, y_train)
pred_svc = svc.predict(X_test)
```

Gambar 13. SVC (Support Vector Classifier)

Dilanjutkan dengan memanggil *classification report* untuk melihat hasil akurasi, presisi, recall, dan f1-score yang dihasilkan.

```
print(classification_report(y_test, pred_svc))
```

	precision	recall	f1-score	support
0	0.87	0.61	0.72	123
1	0.91	0.98	0.94	774
2	0.98	0.81	0.89	144
accuracy			0.91	1041
macro avg	0.92	0.80	0.85	1041
weighted avg	0.92	0.91	0.91	1041

Gambar 14. Hasil klasifikasi SVC

Kemudian dilanjutkan dengan memanggil algoritma kedua yaitu Decision Tree Classifier.

```
1 dtc = DecisionTreeClassifier()
2 dtc.fit(X_train, y_train)
3 pred_dtc = dtc.predict(X_test)
```

Gambar 15. Decision Tree Classifier

Terakhir panggil *classification report* untuk melihat akurasi, presisi, recall, dan f1-score yang dihasilkan.

```
1 print(classification_report(y_test, pred_dtc))
```

	precision	recall	f1-score	support
0	0.90	0.85	0.88	123
1	0.98	0.98	0.98	774
2	1.00	1.00	1.00	144
accuracy			0.97	1041
macro avg	0.96	0.95	0.95	1041
weighted avg	0.97	0.97	0.97	1041

Gambar 16. Hasil Decision Tree Classifier

4. Conclusion

Berdasarkan hasil pengujian, algoritma *decision tree* lebih unggul dibandingkan dengan algoritma *support vector machine* dalam hal klasifikasi kualitas udara. Algoritma *SVM* mendapatkan hasil yaitu akurasi sebesar 91%, *precision* sebesar *87%, recall* sebesar *61%,* dan *f1-score* sebesar *72%, sedangkan* metode *Decision Tree* menghasilkan tingkat akurasi sebesar 97%, dengan presisi sebesar 90%, recall sebesar 85%, dan f1-score sebesar 88%.

References

[1] M. A. Rizaty, “Kualitas Udara Jakarta Pagi Ini Terburuk Kedua di Dunia (Jumat, 17 Juni 2022),” 2022. https://databoks.katadata.co.id/datapublish/2022/06/17/kualitas-udara-jakarta-pagi-ini-terburuk-kedua-di-dunia-jumat-17-juni-2022 (accessed Oct. 03, 2022).

[2] Peraturan Pemerintah RI, “Peraturan Menteri Lingkungan Hidup dan Kehutanan Republik Indonesia No 14 Tahun 2020 tentang Indeks Standar Pencemaran Udara,” pp. 1–16, 2020.

[3] A. Hermawan, “SPKU: Sistem Prediksi Kualitas Udara (Studi Kasus: Dki Jakarta),” 2019, [Online]. Available: http://eprints.uty.ac.id/3552/

[4] A. I. Sang, E. Sutoyo, and I. Darmawan, “Analisis Data Mining Untuk Klasifikasi Data Kualitas Udara DKI Jakarta Menggunakan Algoritma Decision Tree Dan Support Vector Machine,” *e-Proceeding Eng.*, vol. 8, no. 5, pp. 8954–8963, 2021.

[5] I. Habriansyah, “Perbandingan sensor untuk Fault Detection dan Replacement Sensor Temperatur Pada Penyimpanan Sementara Tepung Gandum,” *J. Tek. Mesin Sinergi*, vol. 19, no. 1, pp. 32–41, 2021, doi: 10.31963/sinergi.v19i1.2744.

halaman ini sengaja dibiarkan kosong