IMPLEMENTASI ALGORITMA C-45 UNTUK KLASIFIKASI KUALITAS PRODUKSI SUSU

PENGANTAR

Prediksi kualitas susu dapat dilakukan dengan berbagai macam cara secara komputer, salah satunya menggunakan algoritma C4.5. Untuk klasifikasi kualitas susu, maka yang dibutuhkan adalah atribut.

Salah satu algoritma yang dapat digunakan untuk klasifikasi adalah algoritma C-45. Atribut yang ada pada data milk quality, menjadi studi kasus untuk laporan akhir mata kuliah ini. Data diambil dari https://www.kaggle.com/datasets/cpluzshrijayan/milkquality berisi data kualitas susu dari produsen independen

Dengan menggunakan Algoritma C45, maka prediksi harga saham dapat dilakukan. Hasil implementasinya dianalisis untuk mengetahui seberapa akurat hasil prediksi algoritma C45 untuk studi kasus kualitas susu. Pengujian data dilakukan dengan cara uji coba sebanyak 4 kali ujicoba dengan data yang berbeda-beda (20% data, 30% data, 40%, dan 50% data) untuk melihat dan menguji secara detil akurasinya.

RUMUSAN MASALAH

1. Apakah algoritma C45 dapat melakukan klasifikasi kualitas susu?
2. Berapa tingkat akurasi dari implementasi algoritma C45 untuk prediksi kualitas susu?
3. Apa saja faktor-faktor yang mempengaruhi klasifikasi?

BATASAN MASALAH

1. Implementasi algoritma C-45 menggunakan dataset milkquality berisi 1059 data
2. Ada 8 atribut (kolom) yang digunakan dalam implementasi algoritma ini yaitu pH, Temperature,Taste,Odor, Fat, Turbidity, dan Grade dengan Grade sebagai target
3. Uji coba dilakukan sebanyak 4 kali percobaan dengan bermacam jumlah train dan test size untuk melihat akurasi hasil prediksi algoritma
4. Implementasi menggunakan bahasa Python, dengan library seaborn, sklearn, dan matplotlib

DESAIN

Desain dari implementasi algoritma ini menggunakan sample 40 data head untuk dibahas secara detil dalam algoritma C45, tolong dicermati desain tidak akan 100% sama dengan implementasi. Desain hanya akan menjelaskan cara kerja algortima secara manual.

Secara umum algoritma C4.5 untuk membangun pohon keputusan adalah sebagai berikut :

- Pilih atribut sebagai akar.
- Buat cabang untuk tiap-tiap nilai.
- Bagi kasus dalam cabang.
- Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Rumus yang digunakan adalah entrophy :

$E(A) = \sum_{j-1}^{v} \frac{S_{1j}+\cdots+S_{mj}}{s} (S_{1j,\ldots,}S_{mj})$, dimana A adalah attribute dengan nilai 1 sampai dengan v dan dapat dibagi menjadi beberapa partisi S dengan subset(1 ... m).

Dan Information Gain rumus gain dari atribut A:

$Gain(A) = I(S1, S2, \ldots, Sm) - E(A)$ atau dengan kata lain Gain adalah reduksi nilai dari entrophy.

Data yang kita miliki mempunyai kolom berikut Kita menggunakan milk dataset untuk menentukan kualitas suatu susu berdasarkan parameter parameter tertentu. Kolom 1-7 merupakan atribut atau fitur dan kolom ke 8 merupakan target class, yaitu apakah suatu susu berkualitas low, medium atau high

- pH
  nilai pH dari susu segar normalnya berkisar antara 6.4 hingga 6.8, bergantung pada sumber dari susu tersebut.
- Temperature
  Idealnya, susu disimpan di dalam kulkas pada suhu 40 derajat Fahrenheit atau dibawahnya. Menyimpan susu pada suhu tersebut dapat memperpanjang masa simpan dan memaksimalkan rasa.
- Taste
- Odor
  Susu yang berkualiatas baik memiliki aroma yang segar tanpa ada perbedaan rasa setelahnya.
- Fat
  Jenis jenis susu bervariasi berdasarkan kadar lemak yang terdapat di dalamnya, seperti whole milk (3.25% milk fat), reduced-fat milk (2%), low-fat milk(1%) dan fat-free milk. Informasi itu biasanya tercetak pada kemasan.

- Turbidity
  Kekeruhan pada susu dipengaruhi oleh kadar lemak yang terkandung dalam susu.
- Color

  Warna keputihan susu

pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
6.6	35	1	0	1	0	254	high
6.6	36	0	1	0	1	253	high
8.5	70	1	1	1	1	246	low
9.5	34	1	1	0	1	255	low
6.6	37	0	0	0	0	255	medium
6.6	37	1	1	1	1	255	high
5.5	45	1	0	1	1	250	low
4.5	60	0	1	1	1	250	low
8.1	66	1	0	1	1	255	low
6.7	45	1	1	0	0	247	medium
6.7	45	1	1	1	0	245	medium
5.6	50	0	1	1	1	255	low
8.6	55	0	1	1	1	255	low
7.4	90	1	0	1	1	255	low
6.8	45	0	1	1	1	255	high
6.5	38	1	0	0	0	255	medium
4.7	38	1	0	1	0	255	low
3	40	1	1	1	1	255	low
9	43	1	0	1	1	250	low
6.8	40	1	0	1	0	245	medium
6.6	45	0	1	1	1	250	high
6.5	36	0	0	1	0	255	medium
4.5	38	0	1	1	1	255	low
6.6	45	1	1	1	1	245	high
6.8	35	1	0	1	0	246	medium
6.5	36	0	1	1	0	253	high
8.5	70	0	0	0	0	246	low
6.6	34	0	0	0	1	240	medium
6.5	37	0	0	0	0	245	medium
6.6	37	1	0	1	0	255	high
6.4	45	0	1	0	0	240	medium
6.5	40	0	1	0	1	250	low
6.8	42	1	1	1	1	255	high
6.7	41	1	0	0	0	247	medium
6.7	50	1	1	1	0	245	low
6.8	45	0	1	1	1	255	high
8.6	55	0	1	0	0	255	low
7.4	65	0	0	0	0	255	low
6.8	41	0	0	0	0	255	medium
6.5	38	1	1	1	1	255	high

Yang Pertama kita lakukan adalah mengambil 40 data head dari dataset milknew.csv

Bisa dilihat diatas data yang ingin kita jadikan target calss memiliki 3 tipe yaitu **High,Medium dan Low** untuk kemudahan pembuatan desain agar tidak terlalu bercabang khusus dalam desain saya memasukan data medium kedalam high jadi class target khusus untuk desain ini memiliki 2 tipe yaitu **High dan Low**

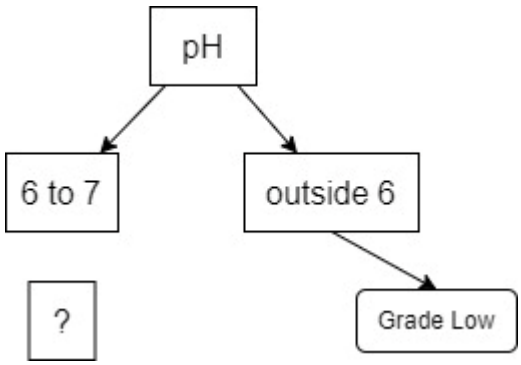
pH	Tempratu	Taste	Odor	Fat	Turbidity	Colour	Grade
6,6	35	1	0	1	0	254	high
6,6	36	0	1	0	1	253	high
8,5	70	1	1	1	1	246	low
9,5	34	1	1	0	1	255	low
6,6	37	0	0	0	0	255	high
6,6	37	1	1	1	1	255	high
5,5	45	1	0	1	1	250	low
4,5	60	0	1	1	1	250	low
8,1	66	1	0	1	1	255	low
6,7	45	1	1	0	0	247	high
6,7	45	1	1	1	0	245	high
5,6	50	0	1	1	1	255	low
8,6	55	0	1	1	1	255	low
7,4	90	1	0	1	1	255	low
6,8	45	0	1	1	1	255	high
6,5	38	1	0	0	0	255	high
4,7	38	1	0	1	0	255	low
3	40	1	1	1	1	255	low
9	43	1	0	1	1	250	low
6,8	40	1	0	1	0	245	high
6,6	45	0	1	1	1	250	high
6,5	36	0	0	1	0	255	high
4,5	38	0	1	1	1	255	low
6,6	45	1	1	1	1	245	high
6,8	35	1	0	1	0	246	high
6,5	36	0	1	1	0	253	high
8,5	70	0	0	0	0	246	low
6,6	34	0	0	0	1	240	high
6,5	37	0	0	0	0	245	high
6,6	37	1	0	1	0	255	high
6,4	45	0	1	0	0	240	high
6,5	40	0	1	0	1	250	low
6,8	42	1	1	1	1	255	high
6,7	41	1	0	0	0	247	high
6,7	50	1	1	1	0	245	low
6,8	45	0	1	1	1	255	high
8,6	55	0	1	0	0	255	low
7,4	65	0	0	0	0	255	low
6,8	41	0	0	0	0	255	high
6,5	38	1	1	1	1	255	high

Data yang berjumlah 40 ini memiliki jumlah high 23 dan low 17.Setelah itu data dimasukan

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
TOTAL DATA		40	17	23	0,984		
pH	6 to 7	25	2	23	0,402	**0,732**	0,251
	outside 6 to 7	15	15	0	0,000		0,000
Temp	34 to 40	18	5	13	0,852	0,053	0,384
	> 40	22	12	10	0,994		0,547
Taste	0 (bad)	19	8	11	0,982	0,000	0,466
	1 (good)	21	9	12	0,985		0,517
odor	0 (bad)	18	7	11	0,964	0,003	0,434
	1 (good)	22	10	12	0,994		0,547
Fat	0 (low)	14	5	9	0,940	0,007	0,329
	1 (high)	26	12	14	0,996		0,647
Turbidity	0 (low)	19	5	14	0,831	0,072	0,395
	1 (high)	21	12	9	0,985		0,517
Colour	255	20	10	10	1,000	0,017	0,500
	< 255	20	7	13	0,934		0,467

ke iterasi pertama dan menghitung information gain

Dari data iterasi pertama yang memiliki gain tertinggi ialah pH maka kita buat dalam langkah pertama dalam tree kita



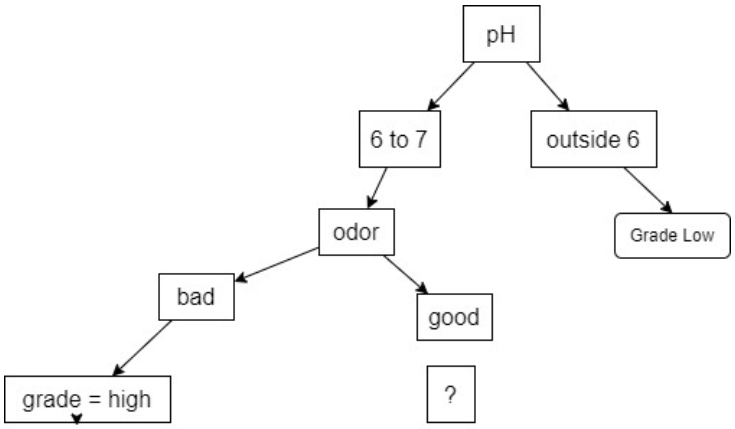
Lalu kita lanjutkan didalam pH 6-7 untuk mencari iterasi ke 2

didalam iterasi ke 2 ini gain tertinggi ada dibagian odor lalu kita masukan ke tree

lalu lanjutkan ke iterasi tiga berisi data odor good dengan ph 6-7

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
pH 6 - 7		25	2	23	0,402		
Temp	34 to 40	15	1	14	0,353	0,003	0,212
	> 40	10	1	9	0,469		0,188
Taste	0 (bad)	12	1	11	0,414	0,000	0,199
	1 (good)	13	1	12	0,391		0,203
odor	0 (bad)	11	0	11	0,000	**0,071**	0,000
	1 (good)	14	2	12	0,592		0,331
Fat	0 (low)	10	1	9	0,469	0,003	0,188
	1 (high)	15	1	14	0,353		0,212
Turbidity	0 (low)	15	1	14	0,353	0,003	0,212
	1 (high)	10	1	9	0,469		0,188
Colour	255	10	0	10	0,000	0,062	0,000
	< 255	15	2	13	0,567		0,340

Tree iterasi ke 2

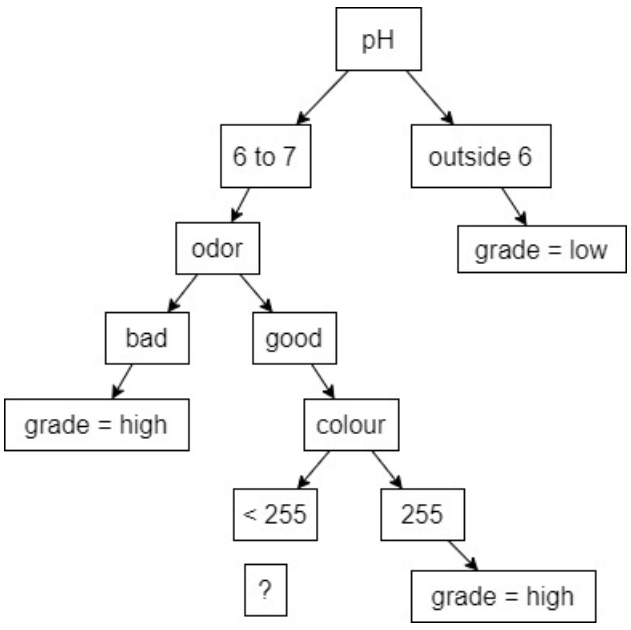


JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Odor Good		**14**	2	12	0,592		
Temp	34 to 40	**5**	1	4	0,722	0,010	0,258
	> 40	**9**	1	8	0,503		0,324
Taste	0 (bad)	**7**	1	6	0,592	0,000	0,296
	1 (good)	**7**	1	6	0,592		0,296
Fat	0 (low)	**4**	1	3	0,811	0,025	0,232
	1 (high)	**10**	1	9	0,469		0,335
Turbidity	0 (low)	**5**	1	4	0,722	0,010	0,258
	1 (high)	**9**	1	8	0,503		0,324
Colour	255	**5**	0	5	0,000	0,100	0,000
	< 255	**9**	2	7	0,764		0,491

Mari kita lanjutkan ke iterasi 3 yang berisi

Disini colour memiliki gain tertinggi maka dari itu kita masukan dalam tree dan menjadikan colour < 255 menjadi basis iterasi ke 4

Tree iterasi 3

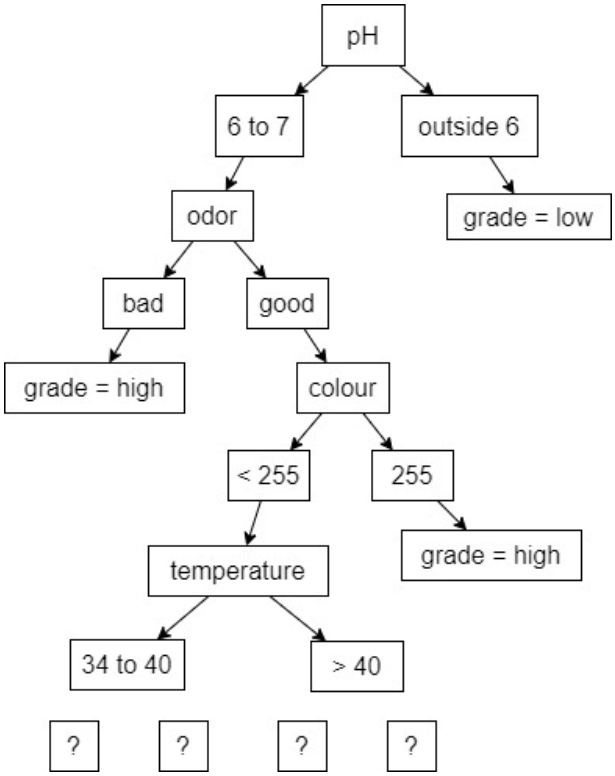


Iterasi ke 4

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Color <255		9	2	7	0,764		
Temp	34 to 40	3	1	2	0,918	0,025	0,306
	> 40	6	1	5	0,650		0,433
Taste	0 (bad)	5	1	4	0,722	0,003	0,401
	1 (good)	4	1	3	0,811		0,361
Fat	0 (low)	4	1	3	0,811	0,003	0,361
	1 (high)	5	1	4	0,722		0,401
Turbidity	0 (low)	5	1	4	0,722	0,003	0,401
	1 (high)	4	1	3	0,811		0,361

Didalam iterasi 4 ini gain tertinggi ada didalam bagian temperatur tapi bisa kita lihat bahwa didalam masing2 kategori baik 34-40 atau > dari 40 belum selesai maka kita akan memecah iterasi untuk iterasi 5 keatas

Tree iterasi ke 4



Kita pisahkan dalam iterasi 5 menjadi iterasi 5.1 yang berisi temp 34-40 dan 5.2 berisi temperature diatas 40

Mari kita mulai Iterasi ke lima

Tabel 5.1

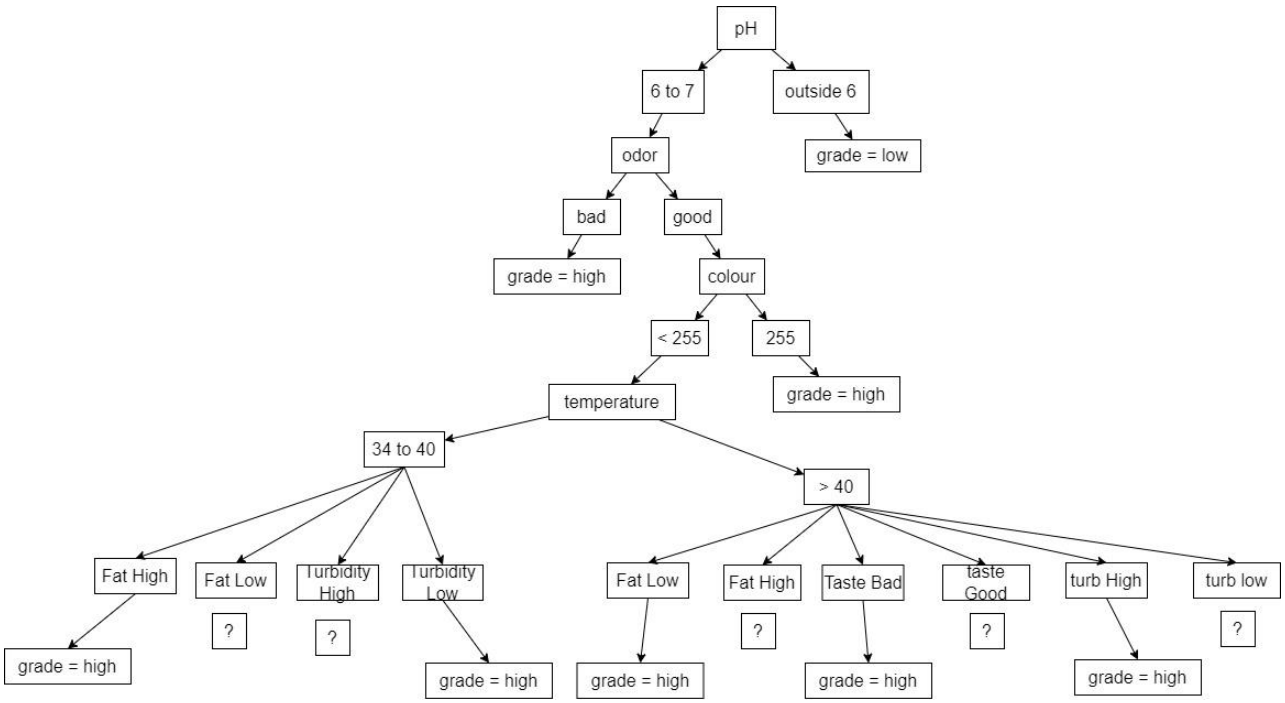
JUMLAH DATA			GRADE		INF.	GAIN	OLOM BAN
			LOW	HIGH			
Temp 34-40		**3**	1	2	0,918		
Taste	0 (bad)	**3**	1	2	0,918	0,000	0,918
	1 (good)	**0**	0	0	0,000		0,000
Fat	0 (low)	**2**	1	1	1,000	0,252	0,667
	1 (high)	**1**	0	1	0,000		0,000
Turbidity	0 (low)	**1**	0	1	0,000	0,252	0,000
	1 (high)	**2**	1	1	1,000		0,667

Nanti kita akan melanjutkan fat low dan turbidity high sedangkan dibagian temperature diatas 40 yaitu

Tabel 5.2

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Temp 40		**6**	1	5	0,650		
Taste	0 (bad)	**2**	0	2	0,000	0,109	0,000
	1 (good)	**4**	1	3	0,811		0,541
Fat	0 (low)	**2**	0	2	0,000	0,109	0,000
	1 (high)	**4**	1	3	0,811		0,541
Turbidity	0 (low)	**4**	1	3	0,811	0,109	0,541
	1 (high)	**2**	0	2	0,000		0,000

Disini kita akan iterasi lagi Taste good, fat high, dan turbidity low kita masukan hasil kedua iterasi 5 ini kedalam tree kita



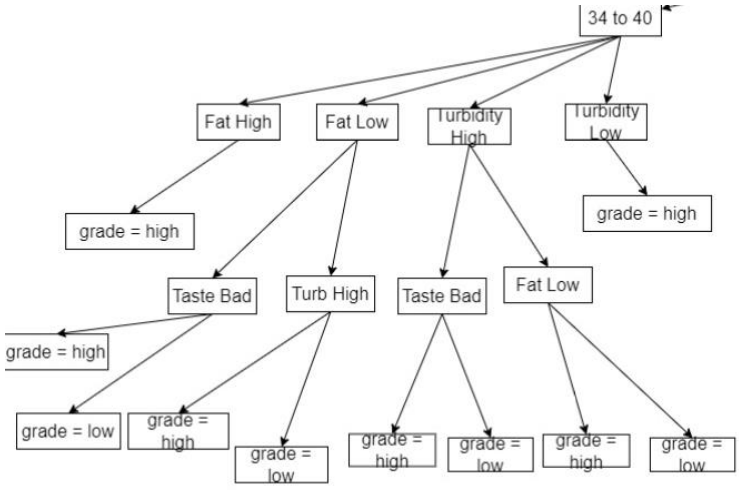
Dibagian Iterasi ke 6 ini kita akan bagi menjadi 2 bagian 6.1 berisi data yang awalnya berasal dengan temp 34-40 dan 6.2 berisi data yang berawal dengan temp diatas 40

Tabel 6.1.1

JUMLAH DATA			GRADE		INF.	GAIN	OLOM BANTU
			LOW	HIGH			
Low Fat		**2**	1	1	1,000		
Taste	0 (bad)	**2**	1	1	1,000	0,000	1,000
	1 (good)	**0**	0	0	0,000		0,000
Turbidity	0 (low)	**0**	0	0	0,000	0,000	0,000
	1 (high)	**2**	1	1	1,000		1,000

Tabel 6.1.2

JUMLAH DATA			GRADE		INF.	GAIN	OLOM BANT
			LOW	HIGH			
TurbidityHigh		**2**	1	1	1,000		
Taste	0 (bad)	**2**	1	1	1,000	0,000	1,000
	1 (good)	**0**	0	0	0,000		0,000
Fat	0 (low)	**2**	1	1	1,000	0,000	1,000
	1 (high)	**0**	0	0	0,000		0,000



Keduanya sudah selesai maka dapat kita masukan ke tree dibagian temp 34-40

Mari kita lanjutkan kebagian kanan dengan Tabel bagian 6.2

Tabel 6.2.1

JUMLAH DATA			GRADE		INF.	GAIN	)LOM BAN
			LOW	HIGH			
>40,TasteGood		4	1	3	0,811		
Fat	0 (low)	1	0	1	0,000	0,123	0,000
	1 (high)	3	1	2	0,918		0,689
Turbidity	0 (low)	3	1	2	0,918	0,123	0,689
	1 (high)	1	0	1	0,000		0,000

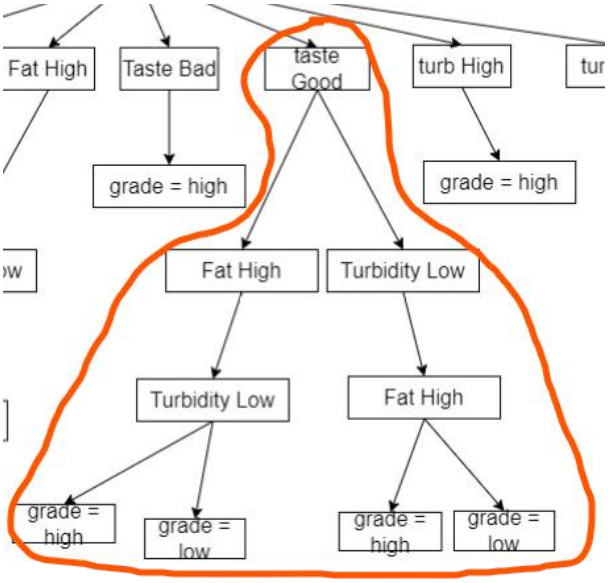
Tabel 6.2.1.1

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
>40,taste good,fathigh		3	1	2	0,918		
Turbidity	0 (low)	2	1	1	1,000	0,252	0,667
	1 (high)	1	0	1	0,000		0,000

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
>40,taste good,Turbiditylov		3	1	2	0,918		
Fat	0 (low)	1	0	1	0,000	0,252	0,000
	1 (high)	2	1	1	1,000		0,667

Tabel 6.2.1.2

Bagian 6.2.1 sudah selesai karena variabel pembanding habis lalu dimasukan ke tree



Bagian 6.2.2

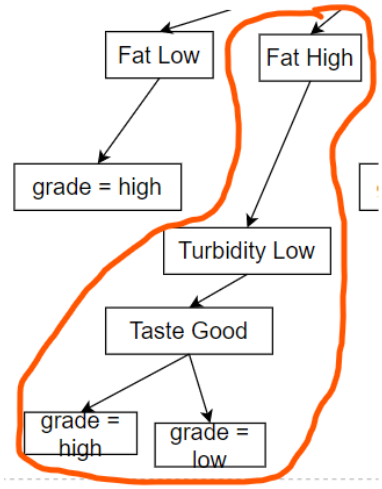
Tabel 6.2.2

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Temp > 40 Fat High		4	1	3	0,811		
Taste	Bad (0)	1	0	1	0,000	0,123	0,000
	Good (1)	3	1	2	0,918		0,689
Turbidity	Low (0)	2	1	1	1,000	**0,311**	0,500
	High (1)	2	0	2	0,000		0

Tabel 6.2.2.1

JUMLAH DATA			GRADE		INF.	GAIN	OLOM BAN
			LOW	HIGH			
Temp > 40 Fat High		**2**	1	1	1,000		
Taste	Bad (0)	0	0	0	0,000	1,000	0,000
	Good (1)	1	0	2	0,000		0,000

Bagian 6.2.2 telah selesai lanjut dengan masukan ke tree



Step selesai lalu masukan ke tree lanjutkan ke 6.2.3

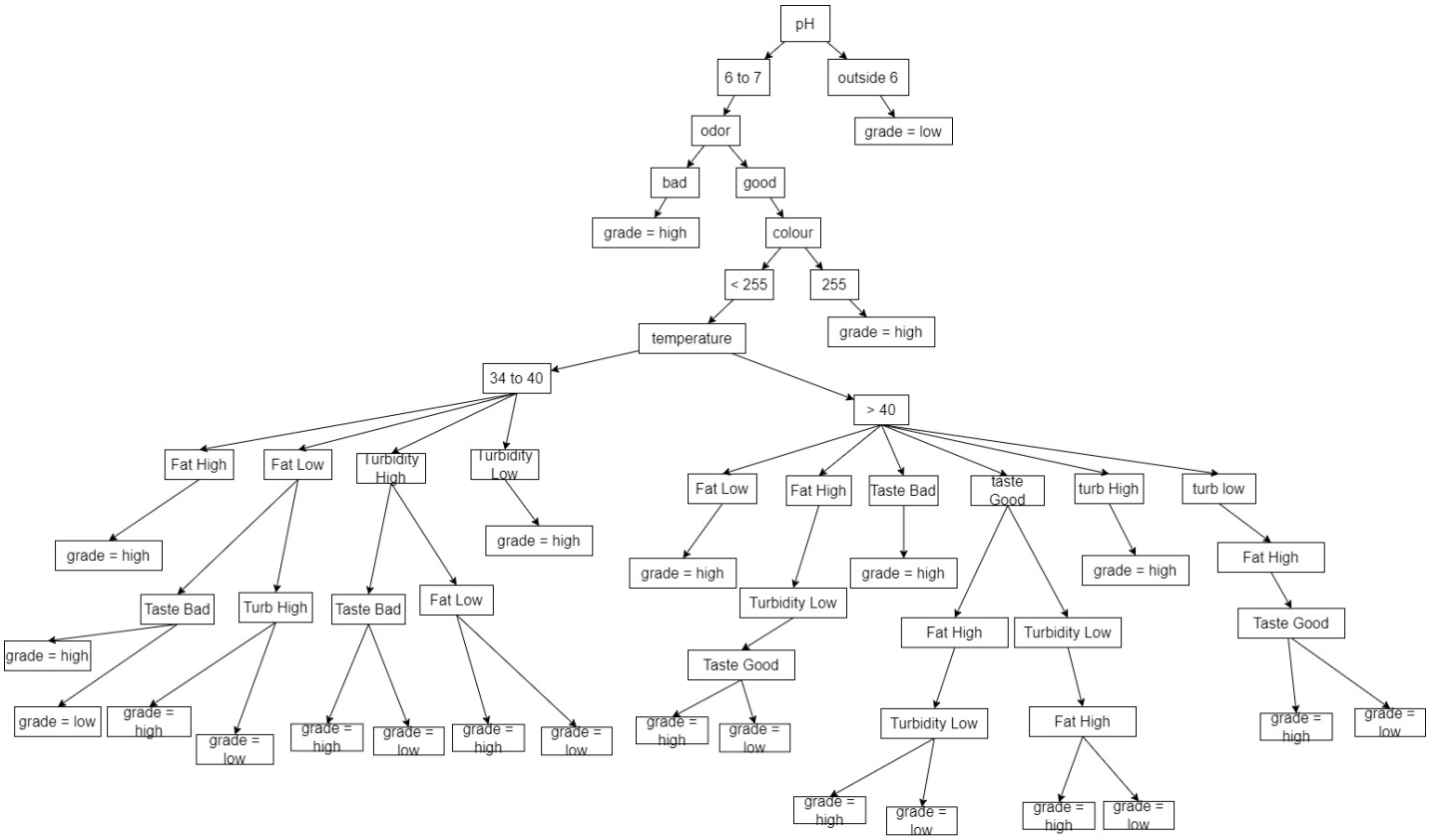
Tabel 6.2.3

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Temp > 40 Turbidity low		**4**	1	3	0,811		
Taste	Bad (0)	1	0	1	0,000	0,123	0,000
	Good (1)	3	1	2	0,918		0,689
Fat	Low (0)	2	0	2	0,000	**0,311**	0,000
	High (1)	2	1	1	1,000		0,500

Tabel 6.2.3.1

JUMLAH DATA			GRADE		INF.	GAIN	KOLOM BANTU
			LOW	HIGH			
Temp > 40 Turbidity low		**2**	1	1	1,000		
Taste	Bad (0)	0	0	0	0,000	0,000	0,000
	Good (1)	2	1	1	1,000		1,000

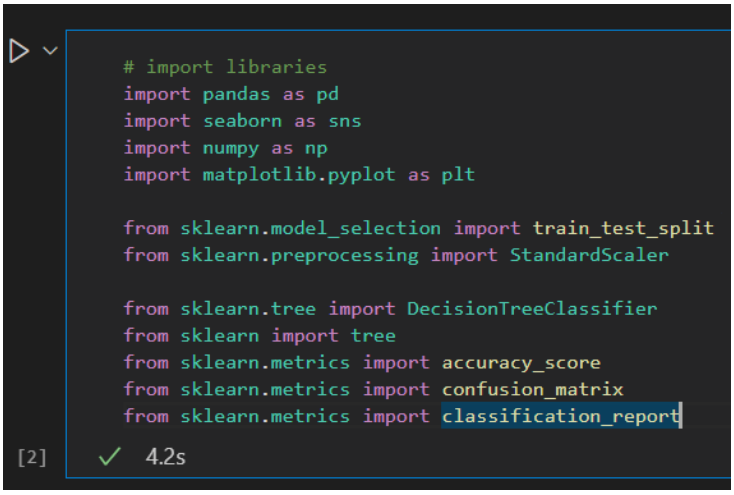
Dengan ini seluruh 40 data sudah diolah dan hasil akhir tree seperti berikut



IMPLEMENTASI

Implementasi Algoritma C4.5 dapat dilakukan sengan menggunakan library scikit-learn dengan **from sklearn.tree import DecisionTreeClassifier** dan memanggil **tree = DecisionTreeClassifier(criterion='entropy')** dengan entropy sebagai criterion nya berikut adalah langkah-langkah implementasi C4.5 dalam python

1. Langkah Pertama adalah import library yang dibutuhkan



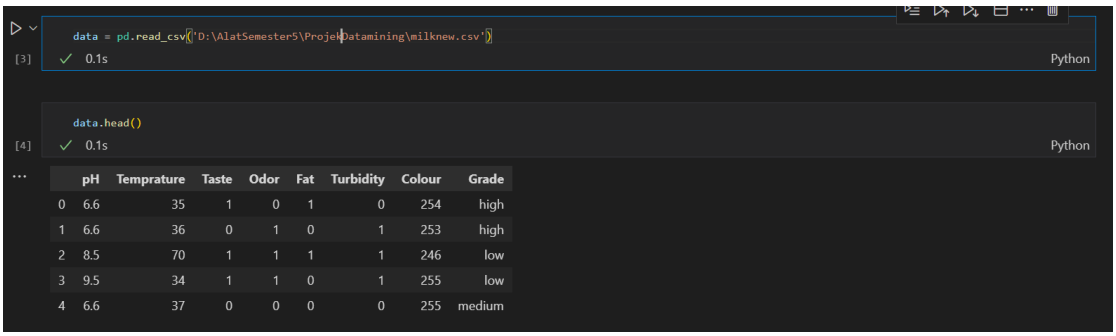
```
# import libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

[2] ✓ 4.2s

Pandas, seaborn, dan numpy digunakan untuk pengolahan data awal, train test split digunakan untuk membagi data training dan testing standard scaler untuk standarisasi data dan bagian metric untuk mengukur kualitas model

2. Pengolahan Data

Data di baca dan di test menggunakan panda



```
data = pd.read_csv("D:\AlatSemester5\ProjekDatamining\milknew.csv")
```

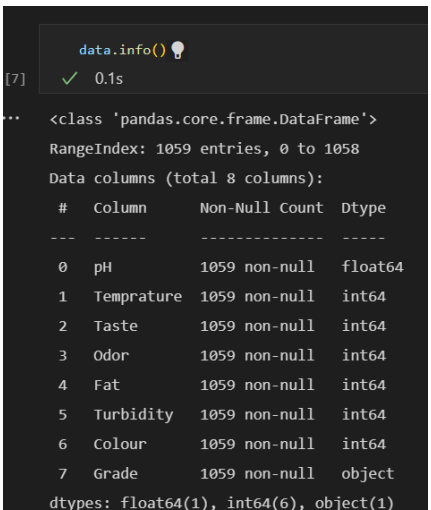
[3] ✓ 0.1s Python

```
data.head()
```

[4] ✓ 0.1s Python

	pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
0	6.6	35	1	0	1	0	254	high
1	6.6	36	0	1	0	1	253	high
2	8.5	70	1	1	1	1	246	low
3	9.5	34	1	1	0	1	255	low
4	6.6	37	0	0	0	0	255	medium

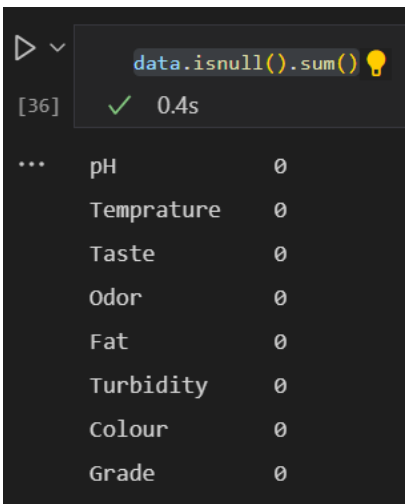
Bisa dilihat data set berhasil terbaca dan memiliki 8 kolom dengan jumlah isi data 1059 data Lalu kita cek apakah ada data kosong dikolom tersebut dengan **data.isnull().sum()**



```
data.info()
```

[7] ✓ 0.1s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1059 entries, 0 to 1058
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   pH          1059 non-null   float64
 1   Temprature  1059 non-null   int64
 2   Taste       1059 non-null   int64
 3   Odor        1059 non-null   int64
 4   Fat         1059 non-null   int64
 5   Turbidity   1059 non-null   int64
 6   Colour      1059 non-null   int64
 7   Grade       1059 non-null   object
dtypes: float64(1), int64(6), object(1)
```



```
data.isnull().sum()
```

[36] ✓ 0.4s

```
pH            0
Temprature    0
Taste         0
Odor          0
Fat           0
Turbidity     0
Colour        0
Grade         0
```

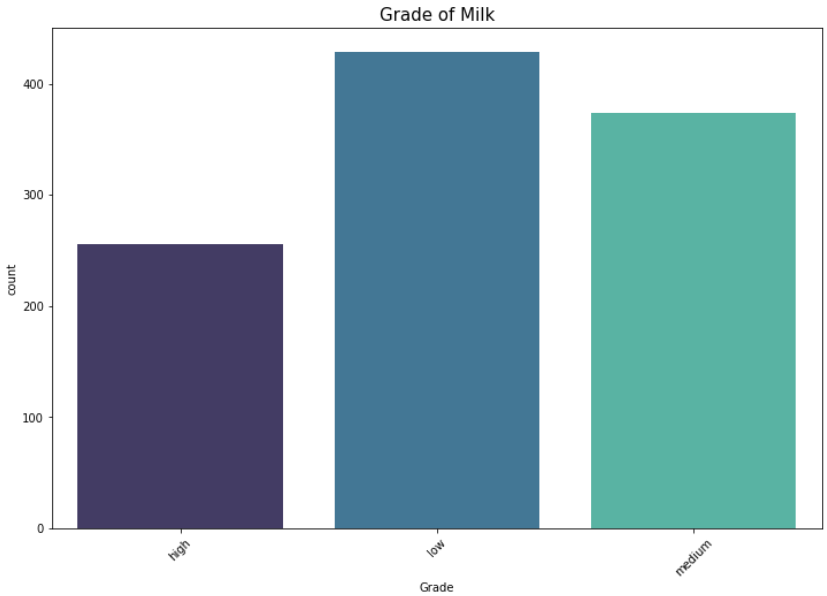
Bisa dilihat tidak ada data kosong didalam dataset lalu kita coba lihat distribusi target

```
plt.figure(figsize=(12,8))
plt.title("Grade of Milk",fontsize=15)
c1=sns.countplot(x='Grade',data=data,palette="mako")
plt.xticks(rotation=45)
plt.show()
```

✓ 1.4s

class kita yaitu **grade**

Lalu kita bisa lihat distrubusi target class kita



Dapat dilihat bahwa mayoritas grade susu dalam data ini adalah low disusul median dan high dengan rincian berikut

```
counts = data['Grade'].value_counts()
high_count = counts['high']
low_count = counts['low']
medium_count = counts['medium']
print("High:",high_count,"Low:",low_count,"Medium:",medium_count)
```

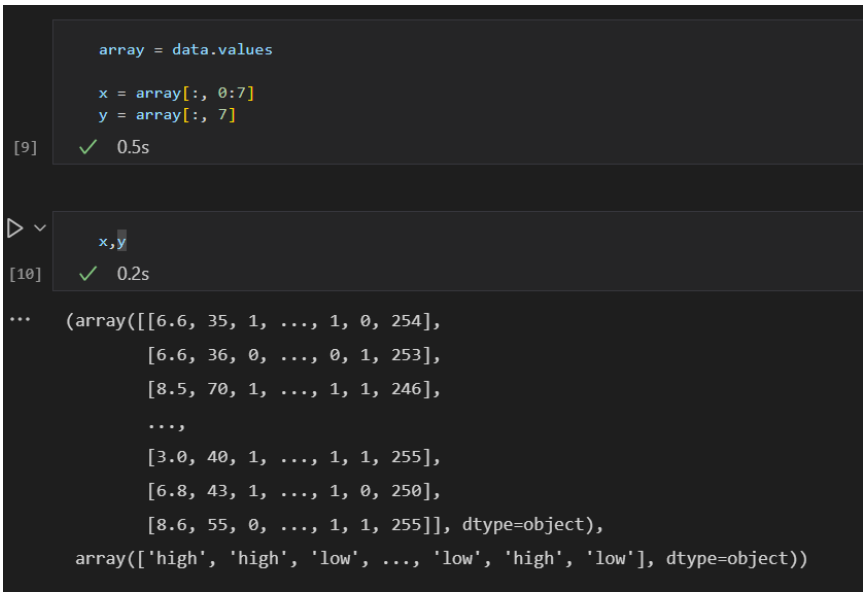
[43] ✓ 0.7s

```
High: 256 Low: 429 Medium: 374
```

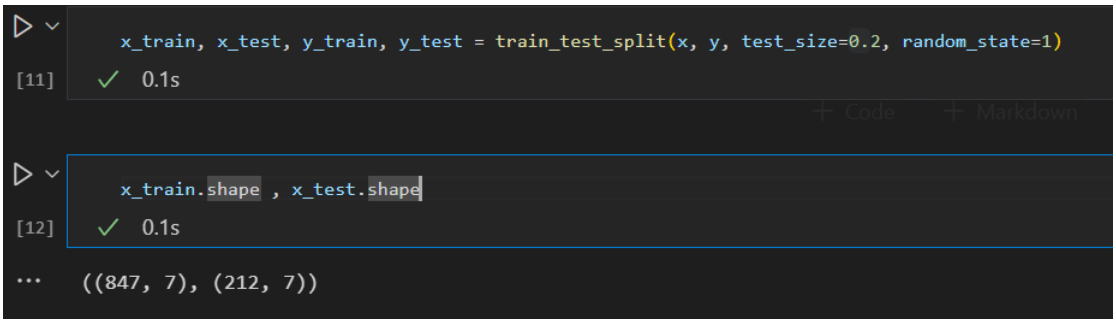
Data bisa dilihat sudah siap untuk diolah lebih lanjut ke langkah selanjutnya

3. Pemecahan data

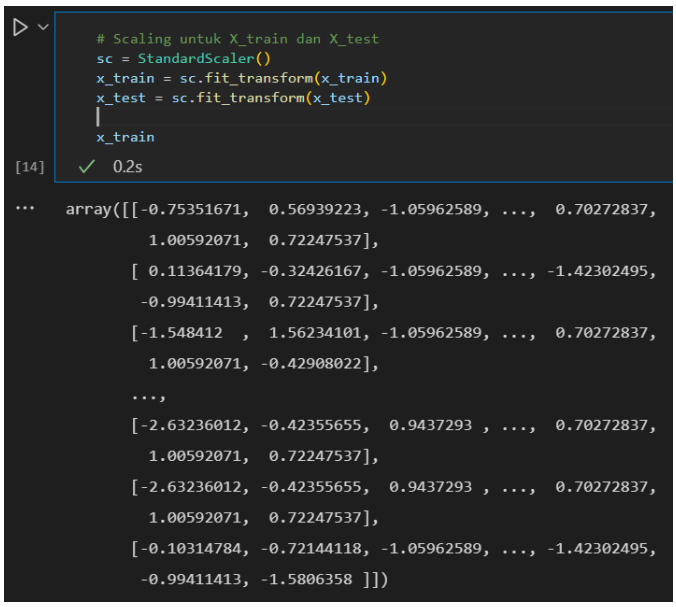
Langkah awal adalah memecah data target dengan data lainya dengan kode berikut



x berisi fitur fitur yang akan digunakan yaitu kolom 1-7, sedangkan y berisi target class, yaitu grade susu setelah itu kita memecah data untuk training dan testing dibagian ini data dibagi 80% untuk training dan 20% untuk testing



pembagian data lebih rinci 847 data untuk training dan 212 untuk testing sebelum masuk lebih dalam kita perlu menormalisasi data agar data dialah dengan benar di dalam algoritma. Disini kita menggunakan **StandardScaler fit.transform**



Data yang sudah di normalisasi dengan scaling dimasukan ke dalam algoritma c4.5 dengan DecisionTreeClassifier

4. Visualisasi Tree

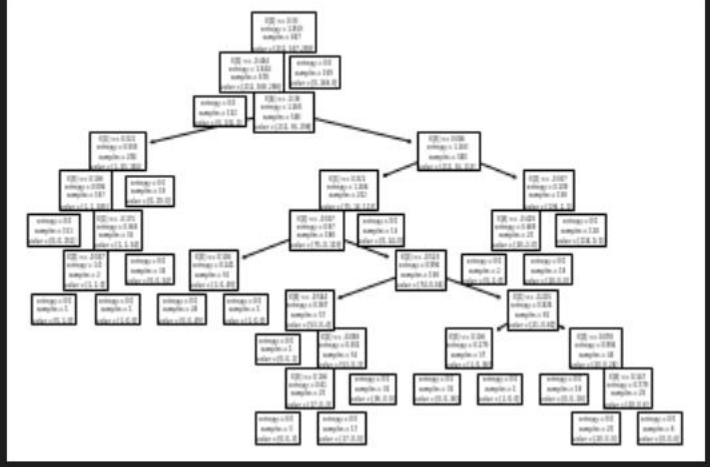
```
[15]
clf_en = DecisionTreeClassifier(criterion='entropy', random_state=1)

# fit ke data training
clf_en.fit(x_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=1)
```

Classifier yang sudah kita fit ke data training dapat kita panggil dengan line berikut

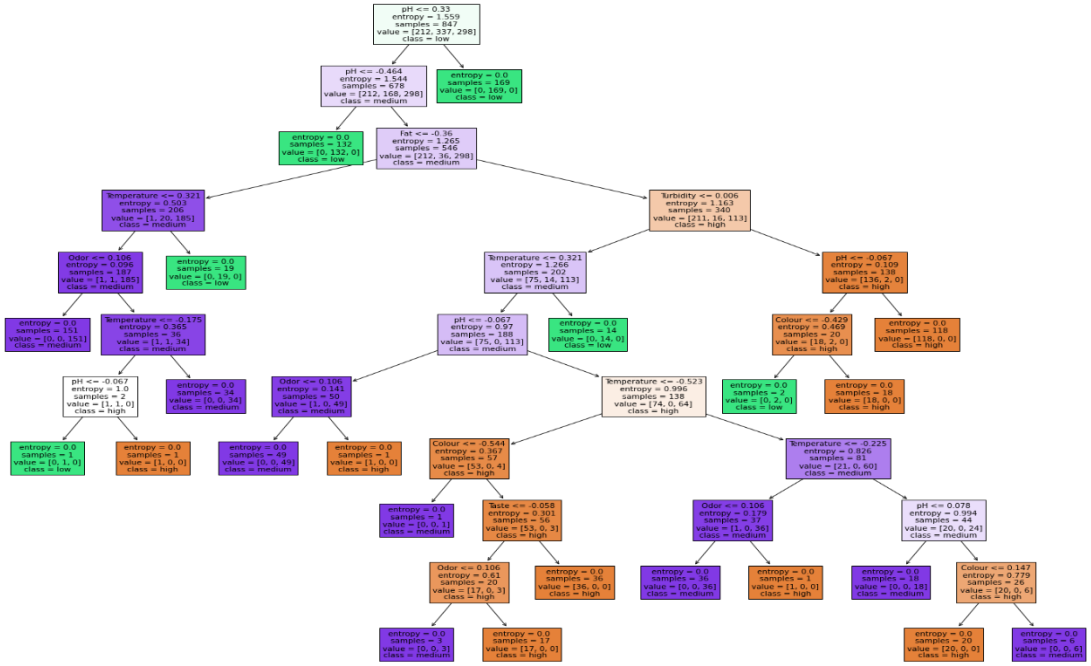
tree sudah berhasil dibuat namun tidak jelas kita bisa buat tree lebih jelas dengan modifikasi dalam plot.tree

```
[16]
tree.plot_tree(clf_en)
y_pred = clf_en.predict(x_test)
```

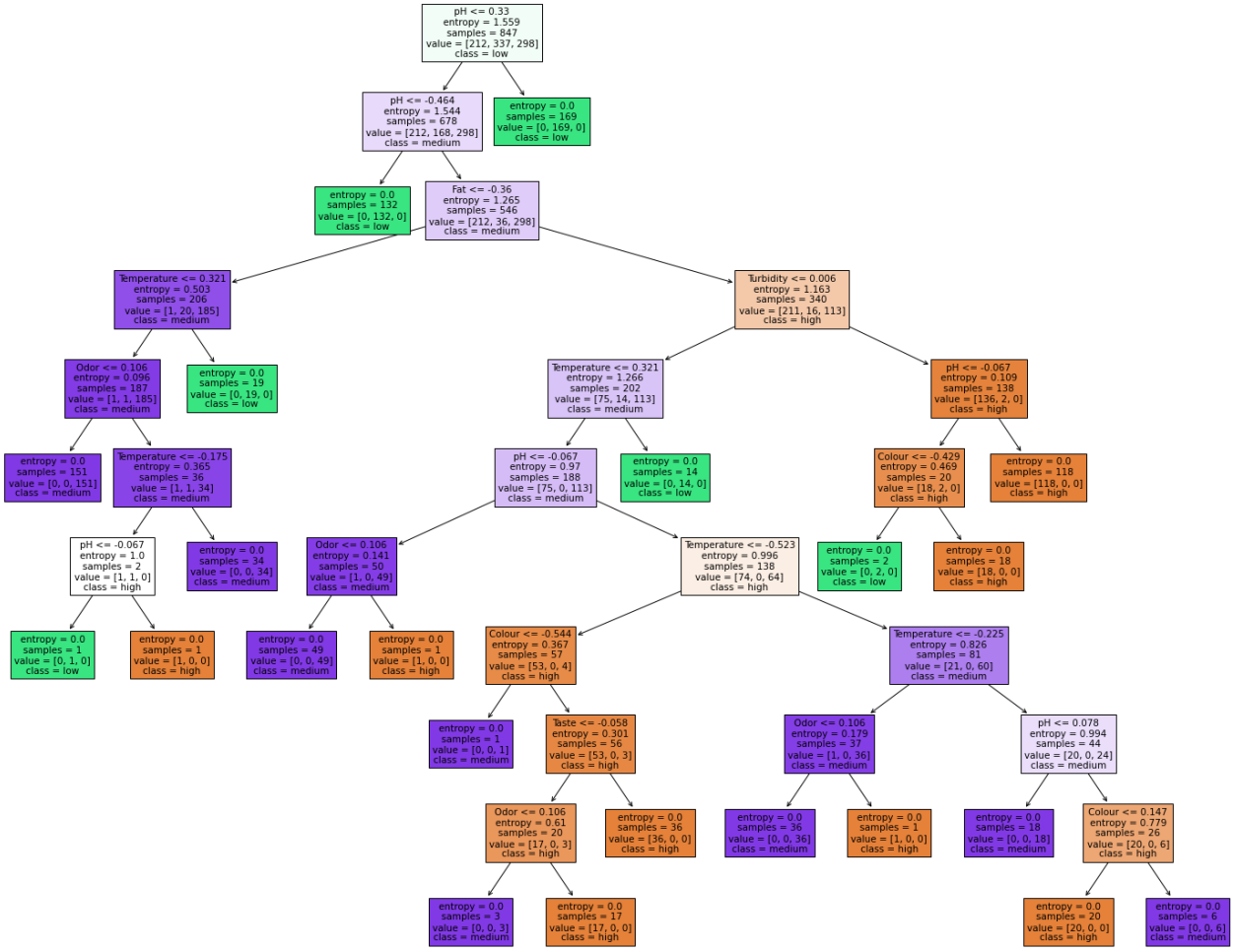


Tambahan feature_names berisi fitur dalam tree dan label_names berisi isi target class sebelumnya yaitu grade, figsize digunakan untuk membesarkan ukuran hasil tree bisa dilihat seperti dibawah ini

```
[17]
feature_names = ['pH', 'Temperature', 'Taste', 'Odor', 'Fat', 'Turbidity', 'Colour']
label_names = ['high', 'low', 'medium']
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(clf_en,
                   feature_names=feature_names,
                   class_names=label_names,
                   filled=True)
```



Versi lebih jelas dari tree diatas



Setelah model tree dibuat kita dapat menilai model dengan classification report, accuracy score, dan confusion matrix kita akan menggunakan data ini untuk analisa

```
akurasi = accuracy_score(y_test, y_pred)
print("Akurasi = ",akurasi*100)
```

✓ 0.4s

```
Akurasi =  96.22641509433963
```

```
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
```

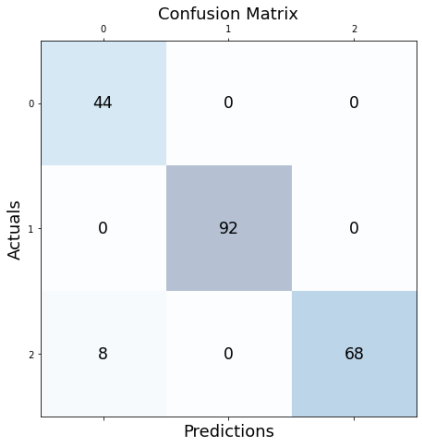
✓ 0.6s

	precision	recall	f1-score	support
high	0.85	1.00	0.92	44
low	1.00	1.00	1.00	92
medium	1.00	0.89	0.94	76
accuracy			0.96	212
macro avg	0.95	0.96	0.95	212
weighted avg	0.97	0.96	0.96	212

```
fig, ax = plt.subplots(figsize=(7.5, 7.5))
ax.matshow(cm, cmap=plt.cm.Blues, alpha=0.3)
for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        ax.text(x=j, y=i,s=cm[i, j], va='center', ha='center', size='xx-large')

plt.xlabel('Predictions', fontsize=18)
plt.ylabel('Actuals', fontsize=18)
plt.title('Confusion Matrix', fontsize=18)
plt.show()
```

✓ 0.2s

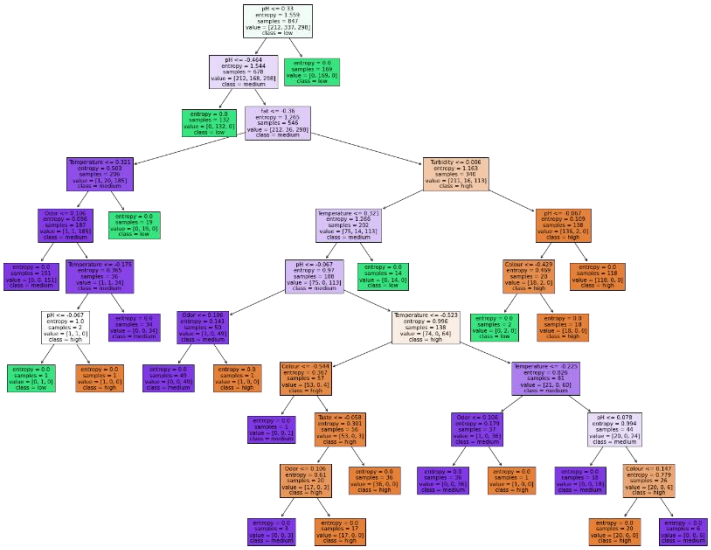
dibawah sebagaimana implementasi nya bisa dilihat seperti ini



Secara umum, precision yang tinggi menunjukkan bahwa model baik dalam menghindari false positives, sedangkan recall yang tinggi menunjukkan bahwa model baik dalam mendeteksi semua instance positif. F1-score merupakan keseimbangan antara precision dan recall, **Hasil dari penilaian model ini akan digunakan dalam analisis**

ANALISIS & PENGUJIAN

Berikut adalah hasil dari 4 kali implementasi diatas dengan berbagai ukuran train dan test:

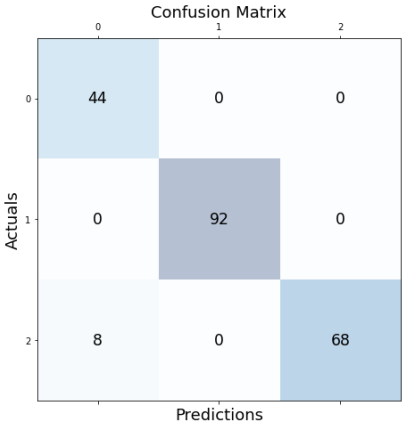


Tree dengan 80% training data dan 20% test

	precision	recall	f1-score	support
high	0.85	1.00	0.92	44
low	1.00	1.00	1.00	92
medium	1.00	0.89	0.94	76
accuracy			0.96	212
macro avg	0.95	0.96	0.95	212
weighted avg	0.97	0.96	0.96	212

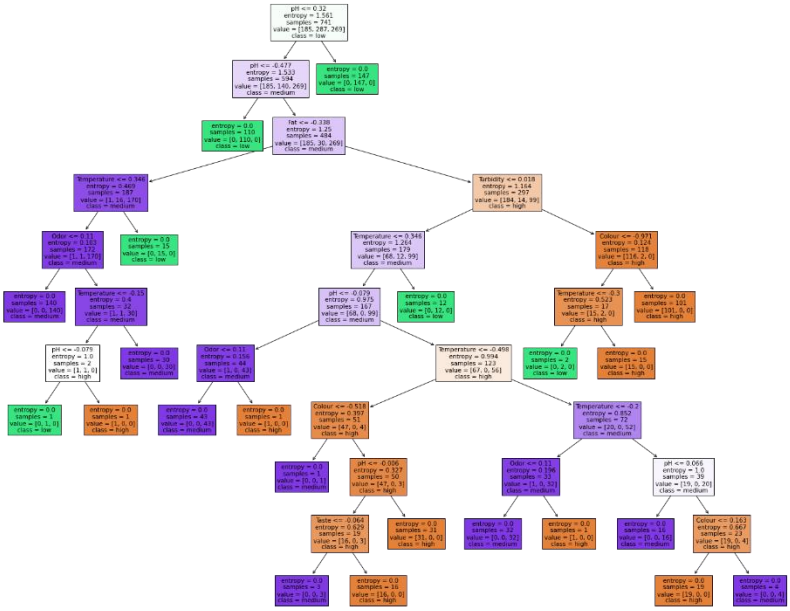
```
[19]  akurasi = accuracy_score(y_test, y_pred)
      print("Akurasi = ",akurasi*100)

...   Akurasi =  96.22641509433963
```

Classification Report Tree 80/20 beserta tingkat akurasi



Confusion Matrix dari tree 80/20

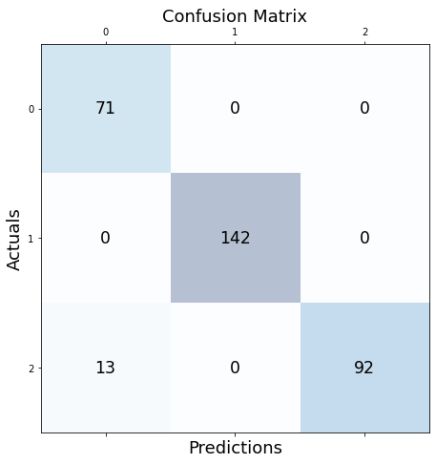
Lalu untuk tree berikutnya menggunakan 70/30 dengan tree berikut



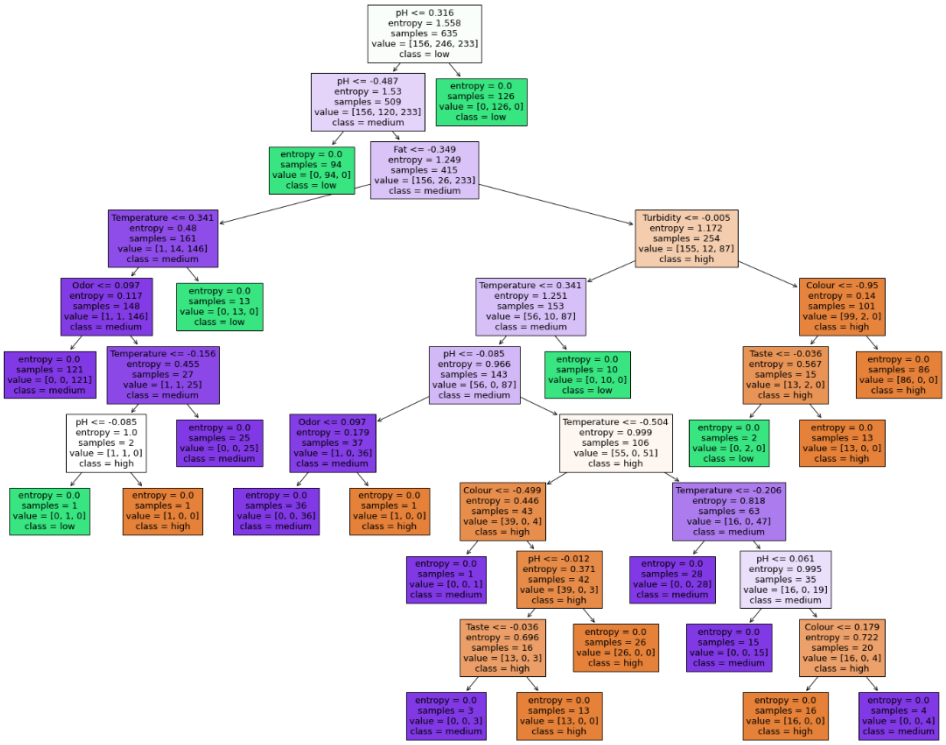
```
[26]
    akurasi = accuracy_score(y_test, y_pred)
    print("Akurasi = ",akurasi*100)

...  Akurasi =  95.9119496855346
```

```
[27]
    print(classification_report(y_test, y_pred))
    cm = confusion_matrix(y_test, y_pred)
```

	precision	recall	f1-score	support
high	0.85	1.00	0.92	71
low	1.00	1.00	1.00	142
medium	1.00	0.88	0.93	105
accuracy			0.96	318
macro avg	0.95	0.96	0.95	318
weighted avg	0.97	0.96	0.96	318



Sekarang tree yang dibuat dengan data 60/40

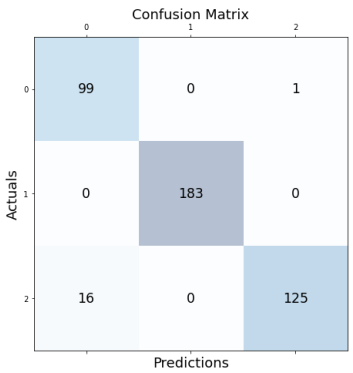



```
[42]
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
```

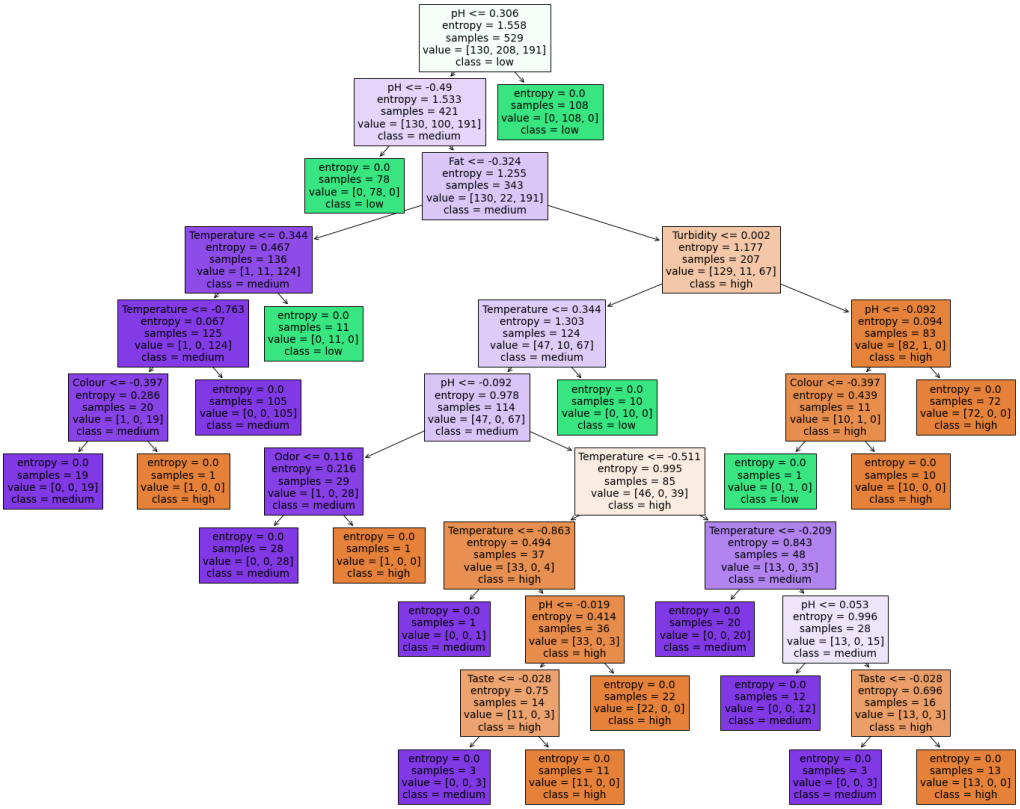
```
              precision    recall  f1-score   support

        high       0.86      0.99      0.92       100
         low       1.00      1.00      1.00       183
      medium       0.99      0.89      0.94       141

    accuracy                           0.96       424
   macro avg       0.95      0.96      0.95       424
weighted avg       0.96      0.96      0.96       424
```

```
[43]
akurasi = accuracy_score(y_test, y_pred)
print("Akurasi = ",akurasi*100)
```

```
Akurasi =  95.99056603773585
```



Lalu yang terakhir 50/50



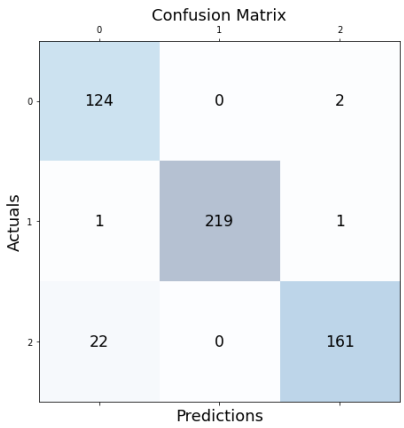

```
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
```

	precision	recall	f1-score	support
high	0.84	0.98	0.91	126
low	1.00	0.99	1.00	221
medium	0.98	0.88	0.93	183
accuracy			0.95	530
macro avg	0.94	0.95	0.94	530
weighted avg	0.96	0.95	0.95	530

```
akurasi = accuracy_score(y_test, y_pred)
print("Akurasi = ",akurasi*100)
```

[36]

```
Akurasi =  95.09433962264151
```

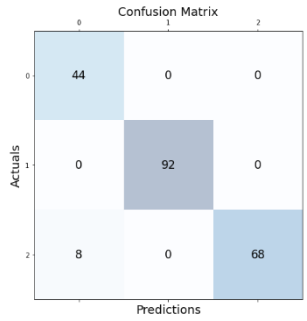
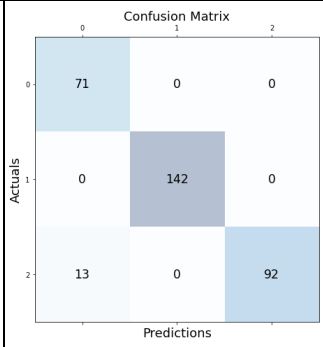
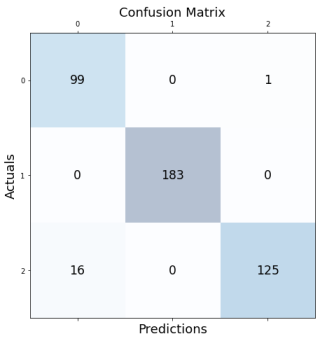
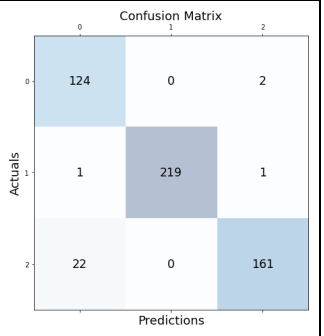


Tree yang dihasilkan 4 percobaan diatas berbentuk hampir sama dengan perbedaan variabel. Hasil kualitas model di masukan ke tabel dibawah

Spliting Data	Jumlah Data Training	Jumlah Data Testing	AKURASI
80/20	847	212	96,22%
70/30	741	318	95,91%
60/40	635	424	95,99%
50/50	529	530	95,09%

Terjadi sedikit penurunan tingkat akurasi

Dan dibawah ini adalah confusion matrix dari masing masing percobaan diatas

80/20	70/30	60/40	50/50
			

Catatan: kolom 0=High,1=Low,3=Medium

Dari hasil pengujian diatas.dapat disimpulkan bahwa model tree c4.5 yang dibuat tergolong akurat dengan akurasi diatas 95%. Menurunkan data training dan meningkatkan data testing sedikit menurunkan tingkat akurasi meskipun tidak signifikan, hal ini bisa dilihat dari confusion matrix masing-masing tree. Meningkatkan data testing membuat model lebih sering membuat kesalahan dalam klasifikasi meskipun masih dalam tingkat kewajaran.

Data Low(kolom 1) memiliki tingkat kesalahan paling kecil diantara data high dan medium, ini disebabkan oleh data low yang mmemiliki jumlah terbanyak dalam dataset milkquality ini.

PENUTUP

Dari hasil implementasi dan analisa , maka dapat disimpulkan bahwa:

1. Algoritma Decision Tree C4.5 dapat digunakan untuk melakukan klasifikasi kualitas susu. Dari hasil pengujian 4x dengan jumlah data yang berbeda menunjukkan bahwa algoritma memberikan tingkat akurasi yang baik (**Semuanya diatas 95%)** serta kesalahan dalam tingkat minimal ketika kita lihat pada confusion matrix 4 implementasi diatas
2. Tingkat akurasi rata-rata ke 4 percobaan diatas adalah **95,80%.**Akurasi pengujian terbaik dari implementasi algoritma C4.5 mencapai angka 96,22% untuk pengujian

dengan 80% data training(847 data) dan 20% data testing(212 data). Akurasi terendah daalam implementasi 95,09% untuk pengujian dengan 50% data training(529 data) dan 50% data testing(530 data).Dari hasil uji coba, menunjukkan bahwa **semakin tinggi data training semakin tinggi akurasi.**

3. Ada berbagai macam faktor yang mempengaruhi ujicoba algoritma C4.5 untuk klasifikasi kualitas susu. Faktor-faktor tersebut didasarkan pada uji coba implementasi yaitu faktor fitur yang digunakan, faktor keutuhan dan bentuk data, dan faktor pemecahan data.

   Dari Faktor fitur yang digunakan yaitu :

   - pH
   - Temperature
   - Taste
   - Odor.
   - Fat
   - Turbidity
   - Color

   Semuanya berpengaruh terhadap kualitas grade susu yang dihasilkan dibuktikan dengan hasil tree

   Untuk Faktor keutuhan data dari dataset yang digunakan data kosong dan outliers dapat menurunkan akurasi dari model, serta data training dan testing yang digunakan untuk mengetes model hasilnya akan jauh berbeda bila data bagian ni tidak di normalisasi yang dapat mengakibatkan overfitting dan ketidak akurasian pada model. Pada data yang digunakan **tidak terdapat data kosong** sehingga membantu dalam tingkat akurasi model tree. **Data training dan testing juga dinormalisasi** agar dapat data memiliki rentang yang lebih baik untuk masuk dalam algoritma tree.

   Faktor pemecahan data untuk training dan testing juga memiliki impact besar dalam kualitas model dari data diatas, sweet spot dengan tingkat akurasi tinggi untuk jumlah data yang digunakan untuk algoritma adalah diantara **20-30%** dari data untuk testing menjadikan **80-70%** data sisa sebagai data training.