Penambangan Data Genap 2023/2024

Assignment 04

Worksheet Association Rule

Aturan asosiasi (*association rule*) adalah sebuah teknik dalam analisis data yang digunakan untuk mengidentifikasi pola hubungan antara item-item dalam kumpulan data. Teknik ini biasanya diterapkan pada data transaksional, seperti data pembelian pelanggan di toko online atau data penjualan di sebuah toko konvensional. Dalam konsepnya, aturan asosiasi mencoba untuk menemukan korelasi antara item-item tertentu dalam dataset, seperti produk yang sering dibeli bersama atau item yang sering muncul bersama dalam transaksi.

Salah satu metode yang paling umum digunakan untuk menemukan aturan asosiasi adalah Algoritma Apriori. Prinsip apriori adalah sebagai berikut, jika suatu item itu sering muncul atau *frequent* maka semua subsetnya pun harus *frequent* sebagai contoh, $\forall A, BA \subseteq B \Leftrightarrow supp(A) \geq supp(B)$ Ide utama Apriori: pertama, tentukan frequent 1-itemset, lalu frequent 2-itemset, dan seterusnya. hanya satu pemindaian per panjang k melalui database transaksi.

Algoritma ini bekerja dengan cara mengidentifikasi item-item yang sering muncul bersama dalam transaksi, lalu membangun aturan asosiasi berdasarkan frekuensi kemunculan item-item tersebut. Sebagai contoh, algoritma ini dapat menghasilkan aturan seperti "Jika pelanggan membeli X dan Y, maka mereka juga cenderung membeli Z."

Aturan asosiasi memiliki berbagai aplikasi yang luas dalam berbagai bidang, termasuk e-commerce, manajemen inventaris, pemasaran, dan keamanan data. Dalam e-commerce, aturan asosiasi digunakan untuk menyusun rekomendasi produk kepada pelanggan berdasarkan pola pembelian mereka sebelumnya. Di bidang manajemen inventaris, aturan ini membantu dalam merencanakan stok barang dan mengoptimalkan rantai pasokan dengan memahami hubungan antara produk-produk yang berbeda. Selain itu, aturan asosiasi juga digunakan dalam pemasaran untuk merancang strategi promosi yang lebih efektif berdasarkan pola pembelian pelanggan.

Dengan demikian, aturan asosiasi merupakan alat yang kuat dalam mengungkap wawasan berharga dari data dan membantu organisasi membuat keputusan yang lebih baik berdasarkan pemahaman yang lebih dalam tentang pola-pola dalam data mereka.

Soal:
Diberikan suatu data transaksi penjualan yang berhasil dilakukan pada suatu toko elektronik sebagai berikut:

Transaction 1: {Laptop, Mouse, Keyboard, Headphones}
Transaction 2: {Laptop, Mouse, Keyboard}
Transaction 3: {Mouse, Keyboard, Headphones}
Transaction 4: {Laptop, Keyboard, Headphones}
Transaction 5: {Laptop, Mouse, Keyboard, Webcam}

Pertanyaan

1. Dengan minimum support = 0.6, implementasikan algoritma secara naif untuk mendapatkan frequent itemset dengan memanfaatkan data transaksi diatas! Jelaskan setiap langkah yang perlu dilakukan.

Jawaban:
Min support 0.6:
K=1
F1 = {Laptop}, {Mouse}, {Keyboard}, {Headphones}
C2 = {{Laptop, Mouse}, {Laptop, Keyboard}, ~~{Laptop, Headphones}~~, {Mouse, Keyboard}, ~~{Mouse, Headphones}~~, {Keyboard, Headphones}}

K=2
F2 = {Laptop, Mouse}, {Laptop, Keyboard}, {Mouse, Keyboard}, {Keyboard, Headphones}
C3 = {{Laptop, Mouse, Keyboard}, ~~{Laptop, Keyboard, Headphones}~~, ~~{Mouse, Keyboard, Headphones}~~}
Catatan: {Laptop, Keyboard, Headphones} dan {Mouse, Keyboard, Headphones} dihapus karena perhitungan nilai minimum support yang kurang dari 0.6.

K=3
F3 = {Laptop, Mouse, Keyboard}
C4 = {}
Catatan: C4 tidak dapat dibentuk karena F3 hanya ada maksimal 3 elemen.

2. Dengan minimum support = 0.6, jalankan algoritma Apriori dengan menuliskan F<K> sebagai *frequent itemsets* pada suatu K dan C<K+1> sebagai kandidat untuk *frequent itemsets* selanjutnya dengan K adalah jumlah item dalam himpunan berdasarkan data transaksi diatas!
   Hint: Anda dapat memanfaatkan Apriori Principle, lihat referensi pada slide berikut ini

Jawaban:
Min support 0.6:
K=1
F1 = {Laptop}, {Mouse}, {Keyboard}, {Headphones}
C2 = {{Laptop, Mouse}, {Laptop, Keyboard}, ~~{Laptop, Headphones}~~, {Mouse, Keyboard}, ~~{Mouse, Headphones}~~, {Keyboard, Headphones}}

K=2
F2 = {Laptop, Mouse}, {Laptop, Keyboard}, {Mouse, Keyboard}, {Keyboard, Headphones}
C3 = {{Laptop, Mouse, Keyboard}, ~~{Laptop, Keyboard, Headphones}~~, ~~{Mouse, Keyboard, Headphones}~~}
Catatan: {Laptop, Keyboard, Headphones} dan {Mouse, Keyboard, Headphones} dihapus karena {Laptop, Headphones} dan {Mouse, Headphones} tidak ada pada F2 (*downward closure properties*).

K=3
F3 = {Laptop, Mouse, Keyboard}
C4 = {}
Catatan: C4 tidak dapat dibentuk karena F3 hanya ada maksimal 3 elemen.

3. Dengan minimum confidence sebesar 0.5, temukan semua aturan asosiasi dari frequent sets yang telah ditemukan sebelumnya!

Min conf = 0.5
Laptop ⇒ Mouse : 0.8 / 0.8 = 1
Laptop, Mouse ⇒ Keyboard : 0.6 / 0.8 = 0.75
dst

4. Untuk data transaksi yang lebih besar, pengerjaan dengan tangan akan sangat tidak efisien sehingga diperlukan untuk dapat mengimplementasikan algoritma untuk menemukan frequent itemsets, implementasikan algoritma untuk menghitung frequent itemsets tersebut tanpa menggunakan library module (*make it from scratch*)!

   Input : Kumpulan himpunan yang merepresentasikan data transaksi
   Output : Himpunan gabungan dari semua *frequent itemsets*

   Hint: Anda dapat memanfaatkan contoh implementasi pseudocode dibawah ini.
   F1 = {All frequent 1-itemsets}
   Generate C2 = pairs of items from F1
   F2 = {All frequent 2-itemsets selected from C2}
   k = 2
   While FK is not empty do begin
   Generate Ck+1 by joining itemset-pairs in Fk;
   Prune itemsets from Ck+1 that violate downward closure;
   Determine Fk+1 by support counting on Ck+1;
   k = k + 1
   return union(F1, F2, ...,Fn)

Jawaban:

```
def load_data():
    import pandas as pd

    data = {
        'item1': [‘a’, ‘b’, ‘c’]
        'item2': [‘b’, 'c’]
        'item3': [‘c']
```

```
    }
    df = pd.DataFrame(data)
    return df

def generate_candidates(itemsets, k):
    candidates = []
    for i in range(len(itemsets)):
        for j in range(i+1, len(itemsets)):
            if itemsets[i][:k-2] == itemsets[j][:k-2]:
                candidates.append(itemsets[i][:k-2] + [itemsets[i][k-2]] + [itemsets[j][k-2]])
    return candidates

def prune_candidates(candidates, frequent_sets_prev):
    pruned_candidates = []
    for candidate in candidates:
        subsets = [candidate[:2], candidate[:1] + [candidate[2]], candidate[1:]]
        if all(subset in frequent_sets_prev for subset in subsets):
            pruned_candidates.append(candidate)
    return pruned_candidates

def count_support(df, candidate):
    support = 0
    for index, row in df.iterrows():
        if all(item in row.values for item in candidate):
            support += 1
    return support

def apriori(df, min_support):
    itemsets = [[item] for item in df.columns]
    frequent_sets = []
    k = 2

    while itemsets:
        candidates = generate_candidates(itemsets, k)
        candidates = prune_candidates(candidates, frequent_sets)
        frequent_sets_curr = []

        for candidate in candidates:
            support = count_support(df, candidate)
            if support >= min_support:
                frequent_sets_curr.append(candidate)

        frequent_sets.extend(frequent_sets_curr)
        itemsets = frequent_sets_curr
        k += 1

    return frequent_sets
```

```
#FINAL:
df = load_data()
frequent_itemsets = apriori(df, min_support=2)
print("Frequent Itemsets:")
for itemset in frequent_itemsets:
    print(itemset)
```

5. Menurut Anda, apa kelebihan dan kekurangan dari pemanfaatan aturan asosiasi sebagai sistem rekomendasi jika dibandingkan dengan kebanyakan *machine learning based algorithm* untuk sistem rekomendasi (seperti matrix factorization)?

Jawaban:

Kelebihan Aturan Asosiasi:

- **Interpretabilitas Tinggi:** Aturan asosiasi menghasilkan aturan yang mudah dimengerti secara intuitif, seperti "Jika pengguna membeli X, maka mereka cenderung juga membeli Y". Ini memungkinkan pemahaman yang lebih baik tentang pola dan hubungan antar item, sehingga dapat digunakan untuk memberikan rekomendasi yang dapat dijelaskan kepada pengguna.

- **Kemampuan Menangani Data Rendah Dimensi:** Aturan asosiasi dapat bekerja dengan baik bahkan ketika dataset memiliki dimensi rendah atau ukuran sampel kecil. Ini membuatnya cocok untuk digunakan dalam kasus-kasus di mana data tersedia terbatas.

- **Fleksibilitas dalam Pengkodean Variabel:** Aturan asosiasi dapat bekerja dengan baik dengan data kategori atau biner tanpa memerlukan tahap kompleks dalam pembersihan dan transformasi data. Hal ini membuatnya lebih mudah diimplementasikan dalam beberapa situasi.

Kekurangan Aturan Asosiasi:

- **Keterbatasan dalam Memahami Hubungan yang Kompleks:** Aturan asosiasi cenderung tidak mampu menangkap hubungan yang lebih kompleks dan subtan, seperti interaksi non-linear antar fitur. Ini dapat menyebabkan kualitas rekomendasi yang dihasilkan kurang akurat, terutama dalam kasus di mana hubungan antar item tidak sederhana.

- **Rentan terhadap Item yang Populer:** Aturan asosiasi cenderung memberikan rekomendasi berdasarkan popularitas item yang tinggi tanpa memperhitungkan preferensi individu yang lebih halus. Ini dapat menyebabkan pengguna mendapatkan rekomendasi yang kurang beragam dan kurang sesuai dengan preferensi pribadi mereka.

Kelebihan Algoritma Berbasis Pembelajaran Mesin (Matrix Factorization):

- **Kemampuan Menangkap Hubungan yang Kompleks**: Algoritma seperti matrix factorization dapat menangkap hubungan yang lebih kompleks dan non-linear antar item dan pengguna. Ini memungkinkan untuk memberikan rekomendasi yang lebih akurat dan personal kepada pengguna.

- **Personalisasi yang Lebih Baik:** Dengan mengekstraksi representasi laten dari data, algoritma ini dapat memberikan rekomendasi yang lebih personal dan sesuai dengan preferensi individu. Ini memungkinkan untuk memberikan pengalaman pengguna yang lebih baik dan meningkatkan kepuasan pengguna.

Kekurangan Algoritma Berbasis Pembelajaran Mesin (Matrix Factorization):

- **Kesulitan dalam Interpretabilitas:** Representasi laten yang dihasilkan oleh algoritma pembelajaran mesin mungkin sulit untuk diinterpretasikan secara langsung, membuatnya sulit untuk memahami mengapa rekomendasi diberikan. Hal ini dapat menyulitkan untuk menjelaskan rekomendasi kepada pengguna.

- **Ketergantungan pada Data yang Kaya dan Berkualitas Tinggi:** Algoritma pembelajaran mesin sering memerlukan data yang kaya dan berkualitas tinggi untuk memberikan hasil yang optimal. Mereka dapat memiliki kinerja yang buruk jika data yang tersedia terbatas atau tidak terstruktur dengan baik.

Pengumpulan Tugas

Pengumpulan tugas ini diperbolehkan untuk ditulis tangan (pengumpulan di-scan) atau diketik dan dikumpulkan dalam format berikut:

Assignment4_[NPM]_[NamaLengkap].pdf

Contoh:
Assignment4_1906438834_TimothyOrvinEdwardo.pdf

Apabila Anda mencari referensi dari internet, Anda WAJIB menyertakan sumbernya. Dilarang keras menyontek. Plagiarisme tidak ditoleransi dan akan dikenai penalti atau nilai akhir E.

Pengurangan nilai akibat keterlambatan pengumpulan tugas akan ditentukan berdasarkan jumlah menit keterlambatan Anda dalam mengumpulkan. Misalnya, apabila terlambat 1 menit, nilai akhir akan dikurangi 1 poin, apabila terlambat 10 menit, nilai akhir akan dikurangi 10 poin, dan seterusnya.