Nama : Andrea Prasetyo Hariawan

NIM : 20220010

Matakuliah : Praktikum Desain Analisis dan algoritma

Laporan praktikum probalistik

1. Algoritma Naive Bayes

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

# Membuat dataset untuk training dan testing
train_data = ['Ini komentar positif', 'Ini komentar negatif', 'Ini
komentar netral']
train_labels = ['positif', 'negatif', 'netral']
test_data = ['Ini komentar baru']

# Mengubah teks menjadi vektor dengan metode TF-IDF
vectorizer = TfidfVectorizer()
train_vectors = vectorizer.fit_transform(train_data)
test_vectors = vectorizer.transform(test_data)

# Membuat model Naive Bayes
clf = MultinomialNB()

# Melatih model dengan dataset training
clf.fit(train_vectors, train_labels)

# Memprediksi label dari komentar baru
predicted_label = clf.predict(test_vectors)

print(predicted_label)
```

Script di atas digunakan untuk melakukan klasifikasi teks dengan menggunakan metode Naive Bayes dan representasi vektor TF-IDF.

Pertama, kita mendefinisikan dataset untuk pelatihan dan pengujian. train_data berisi komentar-komentar yang digunakan untuk melatih model, train_labels adalah label yang

sesuai dengan setiap komentar pelatihan, dan test_data adalah komentar yang ingin kita prediksi labelnya.

Kemudian, kita membuat instance TfidfVectorizer dari sklearn.feature_extraction.text untuk mengubah teks menjadi vektor dengan metode TF-IDF. TfidfVectorizer menghitung frekuensi istilah (term frequency) dan frekuensi dokumen terbalik (inverse document frequency) untuk menghasilkan representasi vektor yang menggambarkan bobot istilah-istilah dalam teks.

Selanjutnya, kita menggunakan fit_transform pada train_data untuk mengubahnya menjadi vektor TF-IDF yang sesuai dengan model kita. Kemudian, kita menggunakan transform pada test_data untuk mengubahnya menjadi vektor yang dapat digunakan untuk prediksi.

Setelah itu, kita membuat instance MultinomialNB dari sklearn.naive_bayes sebagai model Naive Bayes. Model ini cocok digunakan untuk data yang diwakili oleh bilangan bulat non-negatif, seperti representasi vektor TF-IDF.

Kita melatih model dengan menggunakan fit pada train_vectors dan train_labels.

Terakhir, kita menggunakan predict pada test_vectors untuk memprediksi label dari komentar baru. Hasil prediksi disimpan dalam variabel predicted_label dan dicetak.

Dengan menjalankan script ini, Anda akan mendapatkan prediksi label untuk komentar baru yang diberikan berdasarkan pelatihan menggunakan model Naive Bayes dan representasi vektor TF-IDF.

2. Algoritma Hidden Markov Model (HMM)

```
import numpy as np
from hmmlearn import hmm

# Membuat dataset untuk training dan testing
train_data = np.random.rand(10, 3)
test_data = np.random.rand(1, 3)

# Membuat model Hidden Markov Model
model = hmm.GaussianHMM(n_components=2, covariance_type="full")

# Melatih model dengan dataset training
model.fit(train_data)

# Memprediksi label dari data testing
```

```
predicted_label = model.predict(test_data)

print(predicted_label)
```

Script di atas digunakan untuk membangun dan melatih model Hidden Markov Model (HMM) dengan menggunakan paket hmmlearn dari scikit-learn. Model HMM digunakan untuk melakukan prediksi atau klasifikasi pada data yang memiliki sifat sekuensial.

Pertama, kita mengimpor numpy sebagai np untuk menghasilkan bilangan acak dan hmm dari hmmlearn untuk membuat model Hidden Markov Model.

Kemudian, kita membuat dataset untuk pelatihan dan pengujian. train_data berisi data pelatihan dengan ukuran 10 baris dan 3 kolom, sedangkan test_data berisi data pengujian dengan ukuran 1 baris dan 3 kolom.

Selanjutnya, kita membuat instance GaussianHMM dengan menggunakan n_components=2 yang menunjukkan jumlah komponen (hidden states) dalam model HMM, dan covariance_type="full" yang menunjukkan bahwa kita menggunakan matriks kovarian penuh.

Setelah itu, kita melatih model dengan menggunakan fit pada train_data. Model HMM akan mempelajari parameter dari data pelatihan untuk digunakan dalam prediksi.

Terakhir, kita menggunakan predict pada test_data untuk memprediksi label atau hidden state dari data pengujian. Hasil prediksi disimpan dalam variabel predicted_label dan kemudian dicetak.

Dengan menjalankan script ini, Anda akan mendapatkan prediksi label atau hidden state untuk data pengujian menggunakan model Hidden Markov Model yang telah dilatih dengan data pelatihan.

3. Algoritma Gaussian Mixture Model (GMM)

```
import numpy as np
from sklearn.mixture import GaussianMixture

# Membuat dataset untuk training dan testing
train_data = np.random.rand(10, 3)
test_data = np.random.rand(1, 3)

# Membuat model Gaussian Mixture Model
model = GaussianMixture(n_components=2)
```

```
# Melatih model dengan dataset training
model.fit(train_data)

# Memprediksi label dari data testing
predicted_label = model.predict(test_data)

print(predicted_label)
```

Script di atas digunakan untuk membangun dan melatih model Gaussian Mixture Model (GMM) dengan menggunakan paket sklearn.mixture dari scikit-learn. Model GMM digunakan untuk melakukan pemodelan distribusi probabilitas pada data.

Pertama, kita mengimpor numpy sebagai np untuk menghasilkan bilangan acak dan GaussianMixture dari sklearn.mixture untuk membuat model Gaussian Mixture Model.

Kemudian, kita membuat dataset untuk pelatihan dan pengujian. train_data berisi data pelatihan dengan ukuran 10 baris dan 3 kolom, sedangkan test_data berisi data pengujian dengan ukuran 1 baris dan 3 kolom.

Selanjutnya, kita membuat instance GaussianMixture dengan menggunakan n_components=2 yang menunjukkan jumlah komponen dalam model GMM. Jumlah komponen ini menentukan berapa banyak distribusi Gaussian yang akan dipelajari oleh model.

Setelah itu, kita melatih model dengan menggunakan fit pada train_data. Model GMM akan mempelajari parameter dari data pelatihan untuk mengestimasi distribusi Gaussian yang sesuai dengan data.

Terakhir, kita menggunakan predict pada test_data untuk memprediksi label atau komponen yang sesuai dengan data pengujian. Hasil prediksi disimpan dalam variabel predicted_label dan kemudian dicetak.

Dengan menjalankan script ini, Anda akan mendapatkan prediksi label atau komponen yang sesuai untuk data pengujian menggunakan model Gaussian Mixture Model yang telah dilatih dengan data pelatihan.