13522103 - Steven Tjhia

Gaussian Naive Bayes

1. Jelaskan cara kerja dari algoritma tersebut! (boleh dalam bentuk *pseudocode* ataupun narasi)

   ⇒ Pertama, model menghitung rata-rata dan standar deviasi untuk setiap jenis klasifikasi. Data tersebut disimpan untuk digunakan nanti.

   Ketika melakukan prediksi, model akan menghitung *prior probability* terlebih dahulu. *Prior probability* disini dapat dipandang sebagai peluang klasifikasi tanpa mengetahui informasi apapun sebelumnya, jadi semacam peluang kasar.

   Setelah itu lanjut dengan melakukan kalkulasi *likelihood* dengan rumus sebagai berikut:

   $$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

   Setelah nilai *likelihood* didapatkan, kemudian nilai tersebut dijumlahkan dengan nilai *prior probability* dan menjadi *posterior probability*. Nah nilai *posterior probability* inilah yang akan menentukan pemilihan jenis klasifikasi dengan mengambil jenis klasifikasi yang memiliki nilai *posterior probability*.

2. Bandingkan hasil evaluasi model from scratch dan *library*, bagaimana hasil perbandingannya? Jika ada perbedaan, jelaskan alasannya!

Akurasi [holdout 80-20]	
Scratch	***Library***
0.611	0.611
Akurasi [k-fold 5]	
Scratch	***Library***
0.610	0.610
Waktu Eksekusi	
Scratch	***Library***
0.091 detik	0.004 detik

Berdasarkan data hasil percobaan, didapatkan bahwa nilai akurasi yang dihasilkan tidak berbeda. Tetapi dari segi waktu eksekusi, model GNB dari *library* lebih cepat sekitar 20 kali daripada model yang dibuat sendiri. Hal tersebut dikarenakan implementasi model pada *library* melakukan optimasi agar model dapat berjalan dengan cepat sedangkan model yang dibuat sendiri belum mengimplementasikan hal tersebut.

3. Jelaskan *improvement* apa saja yang bisa Anda lakukan untuk mencapai hasil yang lebih baik dibandingkan dengan hasil yang Anda punya saat ini! *Improvement* yang dimaksud tidak terbatas pada bagaimana algoritma diimplementasikan, namun juga mencakup tahap sebelum *modeling and validation*.

   ⇒ Gaussian – > Distribusi Normal : model ini cocok untuk data yang terdistribusi normal, jadi jika data yang dimiliki tidak terdistribusi normal maka hasilnya menjadi kurang baik. Namun hal tersebut dapat diatasi dengan melakukan transformasi data, bisa dengan transformasi logaritmik, box-cox, quantile, ataupun winsorizing. Tentunya hal tersebut belum menjamin 100% akan menjadi lebih baik, tetap harus mempertimbangkan aspek yang lain juga.