Pengenalan Angka Tulis Tangan dengan Mengaplikasikan Tree

Muhammad Bangkit Dwi Cahyono - 13521055[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13521055@std.stei.itb.ac.id

***Abstract*—Pengenalan tulisan tangan dengan menggunakan gambar dapat diaplikasikan dalam berbagai hal yang berkaitan dengan kehidupan sehari-hari. Salah satu metode yang digunakan adalah dengan menggunakan *decision making models.* Penggunaan *Decision Tree* merupakan salah satu metode untuk menghasilkan klasifikasi dengan akurasi yang tinggi dan eksekusi yang cepat dengan struktur data pohon yang merepresentasikan informasi yang dapat dimanfaatkan. Namun, *Decision Tree* saja tidak cukup untuk pengambilan keputusan pada kasus ini, disebabkan data dengan varians yang sangat tinggi. Oleh karena itu, untuk meningkatkan akurasi dalam pengklasifikasian ini, digunakan *Random Forest Algorithm*. Dengan demikian, pengambilan keputusan dengan adanya kemungkinan kesalahan dapat diminimalisir. Validitas dari metode tersebut akan dibahas pada *paper* ini.**

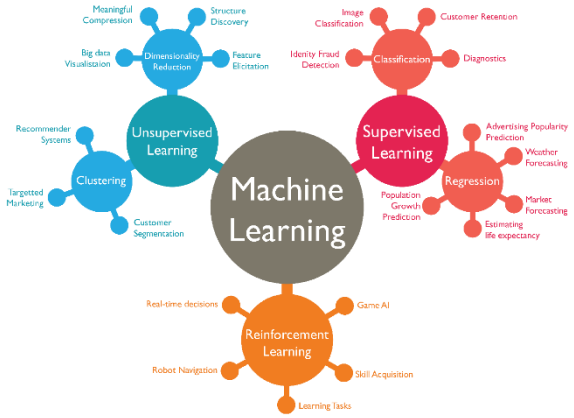
***Keywords*—Random Forest, Decision Tree, Machine Learning, pembelajaran mesin terhadap data (*Data Training*).**

I. PENDAHULUAN

Dalam pembelajaran data, pengambilan keputusan akhir yang optimal adalah kunci dari keberhasilan. Pengambilan ini didasarkan atas kombinasi pertimbangan akan peristiwa saat percobaan dilakukan, baik percobaan yang berhasil maupun gagal. Kebutuhan akan adanya sistem yang dapat melakukan pengambilan keputusan yang baik semakin meningkat. Data dengan jumlah yang banyak bukan lagi sebuah tantangan pada era digital sekarang ini.

Pengambilan keputusan memegang peranan penting pada kehidupan sehari-hari. Sistem pendukung pengambilan keputusan yang dapat membantu manusia menjadi suatu kebutuhan yang semakin meningkat, terutama pada situasi dimana pengambilan keputusan harus dilakukan dengan cepat, namun tetap akurat. Efisiensi dalam suatu sistem dibutuhkan, tentunya dengan *output* yang juga berkualitas tinggi.

Sistem pengambilan keputusan berbasis pembelajaran mesin (*Machine Learning*) menjadi salah satu opsi yang sedang viral akhir ini. Dengan data yang besar, tentunya bidang *Machine Learning* ini dapat mengubah pandangan manusia terhadap teknologi dan mengubah cara hidup manusia, tentunya ke arah yang lebih baik. Mulai dari pembelajaran mesin yang dapat memprediksi cuaca hingga mesin yang dapat menjawab semua pertanyaan manusia yang dilontarkan kepadanya. *Machine Learning* dan *Artificial Intelligence* merupakan elemen yang tidak dapat terlepaskan berkaitan dengan transformasi digital saat ini.



Gambar 1.1 Penerapan Machine Learning di kehidupan sehari-hari
Sumber: https://www.wordstream.com/

Salah satu pengaplikasiannya adalah dalam *Image Classification* yang akan diberikan pemaparan tentang pengaplikasiannya dalam *paper* ini. Algoritma yang digunakan adalah *Random Forest Algorithm* dengan basis pengaplikasian *Tree* dalam *Decision Tree*. Algoritma ini memanfaatkan konsep penggabungan hasil keputusan dari 2 atau lebih *Decision Tree* yang berbeda (*Aggregation).* Oleh karena itu, *class* atau klasifikasi hasil dari algoritma ini akan *less sensitive* terhadap data original.

II. DASAR TEORI

A. Graf (*Graph*)

Graf dapat diartikan sebagai suatu data struktur yang digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut.

Graf dapat dinyatakan sebagai sebuah pasangan himpunan (*V, E*) dengan *V* adalah himpunan tidak kosong dari simpul-simpul (*vertices*) atau dapat dinyatakan sebagai $\{ v_1, v_2, \dots , v_n \}$ dan *E* adalah himpunan sisi (*edges*) yang menghubungkan sepasang simpul atau dapat dinyatakan sebagai $\{ e_1, e_2, \dots , e_n \}$. Penamaan simpul dan sisi pada graf dapat dinyatakan dengan angka, huruf, atau simbol matematis lainnya.

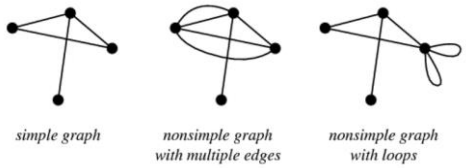
Graf memiliki banyak jenis, dapat dikelompokkan dari aspek jumlah gelang, sisi, simpul, dan kombinasi struktur lainnya.

Berdasarkan keberadaan sisi atau gelang, graf dapat dikelompokkan sebagai berikut.

1. Graf sederhana (*simple graph*)

Graf yang tidak mengandung gelang maupun sisi ganda.

2. Graf tak-sederhana (*unsimple-graph*)
   Graf yang mengandung sisi ganda atau gelang dalam strukturnya.
   2. 1. Graf ganda
      Graf yang mengandung sisi ganda.
   2. 2. Graf semu
      Graf yang mengandung sisi berupa gelang.



Gambar 2.1 Jenis graf berdasarkan keberadaan sisi atau gelang
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

Berdasarkan orientasi arah pada sisi, graf dapat dikelompokkan sebagai berikut.

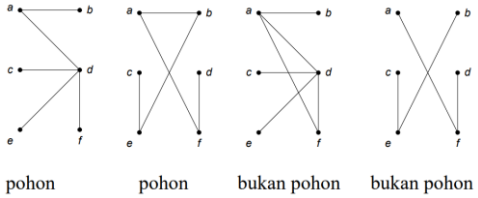
1. Graf tak-berarah (*undirected graph*)
   Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah.
2. Graf berarah (*directed graph*)
   Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah.

Graf merupakan struktur data yang mengandung berbagai terminologi. Terminologi yang berkaitan dengan graf sebagai berikut.

1. Ketetanggaan (*adjacency*)
   Dua buah simpul dapat dikatakan bertetangga jika terhubung langsung.
2. Bersisian (*incidency*)
   Untuk sembarang sisi $e = (v_j, v_k)$, maka e bersisian dengan simpul v_j dan v_k.
3. Simpul terpencil (*isolated vertex)*
   Simpul yang tidak mempunyai sisi yang bersisian dengan simpul itu sendiri.
4. Graf kosong (*null graph*)
   Graf yang himpunan sisinya adalah himpunan kosong.
5. Derajat (*degree*)
   Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul itu sendiri.
6. Lintasan (*path*)
   Barisan berselang-seling simpul dan sisi yang dilalui dari suatu simpul ke simpul lainnya.
7. Siklus (*cycle*) atau Sirkuit (*circuit*)
   Lintasan yang berawal dan berakhir pada simpul yang sama.
8. Keterhubungan (*connectedness*)
   Dua simpul dikatakan terhubung jika terdapat lintasan dari simpul satu ke yang lainnya.

B. Pohon (*Tree*)

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit. Pohon adalah suatu struktur data yang mengandung set suatu elemen dan menyimpan representasi informasi hubungan di antara elemen-elemennya.

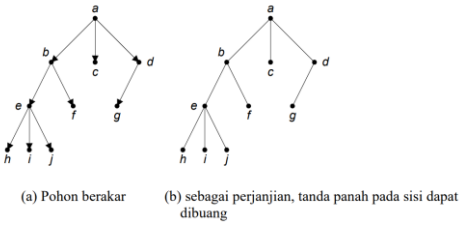


Gambar 2.2 Pohon dalam bentuk Graf
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

Properti pohon dapat dijelaskan sebagai berikut. Misalkan $G = (V, E)$ adalah graf tak-berarah sederhana dan jumlah simpulnya sebanyak n. Pernyataan di bawah ini adalah benar.

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

Aplikasi pohon pada umumnya menggunakan jenis pohon berakar (*rooted tree*). Bentuk pohon berakar memiliki karakter unik menyerupai pohon nyata terbalik, dengan akar berada pada bagian teratas dan daun berada pada bagian paling bawah suatu akar.



Gambar 2.3 Pohon berakar
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

Pohon berakar merupakan struktur data yang mengandung berbagai terminologi. Terminologi yang berkaitan dengan pohon berakar sebagai berikut.

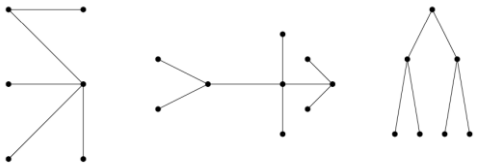
1. Anak (*child*)
   Simpul dapat dikatakan sebagai anak jika terdapat sisi dari simpul tersebut ke simpul lain yang ada di atasnya.
2. Lintasan (path)
   Lintasan adalah runtunan simpul-simpul yang dilalui dari suatu simpul ke simpul lainnya. Setiap simpul yang dilalui merupakan *parent node* dari simpul berikutnya.
3. Saudara kandung (*sibling)*
   Anak dari *parent node* yang sama, berada pada *level* yang sama.
4. Upapohon (*subtree*)
   Upagraf dari pohon berakar yang mengandung simpul dan semua keturunan beserta semua sisi dalam lintasan yang berasal dari simpul berkaitan.
5. Derajat (*degree*)

Jumlah upapohon (atau jumlah anak) yang dimiliki oleh suatu simpul.

6. Daun (*leaf*)
   Simpul berderajat nol (tidak mempunyai anak).
7. Simpul dalam (*internal nodes*)
   Simpul yang mempunyai anak.
8. Aras (*level*) atau Tingkat
   Kedalaman suatu simpul relatif terhadap akar, akar memiliki tingkat 0.
9. Tinggi (*height*) atau Kedalaman (*depth*)
   Level maksimum dari suatu pohon.

C. Hutan (*Forest*)

Hutan adalah kumpulan pohon yang saling lepas atau graf tidak terhubung yang tidak mengandung sirkuit. Setiap komponen di dalam graf terhubung tersebut adalah pohon.

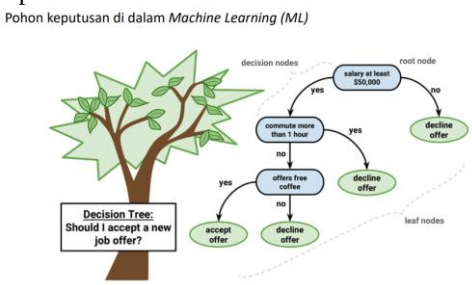


Hutan yang terdiri dari tiga buah pohon

Gambar 2.4 Hutan
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

D. Pohon Keputusan (*Decision Tree*)

Pohon keputusan merupakan jenis pohon yang digunakan untuk mengidentifikasi kemungkinan pilihan. Pohon keputusan terdiri atas akar dan upapohon yang bercabang terus menerus berupa pemilihan biner atau lebih dari dua pilihan.



Gambar 2.5 Decision Tree
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

E. *Random Forest*

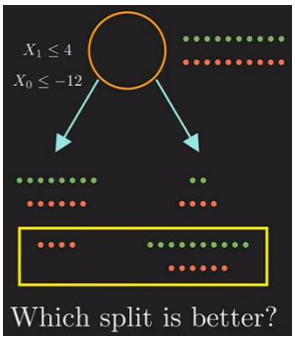
Random Forest adalah algoritma dalam *Machine Learning* yang digunakan untuk pengklasifikasian *data set* dalam jumlah besar karena fungsinya dapat digunakan untuk banyak dimensi dengan berbagai skala dan performa yang tinggi. Klasifikasi ini dilakukan melalui penggabungan pohon dalam *Decision Tree* dengan cara *training data set*.

Dengan melakukan penggabungan hasil klasifikasi dari beberapa *Decision Tree* maka, kemungkinan kesalahan pada saat *training* akan mengecil dan tentunya akurasi akan semakin tinggi. *Random Forest* ini melakukan *voting* pada hasil klasifikasi tadi dan menghasilkan suatu klasifikasi akhir yang nantinya akan dipakai untuk *data testing*.

III. ANALISIS APLIKASI

A. Decision Tree dan *Random Forest* dalam *Machine Learning*

Dalam pengambilan keputusan dalam *Decision Tree* ada yang dinamakan sebagai *decision nodes* atau pada *Gambar 2.5* yang berwarna biru muda. Dalam pengaplikasiannya, penentuan *decision nodes* dilakukan dengan membandingkan *information gain* dari *split* yang dilakukan pada pohon. *Split* yang diambil adalah dengan *split* yang menghasilkan *information gain* yang maksimal.



Gambar 3.1 Penentuan split pada Decision Tree
Sumber: https://www.youtube.com/@NormalizedNerd

Untuk menghitung *information gain*, kita harus mengetahui informasi yang ada di *state*. Dalam *Gambar 3.1* jika kita melihat *root state*, jumlah *point* berwarna hijau sama dengan jumlah *point* berwarna merah. Ini berarti jika kita memprediksi dari data tersebut, kita memiliki 0.5% kemungkinan untuk benar, artinya *root state* memiliki *impurity* tertinggi.

Cara untuk mengukur hal tersebut digunakan *entropy*. *Entropy* adalah ukuran informasi yang terkandung dalam suatu *state*. Jika *entropy* semakin tinggi, maka ketidak-yakinan atas titik yang dipilih secara acak semakin tinggi dan susah untuk mendeskripsikan *state*-nya. Rumus *entropy* adalah sebagai berikut.

$$Entropy = \sum - p_i \log(p_i)$$
$$p_i = \text{probability of class i}$$

Gambar 3.2 Rumus entropy
Sumber: https://www.youtube.com/@NormalizedNerd

Dengan menggunakan formula tersebut, dapat dihitung split mana yang lebih baik. Hasilnya sebagai berikut.

Gambar 3.3 Hasil perhitungan entropy saat pemilihan split
Sumber: https://www.youtube.com/@NormalizedNerd

Lalu *information gain* dapat dihitung menggunakan rumus berikut.

$$\text{IG} = E(parent) - \sum w_i \; E(child_i)$$

Gambar 3.4 Rumus information gain (IG)
Sumber: https://www.youtube.com/@NormalizedNerd

Hasilnya adalah sebagai berikut.

$$\text{IG}_1 = 1 - \frac{14}{20} \times .99 - \frac{6}{20} \times .91 = 0.034$$
$$\text{IG}_2 = 1 - \frac{4}{20} \times 0 - \frac{16}{20} \times .95 = 0.24$$

Gambar 3.5 Hasil dari perhitungan IG pada kedua split
Sumber: https://www.youtube.com/@NormalizedNerd

Dapat dilihat bahwa *split* yang kedua menghasilkan *information gain* yang lebih besar. Oleh karena itu, *Decision Tree* akan memilih *split* tersebut dan melanjutkannya hingga semua data terproses. Pada kenyataannya, *split* yang dihasilkan akan sangat banyak variasi tergantung dari data yang ada. Model akan memilih *split* terbaik hingga pemrosesan seluruh data selesai. Inilah yang menjadi *class-class* yang akan dipakai untuk mencocokkan data uji.

B. Data Structure Decision Tree dan *Random Forest*

Untuk menerapkan konsep *Random Forest* dalam *Machine Learning* kali ini, penulis membuat struktur kode dalam pemrograman bahasa pemrograman Python untuk memudahkan penggunaanya. Berikut adalah struktur data yang dibuat oleh penulis untuk *Decision Tree* dan *Random Forest*.

```
class Node:
   def __init__(self, feature = None, threshold =
None, left = None, right = None, *, value = None):
      self.feature = feature
      self.threshold = threshold
      self.left = left
      self.right = right
      self.value = value

   def is_leaf_node(self):
      return self.value is not None

class DecisionTree:
   def __init__(self, min_samples_split = 2,
max_depth = 100, n_features = None):
      self.min_samples_split = min_samples_split
      self.max_depth = max_depth
      self.n_features = n_features
      self.root = None
```

```
class RandomForest:
   def __init__(self, n_trees = 10, max_depth =
10, min_samples_split = 2, n_feature = None):
      self.n_trees = n_trees
      self.max_depth = max_depth
      self.min_samples_split = min_samples_split
      self.n_features = n_feature
      self.trees = []
```

Beberapa fungsi pembantu juga diperlukan, seperti fungsi untuk membuat *tree*, traversal pada *tree* (rekursif), menghitung *information gain*, *entropy*, dan lainnya. Untuk lengkapnya dapat dilihat pada https://github.com/bangkitdc/handwritten-recognition .

C. Data Training

Dalam *training* suatu model dalam *Machine Learning*, diperlukan data uji dengan jumlah yang banyak. Semakin banyak jumlah data yang diuji, semakin baik hasil data *training* yang ada, tingkat akurasi model akan relatif meningkat. Dalam pembelajaran mesin ini dilakukan 2 tahapan. Untuk mendapatkan model dari hasil pembelajaran mesin, dilakukan *Data Training*. Hasil dari *Data Training* inilah yang akan dipakai untuk menguji suatu pemodelan yang dalam kasus ini adalah gambar angka tulis tangan. Untuk menguji suatu test terhadap model dari *Data Training*, dinamakan *Data Testing*.

Sampel data berkaitan dengan gambar angka tulis tangan memiliki varians yang tinggi, hal ini disebabkan karena cara tulis tangan orang yang berbeda-beda. Oleh karena itu, dibutuhkan data yang banyak agar meminimalisir terjadinya kesalahan saat pengujian nantinya.

Pada makalah ini digunakan *data set* publik yang berjudul "Optical Recognition of Handwritten Digits Data Set" yang dikeluarkan oleh *UCI Machine Learning Repository* sebagai perwujudan sampel data asli. Data yang diberi cukup beragam dan mengandung banyak contoh sehingga cukup untuk dijadikan sebagai bahan model dalam *Machine Learning* tingkat rendah hingga menengah. Dalam kasus ini *data set* mengandung beberapa data dalam matriks 8x8 yang sudah diproses sebelumnya dari gambar agar memudahkan dalam pengaplikasiannya pada Tree. Namun, penulis di sini menggunakan gambar nyata dan tulisan dari penulis sendiri untuk tahapan *testing* agar dapat diaplikasikan langsung dalam kehidupan sehari-hari.

Dengan menggunakan algoritma pemrosesan *Machine Learning* yang telah dijelaskan sebelumnya, *Data Testing* dapat dilakukan. Namun sebelum itu penulis akan memperlihatkan seberapa akurat *data set* yang telah diambil dengan menggunakan algoritma yang telah penulis buat sebelumnya. Penulis juga akan memperlihatkan perbandingan jika menggunakan algoritma *Random Forest* dengan hanya menggunakan algoritma *Decision Tree* saja.

```
data = datasets.load_digits()
X, y = data.data, data.target
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size = 0.2,
random_state = 42)

def accuracy(y_test, y_pred):
    return np.sum(y_test == y_pred) / len(y_test)

# Decision Tree
clf1 = DecisionTree(max_depth = 10)
clf1.fit(X_train, y_train)
predictions1 = clf1.predict(X_test)

acc1 = accuracy(y_test, predictions1)
print(f'Decision Tree Accuracy: {acc1}')

# Random Forest
clf2 = RandomForest(n_trees = 10)
clf2.fit(X_train, y_train)
predictions2 = clf2.predict(X_test)

acc2 = accuracy(y_test, predictions2)
```

```
print(f'Random Forest Accuracy: {acc2}')
```

```
Decision Tree Accuracy: 0.8944444444444445
Random Forest Accuracy: 0.9555555555555556
```

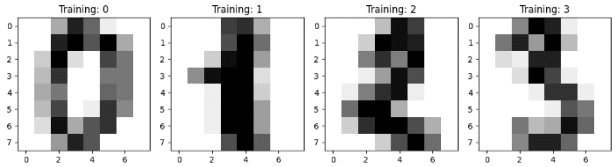
Dapat dilihat bahwa dengan menggunakan algoritma *Random Forest* akurasi dari pemodelan yang penulis buat lebih baik 7% dibandingkan jika hanya menggunakan algoritma *Decision Tree*. Dalam *Machine Learning*, angka 7% ini sangatlah besar dan dapat memengaruhi hasil uji nantinya. Oleh karena itu, algoritma *Random Forest* jauh lebih baik dibandingkan dengan hanya menggunakan algoritma *Decision Tree* saja. Semakin besar akurasi dari pemodelan yang dihasilkan, maka semakin akurat juga hasil data ujinya.

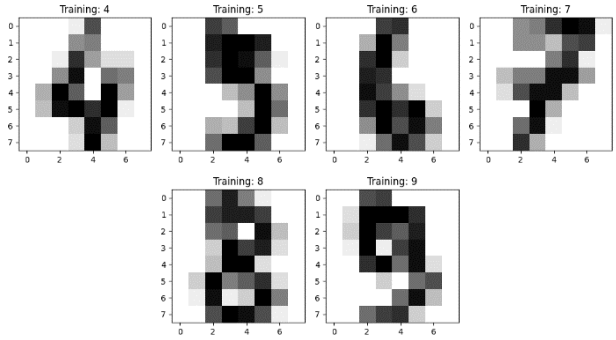
Sebelum lanjut ke tahapan pengujian, alangkah baiknya penulis memberi tahu mengenai isi dari setiap *dataset* yang akan digunakan. Berikut adalah detail terkait data yang dipakai.

```
print(f'Banyak data: {len(X)} \n')
print(X.dtypes)
print(f'\nData dalam matriks:\n
{data.images[0]}')
```

```
Banyak data: 1797

pixel_0_0    float64
pixel_0_1    float64
pixel_0_2    float64
pixel_0_3    float64
pixel_0_4    float64
             ...
pixel_7_3    float64
pixel_7_4    float64
pixel_7_5    float64
pixel_7_6    float64
pixel_7_7    float64
Length: 64, dtype: object

Data dalam 8x8 matriks:
 [[ 0.  0.  5. 13.  9.  1.  0.  0.]
  [ 0.  0. 13. 15. 10. 15.  5.  0.]
  [ 0.  3. 15.  2.  0. 11.  8.  0.]
  [ 0.  4. 12.  0.  0.  8.  8.  0.]
  [ 0.  5.  8.  0.  0.  9.  8.  0.]
  [ 0.  4. 11.  0.  1. 12.  7.  0.]
  [ 0.  2. 14.  5. 10. 12.  0.  0.]
  [ 0.  0.  6. 13. 10.  0.  0.  0.]]
```

Dapat dilihat bahwa *data set* yang digunakan berisi 1797 variasi dengan setiap data mengandung matriks 8x8. Matriks 8x8 inilah yang nantinya akan diklasifikasikan berdasarkan *class*-nya masing-masing. Pada *data set* ini terdapat 10 *classes*, yaitu *class* 0 hingga 9, yang masing-masing *class* menandakan representasi dari angka tersebut. Berikut contoh data untuk representasi masing-masing angka.





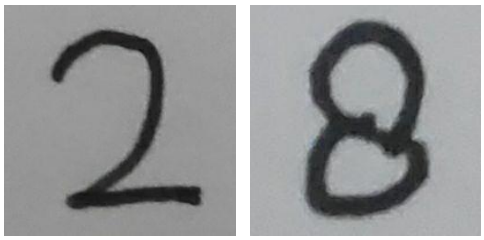
Gambar 3.6 Contoh data set yang tersedia
Sumber: Dokumen Penulis

Karena akurasi dalam pemodelan sudah sangat baik yakni 96%, maka tahapan *Data Testing* dapat dilakukan.

D. Data Testing Sampel Gambar Angka Tulis Tangan

Pemanfaatan praktikal dari model yang telah dibuat digunakan untuk membaca tulisan tangan yang ada. Dengan memanfaatkan pengenalan pola (*pattern* recognition) dari model pohon, data uji yang akan digunakan pada *Random Forest Algorithm* telah dibuat.

Pada pengujian kali ini, data yang akan digunakan adalah tulisan tangan dari penulis. Data berbentuk .png, namun dimensi gambar dibuat 8x8 pixel agar bisa dikenali polanya dari *data set*. Misalkan data yang akan diuji adalah sebagai berikut.

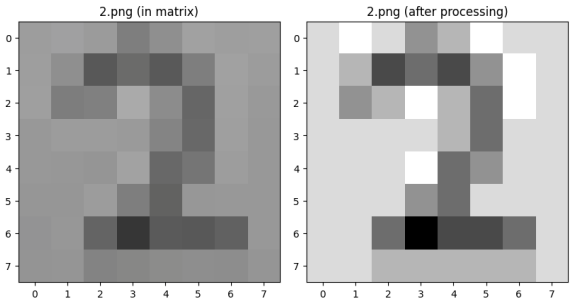


Gambar 3.7 Contoh data test (tulisan tangan penulis) (2.png dan 8.png)
Sumber: Dokumen Penulis

Gambar 2.png dan 8.png ini berukuran 200x200 pixels, kita harus me-*resize* terlebih dahulu menjadi 8x8 pixels agar dapat dibandingkan dengan *data* set, lalu memulai pencocokkan pada *data set*. Berikut contoh *Data Testing* pada gambar 2.png.

```
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
import numpy as np
import cv2

path = '2.png'

def rgb2gray(rgb):
   return np.dot(rgb[...,:3],[0.299,0.587,0.114])

img = mpimg.imread(path)

plt.imshow(img, cmap=plt.cm.gray_r, interpolation
= 'nearest')
plt.title('2.png (in matrix)')
plt.show()

gray = rgb2gray(img)
```

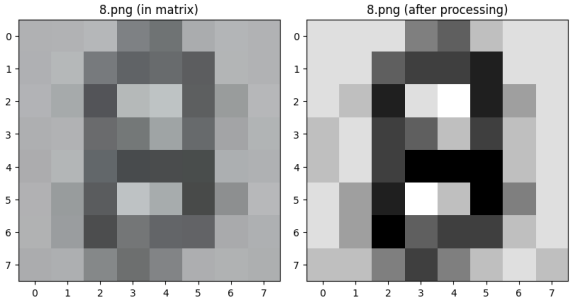
```
a=(16-gray*16).astype(int)

plt.imshow(a, cmap=plt.cm.gray_r, interpolation =
'nearest')
plt.title('2.png (after processing)')
plt.show()

predicted = clf2.predict(a.flatten().reshape(1, -
1))

print("source data in 8x8:\n",a)
print(f'\nHasil prediksi angka dari gambar adalah
angka {predicted[0]}')
```



Gambar 3.8 Hasil pemrosesan data test (2.png)
Sumber: Dokumen Penulis

```
source data in 8x8:
 [[ 6  5  6  8  7  5  6  6]
 [ 6  7 10  9 10  8  5  6]
 [ 6  8  7  5  7  9  5  6]
 [ 6  6  6  6  7  9  6  6]
 [ 6  6  6  5  9  8  6  6]
 [ 6  6  6  8  9  6  6  6]
 [ 6  6  9 12 10 10  9  6]
 [ 6  6  7  7  7  7  7  6]]

Hasil prediksi angka dari gambar adalah angka 2
```

Hal yang sama dilakukan untuk gambar 8.png dan hasilnya adalah sebagai berikut.



Gambar 3.9 Hasil pemrosesan data test (8.png)
Sumber: Dokumen Penulis

```
source data in 8x8:
 [[ 4  4  4  7  8  5  4  4]
 [ 4  4  8  9  9 10  4  4]
 [ 4  5 10  4  3 10  6  4]
 [ 5  4  9  8  5  9  5  4]
 [ 5  4  9 11 11 11  5  4]
 [ 4  6 10  3  5 11  7  4]
 [ 4  6 11  8  9  9  5  4]
 [ 5  5  7  9  7  5  4  5]]

Hasil prediksi angka dari gambar adalah angka 8
```

IV. KESIMPULAN

Penerapan prinsip *Decision Tree* dengan mengunakan *Random Forest Algorithm* dapat menghasilkan keputusan yang akurat. Didukung dengan perkembangan teknologi komputasi dan sumber data yang berkualitas tinggi, keputusan dalam pengklasifikasian angka tulis tangan dapat dihasilkan dalam waktu singkat.

Perwujudan model ini dapat menjadi modal bantuan besar untuk selanjutnya digunakan untuk hal yang bermanfaat bagi kehidupan sehari-hari, misalnya untuk *scan* tulisan tangan menjadi digital, *Machine Learning* dalam citra video, dan lainnya. Tentunya, pada kenyataannya akan ada banyak kasus yang lebih kompleks dan memerlukan algoritma yang lebih mutakhir.

V. UCAPAN TERIMA KASIH

Penulis ingin menyampaikan rasa syukur kepada Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, makalah berjudul "Pengenalan Angka Tulis Tangan dengan Mengaplikasikan Tree" dapat terselesaikan dengan baik dan tepat waktu. Penulis juga ingin menyampaikan terima kasih kepada dosen mata kuliah IF2120 Matematika Diskrit, Dr. Ir. Rinaldi Munir, M. T., Dr. Fariska Zakhralativa Ruskanda, S.T., M.T., dan Dr. Nur Ulfa Maulidevi, S. T, M. Sc. atas bimbingannya selama menjalai perkuliahan. Tidak lupa penulis ingin menyampaikan terima kasih kepada pihak dan sumber yang dijadikan referensi dalam pembuatan makalah ini.

REFERENSI

[1] 7wData, Types of Machine Learning Algorithms. Diakses pada 3 Desember 15.30 WIB.
[2] E. Alpaydin, C. Kaynak, Optical Recognition of Handwritten Digits Data Set . Diakses pada 3 Desember 2022, 15.51 WIB.
[3] Belajar Data Science di Rumah, "Algoritma Supervised Vs Unsupervised Learning, Cari Tahu Bedanya!". Diakses pada 3 Desember 2022, 17.17 WIB.
[4] "Random Forest Algorithm Clearly Explained!" Youtube, uploaded by Normalized Nerd, 21 April 2021. Diakses pada tanggal 3 Desember 2022, 18.00 WIB.
[5] "Random Forest Algorithm Clearly Explained!" YouTube, uploaded by Normalized Nerd, 21 April 2021. Diakses pada tanggal 3 Desember 2022, 18. 20 WIB.
[6] Munir,Rinaldi. 2022. Graf (Bag. 1): Bahan Kuliah IF2120 Matematika Diskrit. Merupakan slide bahan ajar perkuliahan. Diakses pada tanggal 3 Desember 2022.
[7] Munir,Rinaldi. 2022. Pohon (Bag. 1): Bahan Kuliah IF2120 Matematika Diskrit. Merupakan slide bahan ajar perkuliahan. Diakses pada tanggal 3 Desember 2022.
[8] Munir,Rinaldi. 2022. Pohon (Bag. 2): Bahan Kuliah IF2120 Matematika Diskrit. Merupakan slide bahan ajar perkuliahan. Diakses pada tanggal 3 Desember 2022.
[9] "English Handwritten Characters" Kaggle, uploaded by Dhruvil Dave. Diakses pada tanggal 9 Desember 2022, 16.10 WIB.

TAUTAN VIDEO

https://youtu.be/NPhFB4Z6Vi4

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Desember 2022

Muhammad Bangkit Dwi Cahyono 13521055