1. Jelaskan cara kerja dari algoritma tersebut!
Jawab:
Gaussian Naive Bayes adalah algoritma klasifikasi probabilistik berdasarkan Teorema Bayes dengan asumsi bahwa setiap fitur bersifat independen atau tidak saling terkait (naive). Algoritma ini menggunakan distribusi Gaussian (normal) untuk menghitung probabilitas likelihood dari data kontinu (numerik). Cara kerjanya adalah sebagai berikut:

- Hitung prior untuk setiap kelas, yaitu probabilitas awal dari setiap kelas sebelum memperhitungkan fitur. Dihitung sebagai jumlah data pada kelas dibagi dengan total data.
- Untuk setiap fitur numerik dalam data, kita menghitung mean dan variance dari nilai-nilai fitur tersebut dalam setiap kelas. Gaussian Naive Bayes menggunakan mean dan variance untuk menghitung probabilitas likelihood berdasarkan distribusi Gaussian.
- Hitung likehood, yaitu probabilitas dari fitur yang muncul di suatu kelas. Pada Gaussian Naive Bayes, kita menghitung likelihood menggunakan distribusi Gaussian (normal). Rumus untuk distribusi Gaussian adalah:

  $$P(x_i|C) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)$$

  x_i adalah nilai dari fitur x untuk data yang kita prediksi,

  C adalah kelas,

  μ adalah mean dari fitur dalam kelas C,

  σ^2 adalah variance dari fitur dalam kelas C.
- Hitung posterior probability untuk setiap kelas, yaitu probabilitas dari suatu kelas diberikan fitur-fitur yang ada. Teorema Bayes digunakan untuk menghitung ini:

  $$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

  P(C|X) adalah probabilitas kelas C diberikan data fitur X (yang ingin kita hitung),
  P(X|C) adalah likelihood dari fitur X untuk kelas C (dihitung di langkah sebelumnya),
  P(C) adalah prior probability dari kelas C,
  P(X) adalah probabilitas keseluruhan dari fitur.
- Kelas dengan probabilitas posterior terbesar adalah prediksi model. Jadi, setelah menghitung probabilitas posterior untuk semua kelas, pilih kelas yang memiliki nilai terbesar.

4. Bandingkan hasil evaluasi model pada nomor 2 dan 3, bagaimana hasil perbandingannya? Jika ada perbedaan, jelaskan alasannya!
Jawab:

Menggunakan Implementasi From Scratch

```
import importlib
import gaussian_naive_bayes
importlib.reload(gaussian_naive_bayes)

from gaussian_naive_bayes import GaussianNaiveBayes

gnb = GaussianNaiveBayes()

# Using holdout validation
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)
ho_accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {ho_accuracy}")

# Using k-fold cross-validation
cv_accuracy = cross_val_score(gnb, X, y, cv=kfold, scoring="accuracy").mean()
print(f"Cross-validation Accuracy: {cv_accuracy}")
```

✓ 0.3s

```
Accuracy: 0.7868852459016393
Cross-validation Accuracy: 0.784863387978142
```

Menggunakan Implementasi dari Library

```
gnb_lib = GaussianNB()

# Using holdout validation
gnb_lib.fit(X_train, y_train)
y_pred_lib = gnb_lib.predict(X_test)
ho_accuracy_lib = accuracy_score(y_test, y_pred_lib)
print(f'Accuracy: {ho_accuracy_lib}')

# Using k-fold cross-validation
cv_accuracy_lib = cross_val_score(gnb_lib, X, y, cv=kfold, scoring="accuracy").mean()
print(f'Cross-validation Accuracy: {cv_accuracy_lib}')
```

✓ 0.1s

```
Accuracy: 0.7868852459016393
Cross-validation Accuracy: 0.784863387978142
```

Kedua model memberikan hasil akurasi yang sama, baik pada holdout validation maupun pada cross-validation.

5. 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.

Jawab:

- Melakukan pemilihan fitur yang paling informatif atau melakukan metode seleksi fitur seperti chi-square test atau mutual information untuk meningkatkan performa model.