NAMA : FEBI RAHMADIA PUTRI

NIM : 20220021

PRAK2. ALGORITMA PROBALISTIK

1. Jelaskan apa yang dimaksud dengan algoritma probabilistik, dan berikan contoh penerapannya pada kehidupan nyata.

   Jawab :

   Algoritma probabilistik adalah algoritma yang menggunakan konsep probabilitas untuk memodelkan dan memecahkan masalah. Dalam algoritma probabilistik, solusi atau hasil yang dihasilkan bersifat probabilitas atau mengandung unsur ketidakpastian. Algoritma ini menggunakan prinsip statistik dan teori probabilitas untuk mengestimasi dan memprediksi hasil berdasarkan data yang tersedia.

   Contoh penerapan algoritma probabilistik dalam kehidupan nyata adalah:

   a. Deteksi Spam Email: Algoritma probabilistik dapat digunakan untuk membangun model yang dapat membedakan email yang masuk antara spam dan bukan spam. Dengan mempelajari pola dan karakteristik email yang diketahui sebagai spam atau bukan spam, algoritma probabilistik seperti Naive Bayes dapat menghitung probabilitas bahwa suatu email baru adalah spam atau bukan berdasarkan fitur-fitur yang diamati, seperti kata kunci, pengirim, dan struktur email.

   b. Prediksi Cuaca: Algoritma probabilistik juga dapat digunakan dalam memprediksi cuaca. Dengan memanfaatkan data historis cuaca, data sensor cuaca saat ini, dan model statistik, algoritma seperti Bayesian networks atau Hidden Markov models dapat memberikan perkiraan probabilitas untuk berbagai kondisi cuaca di masa depan. Hasil prediksi ini dapat digunakan untuk memberikan peringatan dini tentang cuaca buruk atau membantu perencanaan aktivitas luar ruangan.

   c. Sistem Rekomendasi: Algoritma probabilistik digunakan dalam banyak sistem rekomendasi, seperti rekomendasi produk di situs web e-commerce atau rekomendasi konten di platform streaming. Algoritma ini mengumpulkan informasi tentang preferensi pengguna dan menggunakan model probabilistik untuk memprediksi kemungkinan suatu item akan disukai oleh pengguna berdasarkan data historis. Contoh algoritma probabilistik yang sering digunakan adalah Collaborative Filtering dan Bayesian Personalized Ranking.

   d. Pengenalan Suara: Algoritma probabilistik juga digunakan dalam sistem pengenalan suara. Melalui analisis data suara dan pengenalan pola, algoritma

probabilistik seperti Hidden Markov models dapat mengestimasi probabilitas bahwa suara yang diucapkan adalah kata atau frase tertentu. Hal ini dapat diterapkan dalam aplikasi pengenalan suara, seperti asisten virtual atau sistem otomatisasi telepon.

Algoritma probabilistik memberikan pendekatan yang kuat untuk memodelkan dan memecahkan masalah di mana keberadaan ketidakpastian atau variasi adalah faktor penting. Dengan menggunakan konsep probabilitas, algoritma ini dapat memberikan perkiraan yang berguna dan solusi yang dapat diandalkan dalam banyak domain kehidupan nyata.

2. Jelaskan strategi atau pendekatan apa yang digunakan dalam algoritma probabilistik untuk memperkirakan solusi atau output yang benar, dan berikan contoh penerapannya pada masalah di dunia nyata.

   Jawab :
   Dalam algoritma probabilistik, terdapat beberapa strategi atau pendekatan yang digunakan untuk memperkirakan solusi atau output yang benar. Berikut adalah beberapa pendekatan yang umum digunakan:

   a. Maximum Likelihood Estimation (MLE): Pendekatan ini mencoba untuk mencari nilai yang paling mungkin atau paling mungkin terjadi berdasarkan data yang diamati. Dalam MLE, parameter yang memaksimalkan likelihood fungsi diperkirakan sebagai solusi. Contohnya, dalam klasifikasi teks, MLE dapat digunakan untuk memperkirakan probabilitas suatu dokumen termasuk dalam kategori tertentu berdasarkan kemunculan kata-kata dalam dokumen tersebut.

   b. Bayesian Inference: Pendekatan ini menggunakan teori Bayesian untuk memperkirakan probabilitas posterior atau kemungkinan terjadinya suatu peristiwa berdasarkan data yang ada. Bayesian inference menggabungkan pengetahuan awal yang disebut prior probability dengan data observasi untuk menghasilkan distribusi probabilitas yang diperbarui atau posterior probability. Contoh penerapannya adalah dalam deteksi penyakit. Dengan menggunakan Bayesian inference, kita dapat memperbarui probabilitas seorang pasien menderita penyakit berdasarkan hasil tes diagnostik dan pengetahuan sebelumnya tentang prevalensi penyakit tersebut.

   c. Hidden Markov Models (HMM): HMM adalah model probabilistik yang digunakan untuk memodelkan urutan kejadian yang tak teramati berdasarkan urutan kejadian yang teramati. HMM terdiri dari dua komponen utama, yaitu state yang tidak teramati dan observasi yang teramati. HMM dapat digunakan dalam berbagai aplikasi seperti pengenalan ucapan, pengenalan tulisan tangan, dan pemodelan urutan genetik.

d. Monte Carlo Methods: Pendekatan ini melibatkan penggunaan simulasi Monte Carlo untuk memperkirakan solusi atau hasil yang benar. Metode ini menggunakan pengulangan acak dan sampel yang diperoleh untuk menghitung nilai yang diinginkan. Contohnya adalah penggunaan Monte Carlo dalam permasalahan optimasi, seperti optimisasi portofolio investasi. Metode ini dapat digunakan untuk menghasilkan sejumlah besar portofolio acak dan memperkirakan portofolio optimal berdasarkan hasil simulasi tersebut.

Penerapan algoritma probabilistik dalam dunia nyata sangat luas dan bervariasi tergantung pada domain masalah yang dihadapi. Misalnya, dalam pengenalan wajah, algoritma probabilistik seperti Bayesian networks dapat digunakan untuk memperkirakan identitas seseorang berdasarkan fitur-fitur wajah yang diamati. Dalam pengenalan bahasa alami, algoritma seperti Hidden Markov models dapat digunakan untuk memperkirakan urutan kata yang paling mungkin dalam suatu kalimat berdasarkan model bahasa yang telah dipelajari.

3. Apa perbedaan antara algoritma deterministik dan algoritma probabilistik? Berikan contoh dari setiap jenis algoritma dan jelaskan bagaimana mereka berbeda.

Jawab :
Perbedaan utama antara algoritma deterministik dan algoritma probabilistik terletak pada cara mereka memproses data dan menghasilkan output. Berikut adalah perbedaan antara keduanya:

a. Algoritma Deterministik:

- Sifat: Algoritma deterministik akan menghasilkan output yang sama untuk input yang sama setiap kali dieksekusi.
- Ketidakpastian: Algoritma deterministik tidak mempertimbangkan unsur ketidakpastian atau probabilitas dalam pemrosesan data.
- Contoh: Bubble Sort adalah salah satu contoh algoritma deterministik yang digunakan untuk mengurutkan elemen dalam suatu larik. Algoritma ini mengikuti langkah-langkah yang terdefinisi secara kaku dan akan menghasilkan output yang sama untuk setiap urutan elemen yang diberikan.

b. Algoritma Probabilistik:

- Sifat: Algoritma probabilistik menghasilkan output yang bersifat probabilitas atau mengandung unsur ketidakpastian.
- Ketidakpastian: Algoritma ini mempertimbangkan distribusi probabilitas atau kemungkinan dalam pemrosesan data.
- Contoh: Naive Bayes adalah contoh algoritma probabilistik yang sering digunakan dalam klasifikasi. Algoritma ini menghitung probabilitas bahwa suatu sampel data masuk ke dalam kelas tertentu berdasarkan fitur-fitur yang

diamati. Misalnya, dalam klasifikasi email sebagai spam atau bukan spam, Naive Bayes mengestimasi probabilitas bahwa suatu email baru adalah spam atau bukan berdasarkan kata-kata yang terdapat dalam email tersebut dan pengalaman sebelumnya.

Perbedaan utama antara algoritma deterministik dan algoritma probabilistik terletak pada keberadaan ketidakpastian dan probabilitas dalam hasil yang dihasilkan. Algoritma deterministik menghasilkan output yang pasti dan konsisten, sementara algoritma probabilistik memberikan perkiraan probabilitas atau estimasi yang bergantung pada data yang diamati.

4. Bagaimana algoritma Naive Bayes digunakan dalam klasifikasi teks atau analisis sentimen? Berikan contoh penerapannya pada bahasa pemrograman Python.

   Jawab :

   Algoritma Naive Bayes adalah salah satu algoritma klasifikasi probabilistik yang sering digunakan dalam analisis teks dan sentimen. Algoritma ini memanfaatkan teorema Bayes dan asumsi naif (naive) bahwa setiap fitur atau kata dalam teks adalah independen secara kondisional terhadap kelas yang ditentukan. Di bawah ini adalah contoh penerapannya dalam bahasa pemrograman Python:

```
from sklearn.feature_extraction.text import CountVect
from sklearn.naive_bayes import MultinomialNB

# Contoh data latih
texts = ["Ini adalah contoh kalimat positif.",
         "Saya senang dengan produk ini.",
         "Produk ini sangat buruk.",
         "Saya merasa kecewa dengan kualitas produk i

# Label kelas yang sesuai dengan setiap kalimat
labels = ["positif", "positif", "negatif", "negatif"]

# Membuat vektor fitur menggunakan CountVectorizer
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(texts)

# Membuat model klasifikasi Naive Bayes
classifier = MultinomialNB()
classifier.fit(X, labels)

# Contoh data uji
test_text = "Produk ini bagus sekali!"

# Mengubah data uji menjadi vektor fitur
test_X = vectorizer.transform([test_text])

# Melakukan prediksi kelas
predicted_label = classifier.predict(test_X)
```

```
print("Prediksi kelas:", predicted_label)
```

```
Prediksi kelas: ['negatif']
```

Dalam contoh ini, kita menggunakan pustaka scikit-learn untuk mengimplementasikan algoritma Naive Bayes. Pertama, kita menyiapkan data latih yang terdiri dari kalimat-kalimat dan label kelas yang sesuai. Kemudian, kita menggunakan CountVectorizer untuk mengubah teks menjadi vektor fitur yang merepresentasikan kemunculan kata-kata dalam teks. Setelah itu, kita membuat model MultinomialNB yang merupakan implementasi Naive Bayes untuk klasifikasi teks dengan distribusi multinomial. Model tersebut dilatih dengan memasukkan matriks fitur X dan label kelas labels.

Selanjutnya, kita menyiapkan data uji, yaitu teks yang ingin kita prediksi kelasnya. Teks tersebut juga diubah menjadi vektor fitur menggunakan CountVectorizer.

Terakhir, kita menggunakan model yang telah dilatih untuk melakukan prediksi kelas dari data uji. Hasil prediksi kelas akan dicetak sebagai output.

Dalam contoh ini, kita menggunakan Naive Bayes untuk klasifikasi sentimen. Label kelas yang digunakan adalah "positif" dan "negatif" yang mengindikasikan apakah kalimat tersebut memiliki sentimen positif atau negatif terhadap suatu produk.

5. Apa yang dimaksud dengan algoritma Hidden Markov Model, dan bagaimana mereka digunakan dalam pengenalan pola? Berikan contoh penerapannya pada bahasa pemrograman Python.

   Jawab :
   Hidden Markov Model (HMM) adalah model statistik yang digunakan untuk memodelkan urutan kejadian yang tak teramati berdasarkan urutan kejadian yang teramati. Model ini terdiri dari sekumpulan state yang tak teramati dan menghasilkan observasi yang teramati. Selain itu, HMM memiliki sifat Markovian, yaitu probabilitas transisi dari satu state ke state lainnya hanya bergantung pada state sebelumnya.

   HMM sering digunakan dalam pengenalan pola, terutama dalam pengenalan ucapan dan pengenalan tulisan tangan. Model ini dapat memodelkan urutan kejadian tak teramati, seperti suara ucapan atau urutan tulisan, dan menghasilkan observasi yang teramati, seperti rentetan fonem atau urutan karakter tulisan.

   Berikut ini adalah contoh penerapan Hidden Markov Model dalam bahasa pemrograman Python menggunakan pustaka hmmlearn:

```python
from hmmlearn import hmm
import numpy as np

# Contoh data latih
X = np.array([[0], [1], [0], [1], [0]])  # Contoh urutan observasi (teramati

# Inisialisasi model HMM
model = hmm.MultinomialHMM(n_components=2)

# Melatih model HMM dengan data latih
model.fit(X)

# Contoh data uji
test_sequence = np.array([[0], [1], [0]])  # Contoh urutan observasi (terama

# Melakukan prediksi urutan state tak teramati
predicted_states = model.predict(test_sequence)

print("Urutan state tak teramati yang diprediksi:", predicted_states)
```

Dalam contoh ini, kita menggunakan pustaka hmmlearn untuk mengimplementasikan Hidden Markov Model. Pertama, kita menyiapkan data latih yang terdiri dari urutan observasi (teramati). Dalam contoh ini, observasi adalah bilangan biner 0 dan 1. Kemudian, kita inisialisasi model HMM dengan menyebutkan jumlah komponen atau state yang tak teramati. Dalam contoh ini, kita menggunakan 2 state.

Selanjutnya, model HMM dilatih dengan memasukkan data latih menggunakan metode fit(). Setelah melatih model, kita dapat menggunakan model tersebut untuk melakukan prediksi urutan state tak teramati dari data uji. Dalam contoh ini, data uji berupa urutan observasi yang belum diketahui urutan state tak teramatinya. Metode predict() digunakan untuk melakukan prediksi.

Hasil prediksi urutan state tak teramati akan dicetak sebagai output. Urutan state ini mengindikasikan prediksi terhadap keadaan tak teramati yang mungkin mendasari urutan observasi yang diberikan.