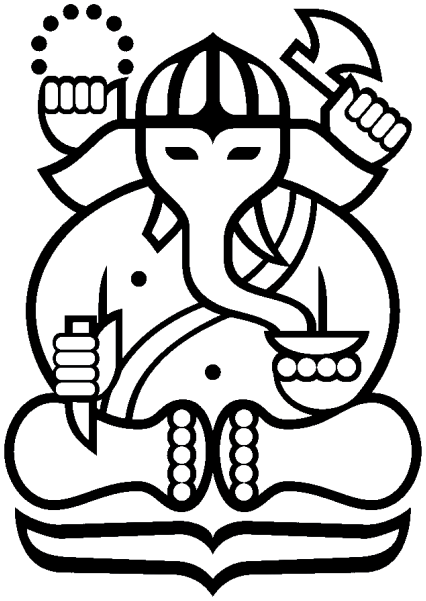
LAPORAN TUGAS BESAR 2 IF3170
INTELEGENSI BUATAN
SEMESTER I 2023-2024

IMPLEMENTASI ALGORITMA KNN DAN NAIVE BAYES



Disusun oleh :

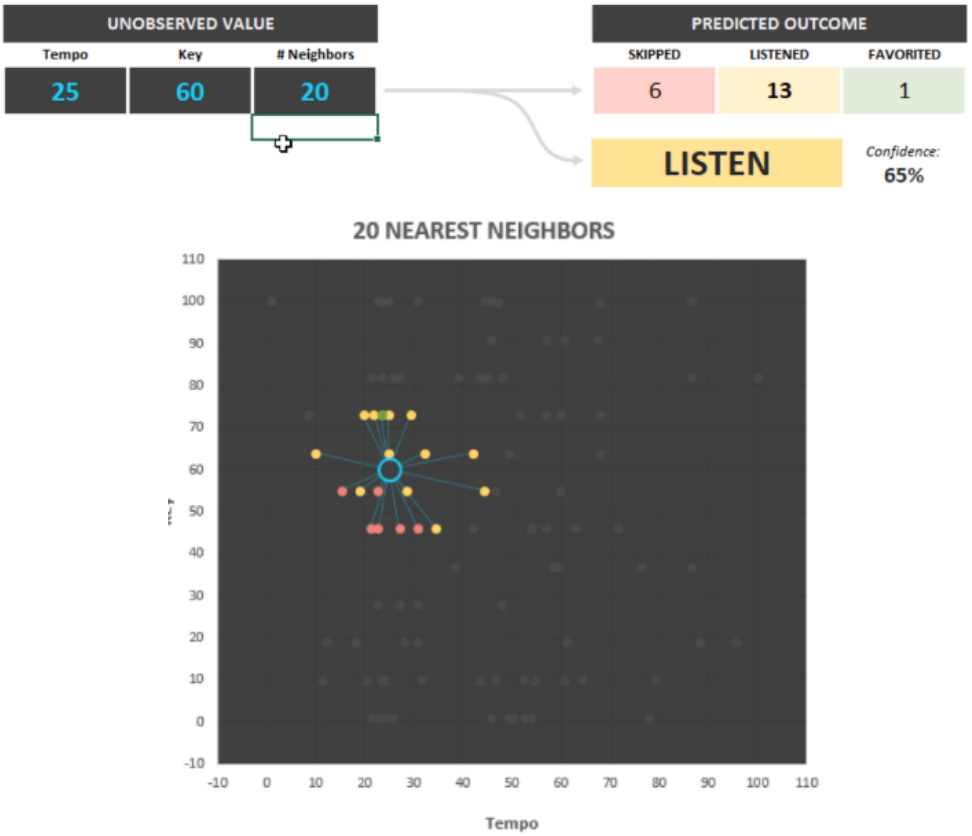
Salomo Reinhart Gregory Manalu	13521063
Ilham Akbar	13521068
Louis Caesa Kesuma	13521069
Addin Munawwar Yusuf	13521085

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2023

K-Nearest Neighbor (KNN)

Algoritma K-*nearest neighbors* – dikenal juga sebagai KNN atau k-NN – adalah sebuah algoritma umum dalam *supervised learning* yang biasa digunakan untuk masalah klasifikasi dan regresi. Algoritma ini bekerja berdasarkan prinsip kedekatan untuk melakukan klasifikasi terhadap poin-poin data individual, yaitu dengan menganggap bahwa objek akan diklasifikasikan berdasarkan mayoritas tetangga terdekatnya.



Gambar 1. Ilustrasi algoritma KNN dalam prediksi data poin baru
[Source : Udemy]

Terdapat 2 tahap dasar dalam pemodelan AI, yaitu *training* dan *test*. Tahap training pada algoritma k-NN sangatlah sederhana, yaitu algoritma hanya mengingat seluruh dataset yang diberikan – sering disebut sebagai *fitting*. Implementasinya dalam python adalah sebagai berikut.

```
class KNN(Model):
    def __init__(self):
        self.k = 19

    # simpan ke memori (variabel lokal)
```

```
    def fit(self, X, y):
        self.X_train = X
        self.y_train = y
```

Sementara itu, pada tahap *test/validation*, algoritma diuji dengan data poin baru yang belum pernah terlihat sebelumnya, kemudian algoritma membuat prediksi klasifikasi dari data tersebut. Ukuran performansi dari k-NN dilakukan dengan 2 variabel, yaitu:

Statistical Terms

True Positive (TP) = Model dengan tepat memprediksi kelas positif (misal. model menebak klasifikasi 2 (dari 0,1,2,3), dan benar, sebenarnya adalah A)

True Negative (TN) = Model dengan tepat memprediksi kelas negatif (misal. model menebak klasifikasi 1, dan ternyata hasil sebenarnya bukan 1 (0, 2, 3))

False Positive (FP) = Model secara keliru memprediksi kelas positif (misal. model menebak bahwa 3 benar dan ternyata hasil sebenarnya adalah 1)

False Negative (FN) = Model secara keliru memprediksi kelas negatif. (misal. model menebak bahwa 0 salah, dan ternyata hasil sebenarnya adalah 0)

1. Accuracy

Ukuran jumlah prediksi yang benar dari data secara keseluruhan . Rumusnya adalah sebagai berikut.

$$\text{Accuracy} = \frac{TP+TN+FP+FN}{TP+TN}$$

2. Precision

Akurasi dari prediksi yang positif. Rumusnya adalah sebagai berikut.

$$\text{Precision} = \frac{TP}{TP+FP}$$

3. Recall

Akurasi dari prediksi yang positif. Rumusnya adalah sebagai berikut.

$$\text{Recall} = \frac{TP}{TP+FN}$$

Langkah-langkah Implementasi

Secara umum, langkah-langkah dari algoritma KNN yang kelompok kami gunakan untuk melakukan prediksi adalah sebagai berikut.

1. **Menentukan nilai K**

K merupakan jumlah tetangga terdekat yang akan digunakan untuk menentukan klasifikasi. Nilai K yang optimal biasanya dipilih melalui proses validasi silang (cross-validation). *Pada kasus yang lebih advanced, penentuan K dilakukan melalui machine learning dan kasus tersebut berada di luar cakupan implementasi ini*

2. **Menghitung Jarak**

Algoritma KNN mengukur jarak antara titik data yang akan diklasifikasikan dengan semua titik data yang sudah disimpan dari training. Metrik jarak yang umum digunakan antara lain adalah jarak Euclidean, jarak Manhattan, atau jarak Minkowski. Pada implementasi kami, jarak yang digunakan adalah jarak euclidean, menimbang jumlah data yang tidak terlalu banyak sehingga penggunaannya masih feasible. Implementasinya dalam python adalah sebagai berikut.

```
        distances = [euclidean_distance(x, x_train) for x_train
in self.X_train]

def euclidean_distance(x1, x2):
    distance = 0
    for i in range(len(x1)):
        distance += (x1[i] - x2[i]) ** 2
    return np.sqrt(distance)
```

3. **Menentukan Tetangga Terdekat**

Setelah menghitung jarak, KNN akan memilih K tetangga terdekat dari titik data yang akan diklasifikasikan. Data tersebut disimpan ke dalam sebuah list.
Implementasinya dalam python adalah sebagai berikut.

```
k_indices = np.argsort(distances)[:self.k]
k_nearest_labels = [self.y_train[i] for i in k_indices]
```

4. **Prediksi berdasarkan Kelas Mayoritas**

Untuk masalah klasifikasi, kelas mayoritas dari K tetangga terdekat akan ditentukan. Berdasarkan mayoritas kelas dari tetangga terdekat, algoritma KNN akan memprediksi kelas dari titik data yang ingin diklasifikasikan. Persentase kemunculan

kelas dari k tetangga teratas menjadi skor kepercayaan diri dari algoritma tersebut. Implementasinya dalam python adalah sebagai berikut.

```
predicted = max(set(k_nearest_labels),
key=k_nearest_labels.count)
```

Kelebihan dari algoritma ini adalah sederhana dan mudah diimplementasikan karena konsepnya relatif sederhana. Selain itu, KNN jarang rentan terhadap overfitting karena tidak memiliki model yang kompleks dan KNN cenderung berhasil dalam mengklasifikasikan data non-linier karena tidak membuat asumsi tentang struktur data.

Perbandingan hasil prediksi dari program knn yang diimplementasikan from scratch dengan program knn yang diimplementasikan menggunakan pustaka *scikit-learn*

Naive-Bayes

Naive Bayes merupakan salah satu metode klasifikasi dalam ranah supervised learning dalam machine learning, yang bertujuan untuk memprediksi kelas atau label dari suatu data berdasarkan fitur-fitur yang diberikan. Dalam konteks supervised learning, model Naive Bayes belajar dari dataset yang telah dilabeli sebelumnya, di mana setiap contoh data memiliki label kelas yang sudah diketahui.

Metode ini dikenal sebagai "*naive*" karena mengadopsi asumsi yang cukup sederhana dan mendasar, yaitu asumsi independensi antara setiap fitur yang digunakan dalam klasifikasi. Dengan kata lain, Naive Bayes menganggap bahwa setiap fitur memberikan kontribusi terhadap klasifikasi secara terpisah, tanpa mempertimbangkan hubungan atau ketergantungan antar fitur. Meskipun asumsi ini sering tidak sesuai dengan kenyataan dalam beberapa kasus, kelebihan Naive Bayes terletak pada sederhananya yang memungkinkan pelatihan model yang cepat dan efisien.

Teorema Bayes

Teorema Bayes adalah dasar dari metode Naive Bayes. Secara matematis, teorema ini dinyatakan sebagai berikut:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

P(A|B): Probabilitas bahwa kejadian A terjadi jika B telah terjadi.

P(B|A): Probabilitas bahwa B terjadi jika A telah terjadi.

P(A) dan P(B): Probabilitas masing-masing kejadian A dan B terjadi secara independen.

Langkah-langkah dalam implementasi Naive-Bayes:

Berikut adalah deskripsi langkah-langkah implementasi Naive Bayes yang kelompok kami telah buat:

1. Pembelajaran (Fit) Model

Dalam metode fit, terdapat proses pembelajaran model Naive Bayes. Looping dilakukan untuk setiap kelas pada dataset. Untuk setiap kelas, perhitungan rata-rata dan deviasi standar dari setiap fitur dilakukan.

2. Perhitungan Probabilitas

Didefinisikan fungsi *gaussProb* untuk menghitung probabilitas distribusi Gaussian (fungsi kepadatan probabilitas) untuk setiap nilai fitur pada setiap kelas.

3. Perhitungan Probabilitas Posterior

Dalam metode *calculate_class_probabilities*, probabilitas posterior untuk setiap kelas dihitung dengan mengalikan probabilitas distribusi Gaussian dari setiap fitur.

4. Prediksi Kelas

Metode *predict* digunakan untuk memprediksi kelas dari suatu data input dengan memilih kelas dengan probabilitas posterior tertinggi.

5. Pembacaan Data dan Pembagian

Data training dan data validasi dibaca dari file CSV. Data dibagi menjadi fitur (X) dan label (y) untuk training dan validasi.

6. Evaluasi Model Naive Bayes (Scratch dan Scikit-Learn)

Model Naive Bayes yang diimplementasikan dari awal digunakan untuk membuat prediksi pada data validasi. Model Naive Bayes dari library Scikit-Learn juga digunakan untuk membandingkan hasilnya dengan implementasi dari awal. Matrik evaluasi seperti akurasi, presisi, dan recall dihitung dan ditampilkan.

7. Penyimpanan Model

Pengguna ditanya apakah ingin menyimpan model yang sudah dibuat. Jika ya, model disimpan ke dalam file dengan format pickle.

Hasil Prediksi

1. Hasil Prediksi KNN

```
Pilih algoritma:
 1 = KNN
 2 = Naive bayes
 3 = Prediksi data baru
1
Apakah ingin menggunakan model?
 1 : ya
 2 : tidak
1
Hasil model KNN:
Akurasi model KNN: 93.83%
Precision model KNN: 90.54%
Recall model KNN: 93.06%

Hasil model KNN sklearn:
Akurasi model KNN sklearn: 93.83%
Precision model KNN sklearn: 93.89%
Recall model KNN sklearn: 93.89%

Apakah ingin menyimpan model?
 1 : ya
 2 : tidak
2
```

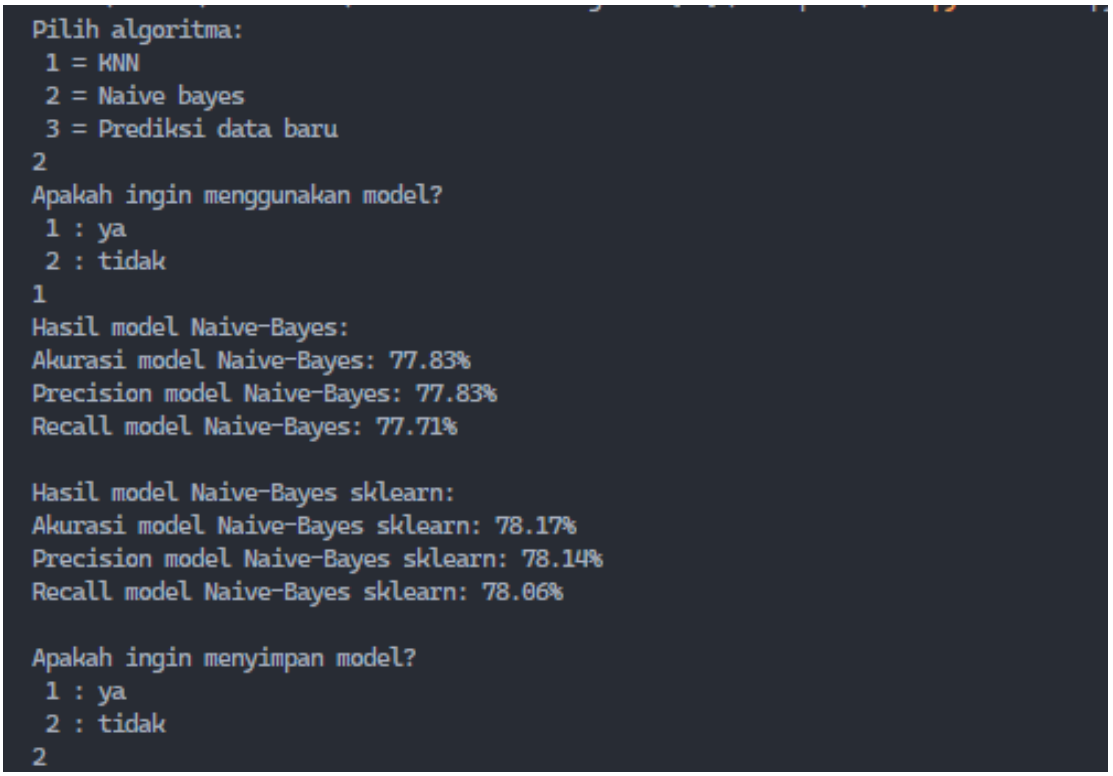
Dari hasil yang disajikan, terlihat adanya dua model K-Nearest Neighbors (KNN) yang berbeda: satu model yang dibuat tanpa menggunakan pustaka scikit-learn dan satu model lainnya yang menggunakan pustaka scikit-learn. Kedua model tersebut dievaluasi menggunakan tiga metrik kinerja utama: akurasi, presisi, dan recall.

Kedua model memiliki tingkat akurasi yang sama, yaitu sebesar 93.83%. Artinya, sekitar 93.83% dari prediksi yang dilakukan oleh keduanya benar. Namun, perbedaan muncul dalam nilai presisi dan recall antara kedua model tersebut. Model KNN yang dibuat tanpa menggunakan pustaka scikit-learn memiliki presisi sebesar 90.54% dan recall sebesar 93.06%. Sementara model KNN yang menggunakan pustaka scikit-learn memiliki presisi dan recall sebesar 93.89%.

Berdasarkan hasil model KNN dan KNN dari Scikit-Learn, kedua model menunjukkan kinerja yang cukup baik dalam mengklasifikasikan data. Akurasi yang sama, sebesar 93.83%, menandakan bahwa sebagian besar data berhasil diklasifikasikan dengan benar oleh kedua model. Namun, terdapat perbedaan pada

metrik presisi dan recall. Presisi model KNN adalah 90.54%, sementara model KNN dari Scikit-Learn memiliki presisi yang sedikit lebih tinggi, yaitu 93.89%. Hasil ini menunjukkan bahwa ketika model memprediksi suatu kelas, model KNN dari Scikit-Learn lebih cenderung memberikan prediksi yang benar. Di sisi lain, recall model KNN adalah 93.06%, sedangkan model KNN dari Scikit-Learn memiliki recall yang sedikit lebih tinggi, yaitu 93.89%. Hal ini menandakan bahwa model KNN dari Scikit-Learn lebih baik dalam menangkap sebagian besar instance yang seharusnya positif.

2. Hasil Prediksi Naive Bayes'



Berdasarkan hasil model Naive-Bayes dan Naive-Bayes dari Scikit-Learn, kedua model menunjukkan kinerja yang serupa dalam mengklasifikasikan data. Akurasi model Naive-Bayes adalah 77.83%, sementara model Naive-Bayes dari Scikit-Learn memiliki akurasi yang sedikit lebih tinggi, yaitu 78.17%. Ini menandakan bahwa sekitar 77-78% dari keseluruhan data berhasil diklasifikasikan dengan benar oleh kedua model. Namun, terdapat kesamaan pada metrik presisi dan recall. Presisi model Naive-Bayes dan Naive-Bayes dari Scikit-Learn sama-sama sebesar 77.83%, menunjukkan bahwa kelas yang diprediksi sebagai positif oleh kedua model memiliki tingkat kebenaran yang serupa. Sementara itu, recall model

Naive-Bayes adalah 77.71%, sedangkan model Naive-Bayes dari Scikit-Learn memiliki recall yang sedikit lebih tinggi, yaitu 78.06%. Hal ini menunjukkan bahwa model Naive-Bayes dari Scikit-Learn lebih baik dalam menangkap sebagian besar instance yang seharusnya positif dibandingkan dengan model Naive-Bayes. Meskipun perbedaan antara keduanya tidak signifikan, hasil ini memberikan gambaran bahwa implementasi Naive-Bayes dari Scikit-Learn dapat memberikan performa yang sedikit lebih baik dibandingkan implementasi dari awal.

Submisi Kaggle

Berikut ini adalah proses dari submisi kaggle yang telah dilakukan. Data latih dibaca dari file *data_train.csv* dan data uji *test.csv* dibaca dari file lain. Data latih dipisahkan menjadi fitur (X_train) dan label (y_train), sedangkan data uji hanya memiliki fitur (X_test). Model akan di-load dari file. Model digunakan untuk melakukan prediksi pada data uji, dan hasil prediksi disimpan dalam data frame *predicted_results*. Hasil prediksi kemudian disimpan dalam file CSV dengan nama *submission.csv*. Model disimpan dalam file dengan format pickle.

Pembagian Kerja

Nama	NIM	Pembagian Kerja
Salomo Reinhart Gregory Manalu	13521063	Naive-Bayes, Laporan
Ilham Akbar	13521068	KNN, Laporan
Louis Caesa Kesuma	13521069	Naive-Bayes, Kaggle, Laporan
Addin Munawwar Yusuf	13521085	KNN, Laporan