SUPERVISED LEARNING – CLASSIFICATION: DATASET KENDARAAN

Observasi

Oleh : Muhammad Shulhannur

1. PENGGUNAAN BAHASA PEMROGRAMAN, TOOLS, DAN LIBRARY

Penulis menggunakan bahasa pemrograman Python, karena diwajibkan, dengan menggunakan compiler online Google Colab, karena Google Colab memberikan kemudahan dalam menulis dokumentasi, markdown dan notes. Adapun hasil visualisasi dari seluruh data yang ditampilkan disini, terlampir langsung pada Google Colab. Sebagai disclaimer, penulis menambahkan bahwa kodingan penulis (09_kendaraan.ipynb), beserta pemanggilan URL di dalamnya dapat berjalan dengan lancar menggunakan Google Colab.

Penulis memanggil library yang terdiri atas :

```
#untuk data analysis and manipulation
import pandas as pd

#untuk matematika
import numpy as np

#untuk matematika
import math

#untuk grafik 2D
import matplotlib.pyplot as plt

#untuk url
import io

#untuk request file from url
import requests

#untuk request file from url
import time
```

2. FORMULASI MASALAH

Diberikan beberapa dataset dan dilakukan pemilihan dataset yang diujikan untuk melakukan learning. Lakukan analisis, desain, dan implementasi Supervised Learning.

3. EKSPLORASI DAN PERSIAPAN DATA

Untuk bagian eksplorasi , dipilih dataset “kendaraan_train.csv” dan dataset "kendaraan_test.csv". Dataset “kendaraan_train.csv” memiliki 285.831 baris data dan dataset “kendaraan_test.csv" memiliki 47.639 baris data. Pada pengambilan data training, kolom ‘id’, ‘Kode_Daerah’,’Umur_Kendaraan’, ’Kanal_Penjualan’, dan ‘SIM’ tidak digunakan pada persiapan data karena kolom tersebut tidak berpengaruh kepada proses learning. Pada pengambilan data testing, kolom ‘Kode_Daerah’,’Umur_Kendaraan’, ’Kanal_Penjualan’, dan ‘SIM’ juga tidak digunakan pada persiapan data karena tidak berpengaruh kepada proses learning.

Sebelum dilakukan normalisasi, data ini harus diproses agar tidak terjadi kesalahan pada learning akibat perbedaan tipe data. Untuk data pada kolom ‘Jenis_Kelamin’, jika data = ‘Pria’ maka akan diberi nilai 1 dan jika data = ‘Wanita’ maka akan diberi nilai 0. Untuk data pada kolom ‘Kendaraan_Rusak’, jika data = ‘Pernah’ akan diberi nilai 1 dan jika data = ‘Tidak Pernah’ akan diberi nilai 0. Setelah tipe data pada data tersebut dirubah, dilakukan pengisian data pada kolom yang memiliki data yang kosong. Kolom data yang kosong tersebut diisi oleh rata-rata dari seluruh nilai pada kolom tersebut.

Data pre-learning ini harus dilakukan scaling/ Normalisasi terlebih dahulu agar nilai data tersebut tidak menyimpang jauh karena perbedaan nilai yang terlalu besar. Normalisasi dilakukan dengan menggunakan rumus :

$$x_n = \frac{x\ - min}{max\ -\ min}$$

Setelah data dinormalisasi, dipilihlah kolom yang akan digunakan pada proses learning. Kolom yang diambil adalah 'Jenis_Kelamin', 'Umur' , 'SIM', 'Sudah_Asuransi', 'Kendaraan_Rusak', 'Premi', dan 'Lama_Berlangganan'

4. PERMODELAN

Pada permodelan/ pembuatan program, digunakan Jupyter (Google Colab) untuk penggunaan resource yang efisien. Algoritma yang digunakan pada pemodelan adalah K-Nearest Neighbor (k-NN).

Pada tahap ini dilakukan beberapa langkah dalam memproses data pre-learning, berikut tahapannya :

1. Tahap pertama, pengambilan nilai k. Nilai k didapatkan dari banyak neighbor / neighbors yang di tentukan (contoh : k = [1,3,5,7,9]).
2. Tahap kedua, penghitungan distance menggunakan rumus Euclidean $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Dilakukan looping untuk setiap data test dilakukan penghitungan distance Euclidean menggunakan data train. Setelah ditemukan distance setiap data test ke setiap data train, dilakukan sorting berdasarkan nilai paling kecil.
3. Tahap ketiga, penentuan hasil test. Hasil sorting data diolah menggunakan algoritma kNN. Data test akan ditentukan hasil prediksi learningnya berdasarkan neighbor / neighbors yang paling besar nilai positif / negatif-nya. Jika ada banyak neighbor / neighbors yang bernilai positif maka hasil data test akan menjadi positif begitu juga sebaliknya.
4. Tahap keempat, pengecekkan hasil test dengan hasil sesungguhnya. Hasil data test akan dicocokkan dengan hasil pada data training. Jika hasil data test sama dengan hasil data training, maka nilai prediksi yang benar akan bertambah. Jika berbeda, maka nilai prediksi yang salah akan bertambah.
5. Tahap kelima, penghitungan akurasi. Total nilai prediksi benar dan salah akan digabungkan untuk menghitung akurasi dari hasil learning kNN.

5. EVALUASI

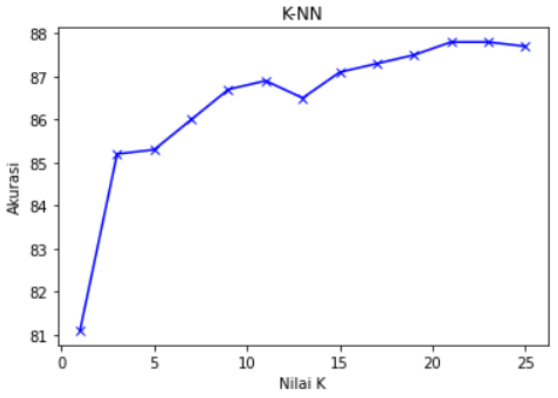
Bagian yang jadi penilaian adalah akurasi learning yang menentukan seberapa akurat hasil train terhadap testing.

6. PENGUJIAN

Pada metode klasifikasi, algoritma yang digunakan adalah algoritma k-NN. Banyak data yang digunakan adalah 5000 data training dan 1000 data test. Banyak neighbor / neighbors (k-Neighbor) yang menjadi acuan adalah 13 kelompok neighbor / neighbors (1,3,5,7,...,25). Penghitungan distance menggunakan rumus Euclidean.

Dari batasan dan aturan eksperimen tersebut didapat hasil learning sebagaimana berikut, dan didapatkan akurasi terbaik pada k=25 dan nilai akurasi sebesar 87,7 :

```
export file  tetangga 1 | k-NN.csv selesai
akurasi  1 - NearestNeighbor : 81.1
Runtime : 422.7732117176056

export file  tetangga 3 | k-NN.csv selesai
akurasi  3 - NearestNeighbor : 85.2
Runtime : 419.2431321144104

export file  tetangga 5 | k-NN.csv selesai
akurasi  5 - NearestNeighbor : 85.3
Runtime : 417.1761510372162

export file  tetangga 7 | k-NN.csv selesai
akurasi  7 - NearestNeighbor : 86.0
Runtime : 416.56434774398804

export file  tetangga 9 | k-NN.csv selesai
akurasi  9 - NearestNeighbor : 86.7
Runtime : 416.7905287742615

export file  tetangga 11 | k-NN.csv selesai
akurasi  11 - NearestNeighbor : 86.9
Runtime : 416.8586688041687

export file  tetangga 13 | k-NN.csv selesai
akurasi  13 - NearestNeighbor : 86.5
Runtime : 418.2335133552551

export file  tetangga 15 | k-NN.csv selesai
akurasi  15 - NearestNeighbor : 87.1
Runtime : 418.97125792503357

export file  tetangga 17 | k-NN.csv selesai
akurasi  17 - NearestNeighbor : 87.3
Runtime : 418.93633913993835

export file  tetangga 19 | k-NN.csv selesai
akurasi  19 - NearestNeighbor : 87.5
Runtime : 417.64794063568115

export file  tetangga 21 | k-NN.csv selesai
akurasi  21 - NearestNeighbor : 87.8
Runtime : 417.79788088798523

export file  tetangga 23 | k-NN.csv selesai
akurasi  23 - NearestNeighbor : 87.8
Runtime : 416.2685465812683

export file  tetangga 25 | k-NN.csv selesai
akurasi  25 - NearestNeighbor : 87.7
Runtime : 412.7276792526245
Total Runtime :  5429.992588996887
```




```
--- Running for 5430.559316396713 seconds ---
```

Pada algoritma dan batasan yang digunakan bisa terlihat bahwa akurasi terbaik berada pada k = 25 dengan nilai akurasi 87,7. Jadi hasil learning terbaik adalah kNN dengan banyak neighbor / neighbors = 25.

7. KESIMPULAN

Dari hasil eksperimen, didapat bahwa akurasi yang paling efektif pada learning ini adalah 87,7 dengan banyak neighbor / neighbors (k-Neighbor) = 25, sehingga banyak data dan banyak neighbor / neighbors mempengaruhi hasil learning kNN. Semakin banyak data yang digunakan maka akan semakin lama waktu eksekusi / runtimeprogram. Untuk penghitungan distance, jika menggunakan rumus selain Euclidean juga bisa merubah hasil dari learning. Didapat juga waktu eksekusi / runtimesetiap kNN hampir sama (nilai sedikit berubah akibat penggunaan resource komputasi yang berubah) .