Membuat Prediksi Kelulusan Pelajar Menggunakan Decision Tree
Mata Kuliah Kecerdasan Buatan

[1] Benedicto Reinaldy Pramananditya, [2] Emanuel Amstrong Hayong
[1] benedicto.reinaldy@student.ukdc.ac.id
[2] emanuel.hayong@student.ukdc.ac.id

ABSTRAK

Decision tree adalah sebuah metode klasifikasi yang cukup popular dan mudah dipahami dan juga tidak terlalu sulit untuk diinterpretasi oleh manusia. *Decision tree* adalah sebuah model prediksi yang menggunakan analogi struktur pohon atau struktur berhirarki. Pada laporan yang kami buat kali ini menjelaskan mengenai bagaimana membuat sebuah prediksi lulus atau tidaknya seorang pelajar menggunakan data performa dari pelajar tersebut menggunakan metode *Decision Tree*.

Kata Kunci: *Decision Tree*, prediksi, pohon keputusan

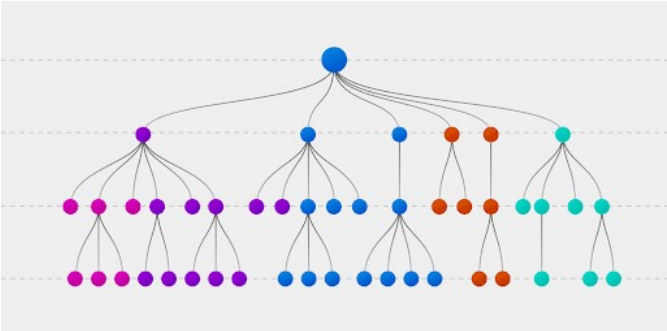
ABSTRACT

Decision tree is a classification method that quite popular dan easy to learn for human. Decision tree is a prediction model that use tree structure analogy or hierarchical structure. In this report, we will be explaining how to make a pass or fail prediction for student using their past performance data with Decision Tree method.

Keywords: Decision Tree, prediction

PENDAHULUAN

Decision tree adalah sebuah metode klasifikasi yang cukup popular dan mudah dipahami dan juga tidak terlalu sulit untuk diinterpretasi oleh manusia. *Decision tree* adalah sebuah model prediksi yang menggunakan analogi struktur pohon atau struktur berhirarki. Konsep dari *decision tree* adalah mengubah data menjadi *decision tree* dan aturan-aturan keputusan lalu dari aturan-aturan keputusan tersebut diambil sebuah pengambil keputusan yang akan menginterpretasikan solusi dari permasalahan [1]. Metode ini mengeliminasi perhitungan atau data-data yang kiranya tidak diperlukan karena sampel yang ada biasanya hanya diuji berdasarkan kriteria atau kelas tertentu saja. Penggunaan *decision tree* umumnya dalam riset operasi khususnya dalam analisis keputusan. Tujuan dari *decision tree* adalah untuk membantu mengidentifikasi strategi atau keputusan yang memungkinkan dalam sebuah pengambilan keputusan. *Decision tree* dapat dilinerisasi menjadi sebuah aturan keputusan dimana hasilnya adalah isi dari simpul daun dan kondisi di sepanjang jalur membentuk konjungsi dalam klausa if.



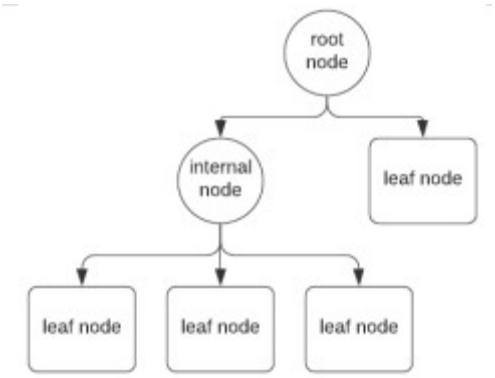
Gambar 1 Decision Tree

Pada laporan yang kami buat, kami akan membuat sebuah model prediksi lulus atau tidaknya seorang siswa berdasarkan data performa siswa tersebut menggunakan metode *decision tree*. Laporan yang kami buat akan berdasarkan coding yang kami kerjakan di google collab dan bisa diakses pada link berikut: https://bit.ly/source-code-decision-tree dan untuk laporan PDF dapat diakses pada link berikut: https://bit.ly/decision-tree-project-book. Untuk dataset mentah yang kami gunakan, kami akses dari link berikut: https://archive.ics.uci.edu/ml/datasets/student+performance. File yang digunakan Bernama 'student-por.csv'. Penjelasan mengenai hal tersebut ada di google collab.

TINJAUAN PUSTAKA

Decision tree adalah salah satu teknik atau metode klasifikasi yang sering digunakan dan cukup popular dan juga mudah dipahami oleh manusia. *Decision tree* adalah struktur *flowchart* yang menyerupai *tree* (pohon) dimana sebuah *node* adalah pengujian terhadap variabel atribut, tiap cabangnya adalah hasil dari pengujian tersebut sedangkan *node* terluarnya adalah *leaf* yang menjadi labelnya. Metode ini sering digunakan dan diterapkan sebagai solusi untuk mengklasifikasikan sebuah masalah dan juga dapat memprediksi pola dari data dan menggambarkan relasi dari *variable* antar atribut dalam bentuk pohon. Pembuatan *decision tree* ini menggunakan metode *supervised learning*. *Decision tree* terdiri dari *root*, *node*, dan *leaf* [2].

Konsep dari *decision tree* adalah setiap *node* menyatakan pengujian terhadap suatu atribut, setiap cabang menyatakan output dari pengujian tersebut dan *leaf node* (daun) menyatakan kelas-kelas atau distribusi kelas. *Node* teratas disebut dengan *root node* atau *node* akar dengan kata lain *node* tersebut tidak memiliki input namun mengeluarkan beberapa output. Sedangkan *leaf node* (daun) memiliki satu *input* tanpa memiliki *output*. *Leaf node* (daun) adalah hasil akhir hasil akhir yang mewakili label kelas dari kombinasi atribut yang terbentuk menjadi rule.

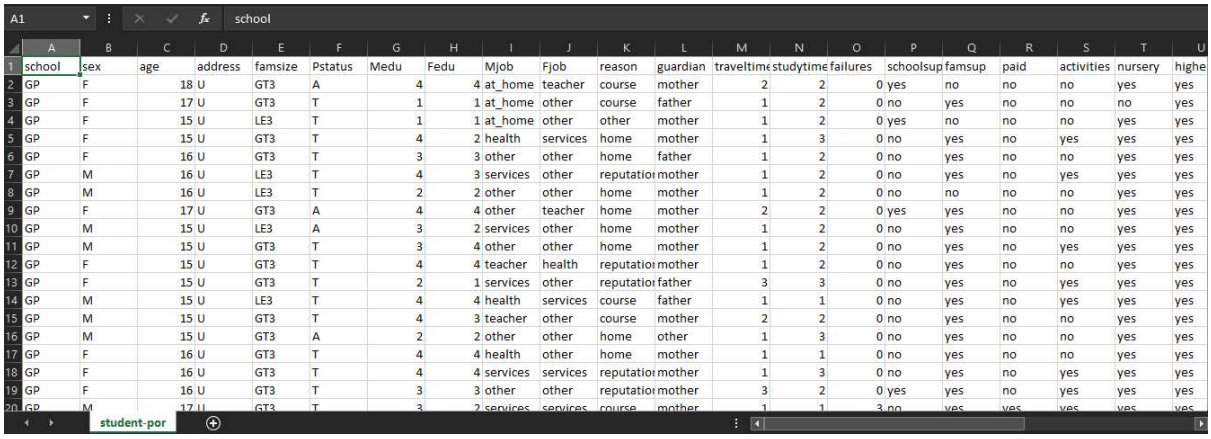


Gambar 2 Decision Tree

Dalam membangun sebuah *decision tree*, tahap awal dari pembuatannya adalah melakukan sebuah evaluasi terhadap semua atribut yang ada untuk mengukur efektifitas suatu atribut dalam mengklasifikasikan suatu kumpulan data. Atribut yang diletakkan pada *root node* (*node* teratas) adalah atribut dengan information *gain* terbesar [3].

HASIL DAN PEMBAHASAN

Sebelum memulai penjelasan *coding* dan pembahasannya, pembaca diminta untuk mendownload *dataset* yang akan digunakan dalam *project* ini pada link berikut: https://archive.ics.uci.edu/ml/datasets/student+performance. *File* yang akan digunakan bernama 'student-por.csv'. Berikut juga dilampirkan link google *collab* jika pembaca ingin mencoba *run code*nya: https://bit.ly/source-code-decision-tree. Tujuan dari *project* ini adalah kami memprediksi lulus atau tidaknya siswa dari data performa siswa menggunakan metode *decision tree*. Lulus atau tidaknya siswa direpresentasikan menggunakan *boolean* atau *biner* yaitu jika 1 maka siswa lulus dan jika 0 maka siswa tidak lulus. *Dataset* yang akan kita gunakan mempunyai total 649 data siswa dengan masing-masing siswa memiliki 30 kolom atau atribut.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	reason	guardian	traveltime	studytime	failures	schoolsup	famsup	paid	activities	nursery	highe
2	GP	F	18	U	GT3	A	4	4	at_home	teacher	course	mother	2	2	0	yes	no	no	no	yes	yes
3	GP	F	17	U	GT3	T	1	1	at_home	other	course	father	1	2	0	no	yes	no	no	no	yes
4	GP	F	15	U	LE3	T	1	1	at_home	other	other	mother	1	2	0	yes	no	no	no	yes	yes
5	GP	F	15	U	GT3	T	4	2	health	services	home	mother	1	3	0	no	yes	no	yes	yes	yes
6	GP	F	16	U	GT3	T	3	3	other	other	home	father	1	2	0	no	yes	no	no	yes	yes
7	GP	M	16	U	LE3	T	4	3	services	other	reputatio	mother	1	2	0	no	yes	no	yes	yes	yes
8	GP	M	16	U	LE3	T	2	2	other	other	home	mother	1	2	0	no	no	no	no	yes	yes
9	GP	F	17	U	GT3	A	4	4	other	teacher	home	mother	2	2	0	yes	yes	no	no	yes	yes
10	GP	M	15	U	LE3	A	3	2	services	other	home	mother	1	2	0	no	yes	no	no	yes	yes
11	GP	M	15	U	GT3	T	3	4	other	other	home	mother	1	2	0	no	yes	no	yes	yes	yes
12	GP	F	15	U	GT3	T	4	4	teacher	health	reputatio	mother	1	2	0	no	yes	no	no	yes	yes
13	GP	F	15	U	GT3	T	2	1	services	other	reputatio	father	3	3	0	no	yes	no	yes	yes	yes
14	GP	M	15	U	LE3	T	4	4	health	services	course	father	1	1	0	no	yes	no	yes	yes	yes
15	GP	M	15	U	GT3	T	4	3	teacher	other	course	mother	2	2	0	no	yes	no	no	yes	yes
16	GP	M	15	U	GT3	A	2	2	other	other	home	other	1	3	0	no	yes	no	no	yes	yes
17	GP	F	16	U	GT3	T	4	4	health	other	home	mother	1	1	0	no	yes	no	no	yes	yes
18	GP	F	16	U	GT3	T	4	4	services	services	reputatio	mother	1	3	0	no	yes	no	yes	yes	yes
19	GP	F	16	U	GT3	T	3	3	other	other	reputatio	mother	3	2	0	yes	yes	no	yes	yes	yes
20	GP	M	17	U	GT3	T	3	2	services	services	course	mother	1	1	3	no	yes	yes	yes	yes	yes

Gambar 3 student-por.csv

Textbox dibawah ini adalah penjelasan mengenai atribut atau kolom pada *dataset*,

Attributes for both student-mat.csv (Math course) and student-por.csv (Portuguese language course) datasets:

1 school - student's school (binary: "GP" - Gabriel Pereira or "MS" - Mousinho da Silveira)

2 sex - student's sex (binary: "F" - female or "M" - male)

3 age - student's age (numeric: from 15 to 22)

4 address - student's home address type (binary: "U" - urban or "R" - rural)

5 famsize - family size (binary: "LE3" - less or equal to 3 or "GT3" - greater than 3)

6 Pstatus - parent's cohabitation status (binary: "T" - living together or "A" - apart)

7 Medu - mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)

8 Fedu - father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)

9 Mjob - mother's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")

10 Fjob - father's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")

11 reason - reason to choose this school (nominal: close to "home", school "reputation", "course" preference or "other")

12 guardian - student's guardian (nominal: "mother", "father" or "other")

13 traveltime - home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)

14 studytime - weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)

15 failures - number of past class failures (numeric: n if 1<=n<3, else 4)

16 schoolsup - extra educational support (binary: yes or no)

17 famsup - family educational support (binary: yes or no)

18 paid - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)

19 activities - extra-curricular activities (binary: yes or no)

20 nursery - attended nursery school (binary: yes or no)

21 higher - wants to take higher education (binary: yes or no)

22 internet - Internet access at home (binary: yes or no)

23 romantic - with a romantic relationship (binary: yes or no)

24 famrel - quality of family relationships (numeric: from 1 - very bad to 5 - excellent)

25 freetime - free time after school (numeric: from 1 - very low to 5 - very high)

26 goout - going out with friends (numeric: from 1 - very low to 5 - very high)

27 Dalc - workday alcohol consumption (numeric: from 1 - very low to 5 - very high)

28 Walc - weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)

29 health - current health status (numeric: from 1 - very bad to 5 - very good)

30 absences - number of school absences (numeric: from 0 to 93)

these grades are related with the course subject, Math or Portuguese:

31 G1 - first period grade (numeric: from 0 to 20)

31 G2 - second period grade (numeric: from 0 to 20)

32 G3 - final grade (numeric: from 0 to 20, output target)

Additional note: there are several (382) students that belong to both datasets .

These students can be identified by searching for identical attributes

that characterize each student, as shown in the annexed R file.

Langkah-langkah,

1. *Import dataset* yang akan digunakan dari *device* kita. *Dataset* yang digunakan bernama 'student-por.csv'.

```
# import / load dataset yang akan digunakan dengan nama file student
-por.csv

import pandas as pd
from google.colab import files
uploaded = files.upload()
d = pd.read_csv('student-por.csv', sep=';')

Output,
```


```
Saving student-por.csv to student-por.csv
```

2. Kita cek *dataset* kita dengan menjalankan *coding* `len(d)`, dimana hasil *run*nya adalah 649 yang berarti dalam *dataset* kita terdapat total 649 data siswa.

```
# dari dataset yang di import berisikan total 649 data siswa
len(d)

Output,
649
```

3. Menambahkan kolom baru yaitu ‘pass’ dimana kolom ini mempunyai parameter 0 (tidak lulus atau 1 (lulus). Perhitungan dari kolom ‘pass’ ini adalah berdasarkan perhitungan penjumlahan nilai G1+G2+G3, apabila nilai dari penjumlahan tersebut adalah lebih dari sama dengan 35 maka siswa tersebut dianggap lulus begitu juga sebaliknya.

```
# menambahkan kolom baru yaitu 'pass' berdasarkan perhitungan penjum
lahan G1-G3
d['pass'] = d.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])
 >= 35 else 0, axis=1)
d = d.drop(['G1', 'G2', 'G3'], axis=1)
d.head() # menampilkan 5 baris pertama dari dataset yang digunakan

Output,
```

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	reason	guardian	traveltime	studytime	f
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	course	mother	2	2	
1	GP	F	17	U	GT3	T	1	1	at_home	other	course	father	1	2	
2	GP	F	15	U	LE3	T	1	1	at_home	other	other	mother	1	2	
3	GP	F	15	U	GT3	T	4	2	health	services	home	mother	1	3	
4	GP	F	16	U	GT3	T	3	3	other	other	home	father	1	2	

4. Selanjutnya kami menggunakan *one-hot encoding* untuk kolom yang bersifat kategorikal (kolom yang menggunakan tipe *biner*/*boolean*), bisa dilihat pada *notepad* txt yang di *download*. Lalu isi dari kolom tersebut dikonversikan ke dalam *biner*. *One-hot encoding* adalah salah satu metode *encoding* yang mana metode ini mempresentasikan data bertipe kategori sebagai *vector biner* yang bernilai 0 dan 1 dimana semua elemen akan bernilai 0 kecuali satu elemen bernilai 1 yaitu elemen yang memiliki kategori tersebut [4].

```
# menggunakan one-
hot encoding untuk kolom kategorikal (kolom yang menggunakan biner/b
ool)
# konversikan ke biner
d = pd.get_dummies(d, columns = ['sex', 'school', 'address', 'famsiz
e', 'Pstatus', 'Mjob', 'Fjob',
                                 'reason', 'guardian', 'schoolsup',
'famsup', 'paid', 'activities',
                                 'nursery', 'higher', 'internet', 'r
omantic'])
d.head()
```

Output,

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	Walc	health	absences	pass	sex_F	sex_M	school_GP	s
0	18	4	4	2	2	0	4	3	4	1	1	3	4	0	1	0	1	
1	17	1	1	1	2	0	5	3	3	1	1	3	2	0	1	0	1	
2	15	1	1	1	2	0	4	3	2	2	3	3	6	1	1	0	1	
3	15	4	2	1	3	0	3	2	2	1	1	5	0	1	1	0	1	
4	16	3	3	1	2	0	4	3	2	1	2	5	0	1	1	0	1	

5. Disini kami memisahkan data untuk *training* dan data untuk *test*. Total dalam *dataset* kita terdapat 649 data siswa. Kami ambil contoh 500 data siswa untuk data *training* dan sisanya (149 data siswa) untuk data test. Lalu kami hitung banyaknya siswa yang tergolong pasif dalam *dataset* kita.

```
# acak kolomnya
d = d.sample(frac=1)
# memisahkan data untuk training dan data untuk test
d_train = d[:500]
d_test = d[500:]

d_train_att = d_train.drop(['pass'], axis=1)
d_train_pass = d_train['pass']

d_test_att = d_test.drop(['pass'], axis=1)
d_test_pass = d_test['pass']

d_att = d.drop(['pass'], axis=1)
d_pass = d['pass']

# dihitung banyaknya siswa pasif dalam dataset kita
import numpy as np
print("Passing %d out of %d (%.2f%%" % (np.sum(d_pass), len(d_pass),
 100*float(np.sum(d_pass)) / len(d_pass)))


Output,

Passing 328 out of 649 (50.54%
```

6. Membuat sebuah model decision tree dengan node maksimal adalah 5.

```
# memasukkan ke model ke dalam decision tree
from sklearn import tree
t = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
t = t.fit(d_train_att, d_train_pass)
```

```
# simpan modelnya
tree.export_graphviz(t, out_file="student-
performance.dot", label="all", impurity=False, proportion=True,
                     feature_names=list(d_train_att), class_names=["
fall", "pass"],
                     filled=True, rounded=True)
```

```
t.score(d_test_att, d_test_pass)
```

Output,

```
0.7181208053691275
```

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(t, d_att, d_pass, cv=5)
# menampilkan rata-rata skor siswa dan dua standar deviasinya
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() *
 2))
```

Output,

```
Accuracy: 0.70 (+/- 0.08)
```

7. Melakukan *cross validation* pada seluruh dataset yang akan membagi data pada salah satu dari 20/80, di mana 20% adalah *testing set* dan 80% pada *training set*. Rata-rata hasilnya 67%. Ini menunjukkan bahwa kami memiliki kumpulan data yang seimbang. Di sini kami memiliki berbagai pilihan yang harus diambil terkait max_depth:

```
for max_depth in range(1, 20):
  t = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max
_depth)
  scores = cross_val_score(t, d_att, d_pass, cv=5)
  print ("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth,
scores.mean(), scores.std() * 2))
```

Output,

```
Max depth: 1, Accuracy: 0.64 (+/- 0.03)
Max depth: 2, Accuracy: 0.69 (+/- 0.04)
Max depth: 3, Accuracy: 0.68 (+/- 0.04)
Max depth: 4, Accuracy: 0.69 (+/- 0.04)
Max depth: 5, Accuracy: 0.71 (+/- 0.08)
Max depth: 6, Accuracy: 0.68 (+/- 0.06)
Max depth: 7, Accuracy: 0.68 (+/- 0.03)
Max depth: 8, Accuracy: 0.69 (+/- 0.02)
Max depth: 9, Accuracy: 0.68 (+/- 0.06)
Max depth: 10, Accuracy: 0.67 (+/- 0.06)
Max depth: 11, Accuracy: 0.68 (+/- 0.06)
Max depth: 12, Accuracy: 0.69 (+/- 0.06)
Max depth: 13, Accuracy: 0.66 (+/- 0.04)
Max depth: 14, Accuracy: 0.66 (+/- 0.05)
Max depth: 15, Accuracy: 0.65 (+/- 0.06)
Max depth: 16, Accuracy: 0.65 (+/- 0.05)
Max depth: 17, Accuracy: 0.64 (+/- 0.06)
Max depth: 18, Accuracy: 0.65 (+/- 0.03)
Max depth: 19, Accuracy: 0.65 (+/- 0.04)
```

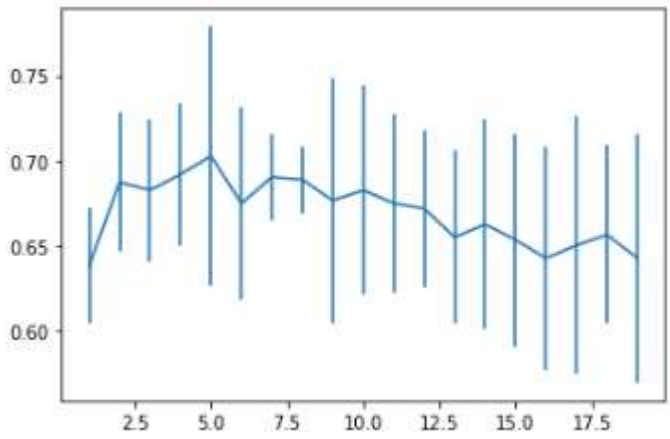
```
depth_acc = np.empty((19,3), float)
i = 0
for max_depth in range(1,20):
  t = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max
_depth)
  scores = cross_val_score(t, d_att, d_pass, cv=5)
  depth_acc[i,0] = max_depth
  depth_acc[i,1] = scores.mean()
  depth_acc[i,2] = scores.std() * 2
  i += 1

depth_acc
```

Output,

```
array([[ 1.        ,  0.63792487,  0.0341954 ],
       [ 2.        ,  0.68725104,  0.0408204 ],
       [ 3.        ,  0.68262373,  0.04184171],
       [ 4.        ,  0.69187835,  0.04231734],
       [ 5.        ,  0.70271914,  0.07638063],
       [ 6.        ,  0.67484794,  0.0570895 ],
       [ 7.        ,  0.69030411,  0.02587868],
       [ 8.        ,  0.68876565,  0.0199723 ],
       [ 9.        ,  0.67648181,  0.07255199],
       [10.        ,  0.68261181,  0.0613624 ],
       [11.        ,  0.67482409,  0.05258023],
       [12.        ,  0.67174717,  0.04590311],
       [13.        ,  0.65482409,  0.05079687],
       [14.        ,  0.66255218,  0.06116406],
       [15.        ,  0.6532737 ,  0.06243323],
       [16.        ,  0.64248062,  0.0660324 ],
       [17.        ,  0.65017293,  0.07564666],
       [18.        ,  0.65637448,  0.05253353],
       [19.        ,  0.64249255,  0.07323162]])
```

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
plt.show()
```

Output,



KESIMPULAN

Decision tree adalah struktur *flowchart* yang menyerupai *tree* (pohon) dimana sebuah *node* adalah pengujian terhadap variabel atribut. Konsep dari *decision tree* adalah setiap *node* menyatakan pengujian terhadap suatu atribut, setiap cabang menyatakan output dari pengujian tersebut dan *leaf node* (daun) menyatakan kelas-kelas atau distribusi kelas. Dalam membangun sebuah *decision tree*, tahap awal dari pembuatannya adalah melakukan sebuah evaluasi terhadap semua atribut yang ada untuk mengukur efektifitas suatu atribut dalam mengklasifikasikan suatu kumpulan data.

Hasil yang kami dapat setelah melakukan proses percobaan source code adalah penentuan lulus tidaknya siswa dari banyaknya indikator yang berjumlah 20. Proses penentuan lulus atau tidaknya mahasiswa dapat dilakukan dengan menghitung persentase jumlah lulus dan gagal yang akan menghasilkan 328 siswa yang lulus dari total 649 siswa. Ini adalah persentase kelulusan yang kira-kira sekitar 50% dari dataset.

DAFTAR PUSTAKA

[1] IYKRA, “Mengenal Decision Tree dan Manfaatnya,” *23 Juli*, 2018. https://medium.com/iykra/mengenal-decision-tree-dan-manfaatnya-b98cf3cf6a8d.

[2] Y. A. Wijaya *et al.*, “Analisa Klasifikasi menggunakan Algoritma Decision Tree pada Data Log Firewall Jurnal Sistem Informasi dan Manajemen,” *Anal. Klasifikasi menggunakan Algoritm. Decis. Tree pada Data Log Firewall*, vol. 9, no. 3, 2021, [Online]. Available: https://ejournal.stmikgici.ac.id/index.php/jursima/article/view/303.

[3] P. Kasih, “Pemodelan Data Mining Decision Tree Dengan Classification Error Untuk Seleksi Calon Anggota Tim Paduan Suara,” *Innov. Res. Informatics*, vol. 1, no. 2, pp. 63–69, 2019, doi: 10.37058/innovatics.v1i2.918.

[4] L. Afifah, “2 Cara Implementasi One-Hot Encoding di Python.” https://ilmudatapy.com/one-hot-encoding-di-python/.