Tugas Besar 2 IF3170 Inteligensi Buatan

Implementasi Algoritma

Oleh:

Febryan Arota Hia	K2	13521120
Michael Utama	K1	13521137
Johann Christian Kandani	K2	13521138
Kandida Edgina Gunawan	K1	13521155

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2023/2024

DAFTAR ISI

Implementasi K-Nearest Neighbors Algorithm (KNN)

Algoritma *K-Nearest Neighbors* diimplementasikan dengan cara naif, yaitu dengan melakukan penghitungan jarak seluruh data validasi terhadap setiap data train, kemudian dicari kelas terbanyak diantara k data *train* terdekat. Data train maupun validasi dinormalisasi terlebih dahulu sehingga range data keduanya berada di antara 0 dan 1.

Seluruh kolom pada *dataset* digunakan untuk menentukan jarak antar *data points*, namun masing-masing memiliki *weight* sesuai pearson correlation coefficient. Pembobotan tersebut dilakukan supaya kolom yang memiliki korelasi tinggi dapat memiliki pengaruh lebih besar terhadap jarak data. Proses voting untuk klasifikasi juga diberi bobot sebesar *inverse distance*, sehingga titik yang lebih dekat memiliki pengaruh lebih besar pada hasil klasifikasi. Perhitungan jarak yang digunakan adalah L2 norm (*euclidean distance*).

Nilai k yang kami gunakan ditentukan dengan mencoba knn dengan berbagai k di sekitar akar kuadrat banyak *data training*, lalu memilih k di area dengan *error* terkecil.

Implementasi Naive-Bayes

- **Multinomial Naive-Bayes Classifier**

  Pada algoritma Multinomial Naive-Bayes Classifier, seluruh kolom pada dataset dianggap sebagai fitur diskret. Oleh karena itu, untuk dapat menerapkan algoritma ini, diperlukan manipulasi terhadap dataset awal. Dataset yang diberikan terbukti mengandung beberapa kolom numerik, khususnya yang memiliki nilai kontinu, yang perlu didiskretisasi. Kolom-kolom numerik ini menjalani proses *binning*, yaitu suatu metode yang digunakan untuk mengubah data kontinu menjadi data diskret dengan cara membaginya ke dalam beberapa interval atau bin. Banyaknya bin untuk setiap kolom ditentukan melalui metode *hyperparameter tuning*, dimana eksperimen dilakukan berulang kali untuk menemukan jumlah bin yang memberikan akurasi tertinggi saat diuji menggunakan validation set. Melalui iterasi berulang ini, ditemukan bahwa jumlah bin yang paling optimal untuk tiap kolom terdiskretisasi adalah sebanyak 4.

  Implementasi dari algoritma ini dibantu dengan penciptaan beberapa kelas, yaitu kelas Attribute, Category, dan Naive Bayes. Kelas Attribute merepresentasikan setiap kolom yang terdapat pada *dataset*. Setiap atribut memiliki *array of* Category yang merupakan *array* yang berisi *value-value* yang mungkin dimiliki oleh setiap *record* untuk

Attribute tersebut. Pada kelar Category, disimpan nilai *lower bound* dan *upper bound* untuk kategori tersebut serta disimpan pula nilai probabilitas P(category-value | price_range = 0) , P(category-value | price_range = 1), P(category-value | price_range = 2), dan P(category-value | price_range = 3).

Ide utama dari Naive Bayes adalah sebagai berikut.

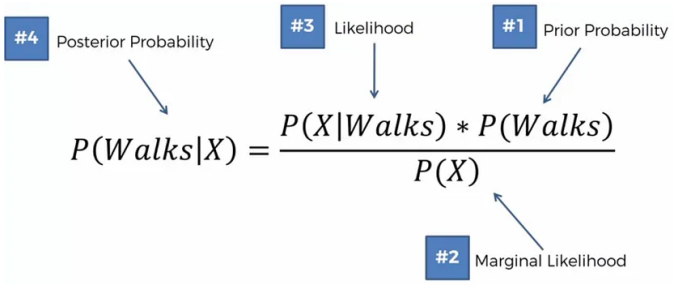
$$P(y \mid x_1, x_2, \ldots, x_n) = \frac{P(x_1 \mid y)\, P(x_2 \mid y) \ldots P(x_n \mid y) P(y)}{P(x_1)\, P(x_2) \ldots P(x_n)}$$

$$P(y|x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

Variabel target yang ingin diprediksi adalah kolom *price_range* yang memiliki 4 kemungkinan nilai, yaitu 0, 1, 2, atau 3. Dengan demikian untuk setiap *row* akan dihitung probabilitas nilai *price_range* 0, 1, 2, dan 3 jika diketahui atribut-atribut prediktor lainnya. Probabilitas-probabilitas tersebut akan dibandingkan dan akan diambil nilai *price_range* dengan probabilitas terbesar jika diketahui nilai dari atribut-atribut prediktor lainnya. Nilai *price_range* tersebutlah yang akan menjadi *target value* pada *row* tersebut.

- **Gaussian Naive-Bayes**

Pada algoritma Gaussian Naive-Bayes, klasifikasi ditentukan berdasarkan probabilitas posterior yang dirumuskan seperti pada gambar di bawah.



Adapun langkah-langkah yang dilakukan dalam pembentukan model Gaussian Naive-Bayes:

1. Melakukan normalisasi pada data train dan data validasi
2. Menghitung mean dan variansi pada setiap data fitur dalam data latih
3. Menghitung probabilitas *likelihood* untuk setiap fitur dengan melibatkan perhitungan distribusi normal seperti pada rumus di bawah.

$$P(X|Y=c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{\frac{-(x-\mu_c)^2}{2\sigma_c^2}}$$

4. Untuk meningkatkan performansi, digunakan log probabilities untuk membantu mengatasi masalah akurasi numerik yang terkait dengan perkalian probabilitas yang sangat kecil. Hal ini bertujuan untuk mengurangi risiko kehilangan presisi saat mengalikan banyak probabilitas kecil bersama-sama. Sehingga posterior probability dapat dirumuskan sebagai berikut.

$$\begin{aligned}\log p(C_k \mid \mathbf{x}) &\propto \log\left(p(C_k)\prod_{i=1}^{n} {p_{ki}}^{x_i}\right)\\ &= \log p(C_k) + \sum_{i=1}^{n} x_i \cdot \log p_{ki}\\ &= b + \mathbf{w}_k^\top \mathbf{x}\end{aligned}$$

5. Melakukan klasifikasi data uji berdasarkan probabilitas posterior. Pada tahap ini pilih probabilitas posterior tertinggi sebagai hasil prediksi.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}}[\log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j)]$$

Perbandingan Hasil Prediksi

- **Prediksi algoritma KNN**

Metrik	Hasil Implementasi	Hasil Pustaka
Accuracy	0.935	0.935
Precision	0.9348889404477712	0.9348889404477712
Recall	0.9352940689718892	0.9352940689718892

Hasil implementasi algoritma KNN menghasilkan hasil yang sama dengan pustaka, karena metodologi implementasi yang digunakan sama.

- **Prediksi algoritma Gaussian Naive-Bayes**

Metrik	Hasil Implementasi	Hasil Pustaka
Accuracy	0.78	0.7816666666666666
Precision	0.7804663979309369	0.7813718889805846
Recall	0.7788856605345197	0.7806462239148014

Hasil implementasi Gaussian Naive-Bayes menghasilkan prediksi dengan tingkat yang serupa dengan hasil prediksi algoritma pada pustaka *sklearn*. Hal ini karena metodologi implementasi yang digunakan serupa, perbedaan dapat dihasilkan oleh proses perhitungan yang berbeda sehingga kestabilan angka tidak sama.

- **Prediksi algoritma Multinomial Naive-Bayes**

Metrik	Hasil Implementasi	Hasil Pustaka
Accuracy	0.7666666666666667	0.5316666666666666
Precision	0.7700290185190727	0.5194204225246274
Recall	0.7656934306352077	0.5323794448063915

Tingkat akurasi lebih tinggi pada hasil implementasi sendiri dihasilkan oleh penanganan data kategorikal tersendiri, sementara data pada kolom lain kemungkinan dianggap sebagai data numerikal pada pustaka *sklearn*. Data kategorikal yang dianggap sebagai data numerikal dapat mengakibatkan proses pengkategorian (*binning*) yang tidak akurat terutama jika data numerik tidak terdistribusi normal.

Kaggle

Sebelum data dimodelkan menggunakan algoritma-algoritma yang telah dijelaskan, data pada umumnya diproses terlebih dahulu. Metode-metode *preprocessing* yang dijalankan adalah mengimputasi nilai-nilai yang tidak valid seperti nilai 0 pada kolom "*sc_w*", menghapus baris-baris data dengan nilai tidak valid, dan juga menghapus kolom-kolom yang memiliki banyak data tidak valid. Namun setelah melakukan beberapa percobaan terhadap metode-metode *preprocessing* data, hasil prediksi terbaik (menggunakan algoritma KNN) adalah ketika menggunakan data yang tidak dilakukan *preprocessing.* Data *testing* akan dilabeli menggunakan data latih menggunakan algoritma-algoritma yang telah dijelaskan. Data baru yang berupa hasil pelabelan akan dimuat ke dalam file csv untuk dikumpulkan ke *kaggle*.

Pembagian Tugas

NIM	Nama	Tugas
13521120	Febryan Arota Hia	Naive Bayes
13521137	Michael Utama	KNN
13521138	Johann Christian Kandani	KNN
13521155	Kandida Edgina Gunawan	Naive Bayes

Link Github

https://github.com/kandidagunawan/Tubes2_AI