



Tugas 2

MACHINE LEARNING

Aturan Asosiasi (Algoritma Apriori)

unsia.ac.id



YAYASAN MEMAJUKAN ILMU DAN KEBUDAYAAN

UNIVERSITAS SIBER ASIA

Kampus Menara, Jl. RM. Harsono, Ragunan - Jakarta Selatan. Daerah Khusus Ibukota Jakarta 12550.

Telp. (+6221) 27806189. asiacyberuni@acu.ac.id. www.unsia.ac.id

Mata Kuliah	: Machine Learning
Kelas	: IT-602
Prodi	: PJJ Informatika
Nama Mahasiswa	: Hendro Gunawan
NIM	: 200401072103
Dosen	: Syahid Abdullah, S.Si., M.Kom.

unsia.ac.id



Kata Pengantar

Assalamu'alaikum Warohmatullohi Wabarokatu.

Era industri 4.0 diwarnai dengan penerapan teknologi informasi di berbagai bidang. Perkembangan sistem berbasis komputer yang didukung dengan peningkatan kualitas perangkat lunak dan perangkat keras telah memberikan kontribusi pada munculnya era big data dan saint data. Pengelolaan big data menuntut metode komputasi yang handal, efektif dan efisien dalam menggali pola menarik dari big data untuk mendukung pengambilan keputusan dalam menyelesaikan masalah nyata (riil). Salah satu pendekatan dalam mendapatkan insight dari big data adalah machine learning yang merupakan salah satu cabang dari kecerdasan buatan. Pemahaman mengenai jenis-jenis pembelajaran mesin dan algoritmanya diperlukan untuk dapat menentukan metode/algoritma yang sesuai dalam menyelesaikan permasalahan dengan pendekatan machine learning. Laporan penelitian ini diharapkan dapat menjadi salah satu referensi untuk mengenal jenis-jenis metode dalam machine learning dan algoritma-algoritma machine learning yang umumnya digunakan dalam membuat model prediksi atau klasifikasi, serta mendapatkan pola menarik dari data berukuran besar.

Laporan penelitian ini merupakan Tugas 2 dari mata kuliah Machine Learning yang diberikan oleh dosen saya yaitu Bapak Syahid Abdullah S.Kom., M.Kom yang diadakan di Universitas Siber Asia (UNSA) Jakarta. Dengan beberapa materi yang telah diajarkan kepada kami yaitu dari pertemuan ke satu hingga ke lima belas. Namun kali ini saya akan membahas tentang materi pada pertemuan ke sebelas yaitu Aturan asosiasi (Algoritma Apriori).

Gresik, 20 Maret 2023

Penulis

(Hendro Gunawan)



Daftar Isi

Kata Pengantar.....	i
Daftar Isi.....	ii
BAB I PENDAHULUAN.....	1
BAB II TINJAUAN PUSTAKA.....	3
BAB III HASIL DAN PEMBAHASAN.....	6
BAB IV PENUTUP.....	28
DAFTAR PUSTAKA.....	31



BAB I

PENDAHULUAN

Halo para pembaca yang budiman, ini adalah laporan penelitian tentang machine learning. Di dalam laporan penelitian ini ini kita akan belajar tentang machine learning secara sebagian. Karena laporan penelitian ini hanya menjelaskan tentang Unsupervised Learning khususnya membahas tentang Aturan Asosiasi (Algoritma Apriori) beserta implementasinya dengan menggunakan perhitungan Python. Di sini saya akan membagikan ilmu saya kepada teman-teman bagaimana cara mengolah dataset dengan menggunakan aplikasi Google Collab dengan menggunakan bahasa pemrograman Python.

Pada dasarnya, machine learning digunakan untuk menggantikan manusia dalam mengambil keputusan. Karena machine learning tidak memiliki perasaan, maka keputusan yang akan diambil berdasarkan data yang sudah diolah, kemudian data tersebut akan diekstraksi pengetahuannya dan didapatkan hasil yang sesuai. Jadi memahami machine learning secara umum itu sama seperti tugas manusia yang digantikan oleh machine. Jika kita bisa memprediksikan siapa yang akan terpilih menjadi calon presiden Indonesia di tahun 2024 dengan rumus secara manual, maka machine learning juga dapat melakukan hal tersebut. Jika manusia bisa mengklasifikasikan buah-buahan mana yang mentah dan mana yang matang, maka machine learning juga bisa melakukan hal yang sama. Jika manusia bisa mengelompokkan dokumen berdasarkan kategori-kategori yang ada, maka machine learning pun juga bisa melakukan hal yang sama.

1.1 Latar Belakang Masalah

Secara umum machine learning artinya pembelajaran mesin, pada awalnya manusia belajar untuk hal yang belum ia ketahui. Seperti anak kecil yang bertanya kepada ibunya tentang suatu benda, maka ibunya akan memberi informasi terkait benda tersebut. Saat anak kecil itu bermain sendirian dan menemukan benda yang mirip dengan benda yang sebelumnya, maka anak tersebut akan dengan mudah mengenali benda itu dan menyebutkan nama benda tersebut sesuai dengan informasi yang ia dapatkan sebelumnya. Machine learning meniru cara belajar manusia sehingga yang dihasilkan adalah pengetahuan, bukan sekedar informasi.

Ada suatu perusahaan yang memiliki data transaksi tahun 2010 dan tahun 2011. Data tahun 2010 itu lebih kecil penjualannya daripada data transaksi tahun 2011, sebab berisi informasi bahwa ada kenaikan penjualan dari tahun sebelumnya. Ini adalah sebatas informasi saja, karena informasi tersebut dapat dengan mudah diperoleh berdasarkan data yang dijumlahkan setiap tahun dan dibandingkan dengan tahun tahun sebelumnya.



Atau pada kasus lain, ada data transaksi dari suatu minimarket yaitu data transaksi mie instan, coklat, dan telur. Misalnya kita hanya ingin mengambil informasi, maka kita bisa dengan mudahnya melihat berapa jumlah mie instan, coklat dan telur yang terjual. Namun kita tidak bisa menemukan pola berapa persen orang yang membeli mie instan dan juga coklat, dan berapa persen orang yang membeli ketiganya sekaligus?

1.2 Rumusan Masalah

Dengan melihat latar belakang masalah yang telah dikemukakan maka, beberapa masalah yang dapat penulis rumuskan dan akan dibahas dalam laporan penelitian ini adalah:

1. Apakah aturan asosiasi (algoritma apriori) itu?
2. Apakah cara kerja algoritma apriori itu?
3. Apakah aturan asosiasi dalam penjualan itu?
4. Apakah association rule mining itu?

1.3 Tujuan Dan Manfaat Penelitian

Tujuan dan manfaat penelitian yang ingin dicapai adalah:

1. Mengetahui pengertian dari aturan asosiasi (algoritma apriori)?
2. Mengetahui bagaimana cara kerja algoritma apriori?
3. Mengetahui bagaimana manfaat aturan asosiasi bagi manajer?
4. Mengetahui bagaimana memprediksi kemunculan suatu itemset?

1.4 Metode Penelitian

Tahapan percobaan dalam penelitian ini yaitu pertama melakukan pengumpulan data dengan cara browsing di internet seperti Edge, Google search, bing, dan chat open AI. Untuk file datasetnya peneliti menggunakan dataset dari Kaggle yaitu Grocery_Store_DataSet.csv yang dapat di download secara gratis di halaman website berikut ini <https://www.kaggle.com/datasets/irfanasrullah/groceries>. Di sini peneliti memilih metode Unsupervised Learning dengan menggunakan Algoritma Apriori untuk mengolah dan memproses dataset. Tools yang digunakan dalam penelitian ini yaitu aplikasi Google Collab, Python versi 3.5, Portable Computer (PC) dengan spesifikasi yang digunakan yaitu CPU Intel® Core™ i9-12900KF LGA 1700 yang berjalan pada 3.19 GHz dan GPU NVIDIA GeForce GTX 1650 OC Edition 4 GB DUAL, dengan RAM terinstal 24 GB, sistem operasi 64-bit, dengan spesifikasi windows 11 Pro Insider Preview. Versi 22H2, Build OS 23493.1000.

1.5 Sistematika Penulisan

Dalam penyusunan laporan penelitian ini terdiri dari hal-hal yang saling berkaitan antara bab I sampai dengan bab IV yang memuat beberapa isi sebagai berikut:

BAB I Pendahuluan



Membahas tentang latar belakang masalah, rumusan masalah, tujuan dan manfaat penelitian, metode penelitian, dan sistematika penulisan.

BAB II Tinjauan Pustaka

Membahas tinjauan tentang aturan asosiasi (algoritma apriori) beserta pengertiannya, membahas tinjauan tentang cara kerja algoritma apriori beserta pengertiannya, membahas tinjauan tentang aturan asosiasi dalam penjualan beserta pengertiannya, dan membahas tinjauan tentang aturan asosiasi dalam penjualan beserta penjelasannya.

BAB III Hasil dan Pembahasan

Membahas tentang apakah Algoritma Apriori Itu?, apakah Asosiasi (Algoritma Apriori) Itu?, apakah Cara Kerja Algoritma Apriori Itu?, apakah Aturan Asosiasi dalam Penjualan Itu?, apakah Association Rule Mining Itu?

BAB IV Penutup

Membahas tentang kesimpulan, saran, ucapan terima kasih dan daftar pustaka.

BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Tentang Aturan Asosiasi (Algoritma Apriori)

2.1.1 Pengertian Aturan Asosiasi (Algoritma Apriori)

Algoritma Apriori adalah algoritma yang digunakan untuk menghitung aturan asosiasi antar objek. Aturan asosiasi menjelaskan bagaimana dua atau lebih objek terkait satu sama lain. Dengan kata lain, algoritma apriori adalah algoritma berbasis aturan asosiasi yang menganalisis apakah orang yang membeli produk A juga membeli produk B. Algoritma ini dikemukakan oleh ilmuwan R. Agrawal dan Srikant pada tahun 1994. Algoritma ini kebanyakan digunakan untuk analisis keranjang belanja dan membantu menemukan produk yang dapat dibeli bersama. Apriori juga dapat digunakan di bidang perawatan kesehatan untuk menemukan reaksi obat untuk pasien. Pada data mining, algoritma Apriori banyak digunakan untuk menemukan data yang paling sering muncul dalam sebuah database. Item data transaksi pada database membentuk itemset. Itemset yang paling sering muncul dipilih oleh algoritma Apriori agar dapat digunakan untuk menentukan aturan asosiasi yang menyoroti tren umum dalam database.

2.2 Tinjauan Tentang Cara Kerja Algoritma Apriori

2.2.1 Pengertian Cara Kerja Algoritma Apriori

Algoritma Apriori bekerja untuk menemukan aturan asosiasi yang relevan dalam dataset transaksi. Aturan asosiasi adalah aturan yang menunjukkan hubungan antara satu set item dengan item lainnya dalam transaksi. Misalnya, jika seseorang membeli roti, maka kemungkinan besar dia juga akan membeli mentega. Algoritma Apriori membantu



mengidentifikasi asosiasi semacam itu berdasarkan dukungan (support) dan kepercayaan (confidence) dari kombinasi item.

Berikut adalah langkah-langkah utama cara kerja algoritma Apriori:

1. Hitung Dukungan (Support): Dukungan (support) adalah seberapa sering sebuah itemset (kombinasi beberapa item) muncul dalam dataset transaksi. Dukungan dihitung sebagai jumlah transaksi yang mengandung itemset tersebut dibagi dengan total jumlah transaksi. Itemset dengan dukungan di atas ambang batas (min_support) akan dipertimbangkan sebagai kandidat untuk aturan asosiasi.
2. Hasilkan Itemset Kandidat: Pada awalnya, algoritma Apriori memulai dengan menghasilkan itemset kandidat satu item (singleton) berdasarkan item-item unik yang ada dalam dataset. Selanjutnya, itemset kandidat yang lebih besar dihasilkan dengan menggabungkan itemset kandidat yang lebih kecil berdasarkan aturan yang telah ditentukan.
3. Prune Itemset Kandidat: Setelah menghasilkan itemset kandidat, langkah selanjutnya adalah memangkas atau membuang itemset kandidat yang tidak memenuhi ambang batas dukungan (min_support). Itemset yang tersisa kemudian menjadi itemset yang sering muncul dalam dataset.
4. Hitung Kepercayaan (Confidence): Kepercayaan (confidence) adalah seberapa sering aturan asosiasi tertentu ditemukan benar. Kepercayaan dihitung sebagai dukungan dari gabungan itemset yang ada dibagi dengan dukungan dari itemset yang ada di sebelah kiri aturan asosiasi. Aturan asosiasi dengan kepercayaan di atas ambang batas (min_confidence) akan dipertimbangkan sebagai aturan asosiasi yang relevan.
5. Pilih Aturan Asosiasi: Aturan asosiasi yang relevan akan dipilih berdasarkan ambang batas dukungan (min_support) dan ambang batas kepercayaan (min_confidence). Aturan asosiasi yang memenuhi kedua kriteria ini dianggap cukup kuat dan relevan dalam dataset transaksi.
6. Iterasi Lanjutan: Proses perhitungan dukungan, itemset kandidat, dan kepercayaan berlanjut hingga tidak ada itemset kandidat baru yang memenuhi ambang batas dukungan (min_support) atau tidak ada aturan asosiasi baru yang memenuhi ambang batas kepercayaan (min_confidence).

Dengan cara ini, algoritma Apriori secara iteratif mencari itemset dan aturan asosiasi yang kuat dalam dataset transaksi. Dengan mengidentifikasi asosiasi antara item, algoritma Apriori membantu kita memahami pola pembelian dan preferensi pelanggan dalam data transaksi sehingga dapat digunakan untuk pengambilan keputusan bisnis, pemasaran, dan rekomendasi produk.



2.3 Tinjauan Tentang Aturan Asosiasi dalam Penjualan

2.3.1 Pengertian Aturan Asosiasi dalam Penjualan

Association Rules adalah proses mendeteksi kumpulan atribut-atribut yang muncul bersamaan (co-occur) dalam frekuensi yang sering, dan membentuk sejumlah kaidah dari kumpulan-kumpulan tersebut. Contoh: 90% orang yang berbelanja di suatu supermarket yang membeli roti juga membeli selai, dan 60% dari semua orang yang berbelanja membeli keduanya.

Association rule mining adalah analisa dari kebiasaan belanja konsumen dengan mencari asosiasi dan korelasi antara item-item berbeda yang diletakkan konsumen dalam keranjang belanjanya. Contoh association rule misalnya: “70 % dari orang yang membeli mie, juice dan saus akan membeli juga roti tawar”. Dengan kemajuan teknologi, data penjualan dapat disimpan dalam jumlah besar yang disebut dengan “basket data”.

Aturan asosiasi yang didefinisikan pada basket data tersebut, dapat digunakan untuk menganalisa data dalam rangka:

1. keperluan desain katalog promosi,
2. proses pembuatan keputusan bisnis,
3. segmentasi konsumen dan
4. target pemasaran.

2.4 Tinjauan Tentang Association Rule Mining

2.4.1 Pengertian Association Rule Mining

Association Rule Mining, atau penambangan aturan asosiasi, adalah teknik dalam bidang analisis data dan data mining yang bertujuan untuk menemukan hubungan atau pola asosiasi antara item atau atribut dalam dataset transaksi. Tujuan utama dari *Association Rule Mining* adalah untuk mengidentifikasi aturan asosiasi yang kuat atau relevan, yang dapat memberikan wawasan dan informasi yang berharga untuk pengambilan keputusan bisnis, pemasaran, dan rekomendasi produk.

Berikut adalah pengertian *Association Rule Mining* menurut beberapa ahli:

1. Jiawei Han dan Micheline Kamber dalam buku "*Data Mining: Concepts and Techniques*" mendefinisikan Association Rule Mining sebagai teknik untuk menemukan semua aturan asosiasi menarik atau kuat antara item-item dalam kumpulan transaksi.
2. Rakesh Agrawal, Tomasz Imielinski, dan Arun Swami dalam makalah "*Mining Association Rules between Sets of Items in Large Databases*" menyatakan bahwa Association Rule Mining adalah proses mencari aturan asosiasi di antara set item yang ada dalam database besar.



3. Mohammed J. Zaki dan Ching-Jui Hsiao dalam makalah "*Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure*" mendefinisikan Association Rule Mining sebagai teknik untuk menemukan pola asosiasi yang menarik antara item dalam data transaksi.

Secara umum, Association Rule Mining berfokus pada menemukan item atau atribut yang sering muncul bersama dalam transaksi atau data, dan menghasilkan aturan asosiasi yang kuat berdasarkan ukuran dukungan (*support*) dan kepercayaan (*confidence*) dari kombinasi item. Aturan asosiasi yang ditemukan dapat digunakan untuk mendapatkan wawasan tentang kebiasaan pembelian, preferensi pelanggan, dan pola lainnya dalam data, sehingga dapat mendukung pengambilan keputusan yang lebih baik dan strategi bisnis yang lebih efektif.

BAB III

HASIL DAN PEMBAHASAN

3.1 Apakah Asosiasi (Algoritma Apriori) Itu?

ISTILAH DASAR DALAM APRIORI

- Pengetahuan apakah yang hendak dihasilkan dalam aturan apriori?
- Pengetahuan untuk mengetahui item-item belanja yang sering dibeli secara bersamaan dalam suatu waktu.
- =====
- **Item** = barang yang dibeli
- **Himpunan Item (I)** adalah himpunan dari semua jenis item yang akan di bahas.
- Contoh :
{Asparagus, Beans, ..., Tomatoes}

Gambar 1. Ilustrasi metode apriori

Aturan asosiasi (*association rules*) sering disebut sebagai analisis afinitas (*affinity analysis*) atau analisis pertalian. Aturan asosiasi merupakan studi mengenai ‘apa bersama apa’ atau “sesuatu memiliki pertalian dengan sesuatu”. Misalnya saat seseorang belanja di supermarket, jika seseorang membeli susu bayi biasanya seseorang juga membeli diapers, dapat dikatakan susu bayi bersama diapers atau susu bayi memiliki pertalian dengan diapers. Karena studi ini diawali pada database transaksi pelanggan, maka studi ini juga disebut “*market basket analysis*”.

Algoritma apriori adalah sebuah algoritma klasik pada data mining. Algoritma ini menggunakan frekuensi itemset untuk menghasilkan aturan asosiasi. Hal ini berdasarkan konsep bahwa subset dari frekuensi itemset. Lalu, apa itu frekuensi item set?



Frekuensi item set merupakan nilai item set minimum yang muncul di himpunan seluruh transaksi (transaksi 1, transaksi 2, transaksi 3, dan seterusnya) atau disebut juga minimum support. Selain frekuensi item set, terdapat beberapa terminologi atau istilah lain yang perlu kita pahami pada algoritma apriori seperti:

- *K*-itemset merupakan item set yang terdiri dari *K* buah item yang terdapat di dalam himpunan. Misalnya {roti, susu, kopi} merupakan 3-itemset; {madu, es krim} merupakan 2-item set; {keju} merupakan 1-item set.
- *Support* merupakan sebuah item set yang memiliki support. Misalnya 10%, jika 10% dari dalam *database* berisi item-item tersebut.
- *Minimum support* merupakan nilai minimum yang telah ditentukan sebelum algoritma apriori dimulai.

3.2 Cara Kerja Algoritma Apriori

1. Menentukan nilai *minimum support*
2. Iterasi 1: menghitung item-item dari *support* (transaksi yang memuat seluruh item) dengan men-*scan* database untuk 1-itemset, setelah 1-itemset didapatkan, dari 1-itemset apakah di atas *minimum support*, apabila telah memenuhi *minimum support*, 1-itemset tersebut akan menjadi pola frekuensi tinggi.
3. Iterasi 2: untuk mendapatkan 2-itemset, harus dilakukan kombinasi dari *k*-itemset sebelumnya, kemudian scan database lagi untuk menghitung item-item yang memuat *support*. Itemset yang memenuhi *minimum support* akan dipilih sebagai pola frekuensi tinggi dari kandidat.
4. Tetapkan nilai *k*-itemset dari *support* yang telah memenuhi *minimum support* dari *k*-itemset.
5. Lakukan proses untuk iterasi selanjutnya hingga tidak ada lagi *k*-itemset yang memenuhi *minimum support*.

3.3 Aturan Asosiasi dalam Penjualan

Tersedianya database market basket/transaksi pembelian pada pusat-pusat penjualan (apapun) mendorong pengembangan teknik-teknik yang secara otomatis menemukan asosiasi produk atau item-item yang tersimpan dalam database tersebut. Market basket database tersebut mengandung record dalam jumlah yang amat besar. Tiap record mencatat semua item yang dibeli oleh pelanggan dalam transaksi tunggal. Pengambilan keputusan memerlukan data mengenai pola transaksi.

Manfaat bagi manajer yaitu diantaranya:

- Dapat ditentukannya penempatan barang-barang dalam layout dengan lebih tepat.

Misal susu diletakkan berdekatan dengan diapers.



- Promosi produk.
- Segmentasi pembeli.
- Pembuatan katalog.
- Melihat pola kecenderungan pola belanja pelanggan.

Aturan asosiasi juga dapat diterapkan dalam bentuk sistem rekomendasi, misal:

- Sistem rekomendasi pembelian buku atau dvd online (www.amazon.com).
- Sistem rekomendasi pencarian artikel dalam search engine.
- Sistem rekomendasi peminjaman atau pengadaan buku pada perpustakaan.

Hal penting yang dilakukan oleh aturan asosiasi adalah:

- Penyajian informasi transaksi ke dalam bentuk “*if-then*” atau “jika maka”.
- Aturan ini dihitung dari sifat probabilistik yang dimiliki oleh data yang ada.

3.4 Association Rule Mining

Jika diberikan sekumpulan data transaksi, tentukan suatu aturan yang akan memprediksi kemunculan suatu item berdasar kemunculan item yang lain dalam transaksi tersebut.

Tabel 1. Market-Basket transactions

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Contoh dari association Rules

$\{\text{Diaper}\} \rightarrow \{\text{Beers}\},$

$\{\text{Milk, Bread}\} \rightarrow \{\text{Eggs, Coke}\},$

$\{\text{Beer, Bread}\} \rightarrow \{\text{Milk}\},$

Pengertian dari pernyataan tersebut adalah *co-occurrence*, bukan sebab akibat (kausalitas)!

3.5 Definisi Frequent Itemset

Item set merupakan sekumpulan satu atau lebih item

Misal: $\{\text{Milk, Bread, Diaper}\}$

- **K-itemset**

Suatu item set yang terdiri dari k item

- **Support count (σ)**

Frekuensi kemunculan suatu itemset



Misal: $\sigma(\{\text{Milk, Bread, Diaper}\}) = 2$

- Support

Pecahan transaksi yang terdiri dari suatu itemset

Misal: $s(\{\text{Milk, Bread, Diaper}\}) = 2/5$

- Frequent Itemset

Suatu itemset yang memiliki nilai support lebih tinggi atau sama dengan batas minimum support (*minsup*)

- Association Rule

Persamaan dalam bentuk $X \rightarrow Y$, di mana X dan Y merupakan itemset

Misal: $\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

- Rule Evaluation Metrics

Terdiri dari

- Support (s)
 - Pecahan transaksi yang terdiri dari kedua item X dan Y
- Confidence (c)
 - Ukuran seberapa sering item dalam y muncul dalam transaksi yang terdiri dari X.

Tabel 2. Market-Basket transactions

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Misal: $\{\text{Milk, Diaper}\} \rightarrow \text{Beer}$

$$s = \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67 = 0.7$$

3.6 Tugas Association Rule Mining

Jika diberikan sekumpulan transaksi T, tujuan dari association rule mining adalah menemukan semua aturan yang memiliki:

- Support \geq batas minsup
- Confidence \geq batas minconf



- Pendekatan Brute-force:
 - Daftar semua association rules yang mungkin
 - Hitung support dan confidence untuk masing-masing
 - Pangkas (Prune) aturan yang tidak memenuhi batas minsup dan minconf
 - ➔ Computationally prohibitive!
- Mining Association Rules

Tabel 3. Market-Basket transactions

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Contoh dari aturan:

$\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\} (s = 0.4, c = 0.67)$

$\{\text{Milk, Beer}\} \rightarrow \{\text{Diaper}\} (s = 0.4, c = 1.0)$

$\{\text{Diaper, Beer}\} \rightarrow \{\text{Milk}\} (s = 0.4, c = 0.67)$

$\{\text{Diaper}\} \rightarrow \{\text{Milk, Beer}\} (s = 0.4, c = 0.5)$

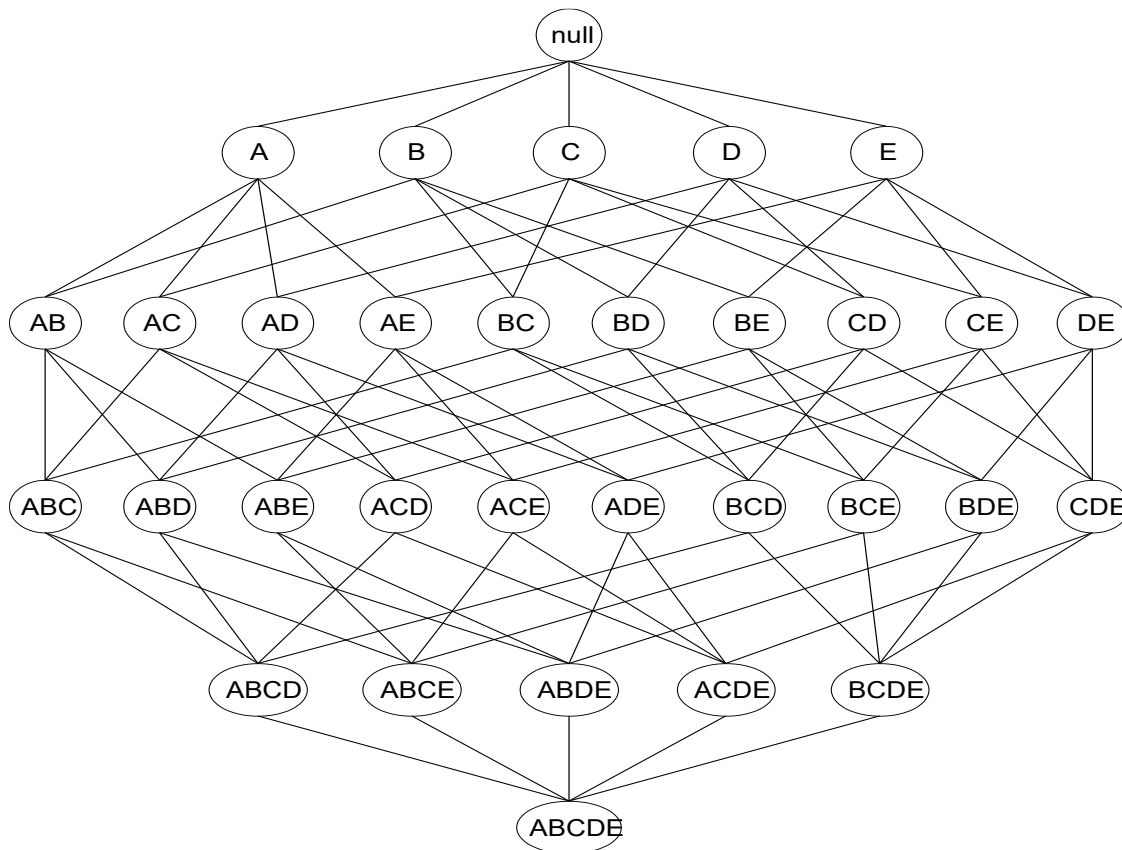
$\{\text{Milk}\} \rightarrow \{\text{Diaper, Beer}\} (s = 0.4, c = 0.5)$

Pengamatan:

- Semua aturan di atas merupakan partisi biner dari itemset yang sama:
 $\{\text{Milk, Diaper, Beer}\}$
- Aturan yang dibentuk dari itemset yang sama memiliki *support* yang identik tetapi dapat memiliki nilai *confidence* yang berbeda.
- Dengan demikian kita dapat memisahkan persyaratan *support* dan *confidence*.
- Dua tahap pendekatan dalam proses mendapatkan aturan asosiasi, yaitu:
 - Frequent Itemset Generation
 - Membentuk semua itemset yang memiliki $support \geq minsup$
 - Rule Generation
 - Membentuk *high confidence rule* dari masing-masing *frequent itemset*, di mana setiap *rule* merupakan *binary partitioning* dari suatu *frequent itemset*.

- Pembentukan *frequent itemset* masih merupakan proses komputasi yang mahal.

3.7 Frequent Itemset Generation

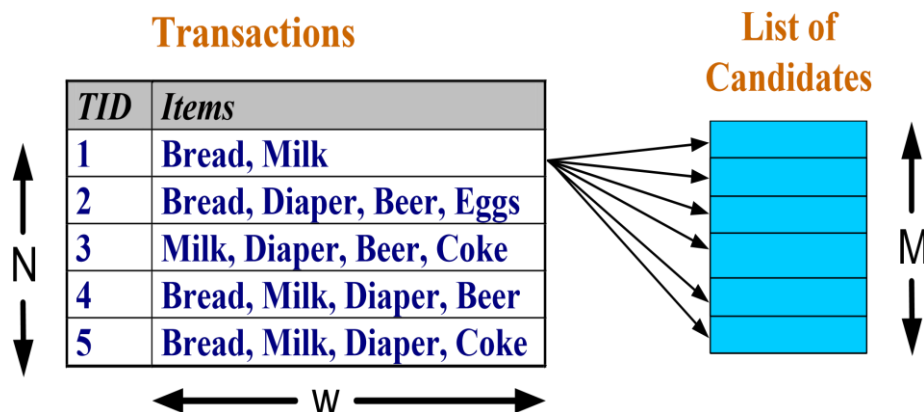


Gambar 2. Frequent Itemset Generation

Jika diberikan d item, maka akan terdapat 2^d kemungkinan kandidat itemset.

- Brute-force approach:
 - Masing-masing itemset dalam kisi-kisi tersebut merupakan candidate frequent itemset.
 - Hitung support untuk masing-masing candidate dengan mencarinya dalam database.
 - Cocokkan masing-masing transaksi dengan setiap kandidat yang ada.
 - Kompleksitas adalah ekuivalen dengan $O(NMw) \Rightarrow$ Expensive karena $M = 2^d$!!!

Tabel 4. Market-Basket transaction and List of candidate



3.8 Strategi Pembentukan Frequent Itemset

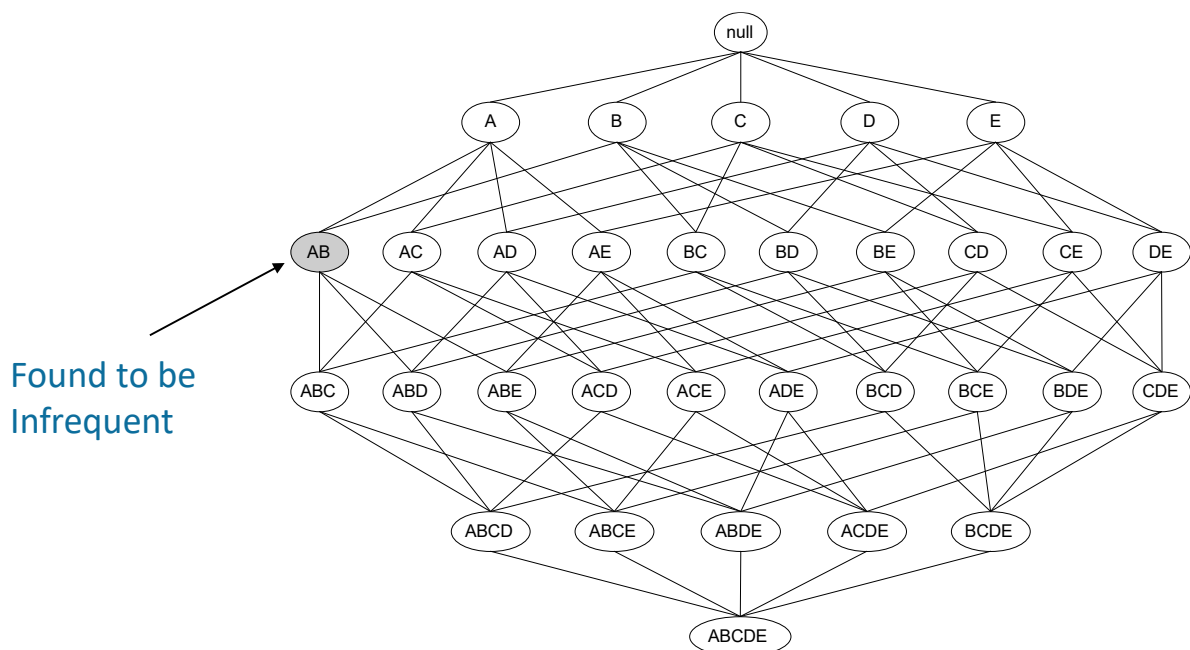
- Kurangi jumlah kandidat (M)
 - Complete search: $M = 2^d$
 - Gunakan teknik pemangkasan (pruning) untuk mengurangi M.
- Kurangi jumlah transaksi (N)
 - Kurangi ukuran N saat ukuran dari itemset meningkat.
- Kurangi jumlah proses pencocokan (NM)
 - Gunakan struktur data yang efisien untuk menyimpan kandidat ataupun transaksi.
 - Tidak diperlukan untuk mencocokkan setiap candidate dengan tiap-tiap transaksi yang ada.

3.9 Mengurangi Jumlah Kandidat

- Prinsip Apriori:
 - Jika suatu itemset seringkali muncul, maka semua himpunan bagiannya semestinya juga sering muncul.
- Prinsip apriori memiliki kecenderungan sifat ukuran support sebagai berikut:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$
 - Support dari suatu itemset tidak pernah melampaui support dari subsetnya.
 - Hal ini dikenal sebagai sifat anti-monotone dari support.

3.10 Gambar Prinsip Apriori



Gambar 3. Gambaran prinsip apriori



Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Minimum Support = 3

Itemset	Count
{Bread, Milk}	3
{Bread, Beer}	2
{Bread, Diaper}	3
{Milk, Beer}	2
{Milk, Diaper}	3
{Beer, Diaper}	3

Pairs (2-Itemsets)

(Tidak diperlukan membentuk kandidat yang melibatkan Coke ataupun Eggs)

Itemset	Count
{Bread, Milk, Diaper}	3

3.11 Contoh Perhitungan Manual

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

Scan D

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

C_3

itemset
{2 3 5}

L_3

itemset	sup
{2 3 5}	2



3.12 Algoritma Apriori

- Langkah penggabungan: C_k dibangun dengan menggabungkan L_{k-1} dengan dirinya sendiri.
- Langkah pemangkasan: Setiap (k-1) itemset yang tidak sering muncul (not frequent) tidak dapat menjadi subset dari frequent k-itemset
- Pseudo-Code:

C_k : Candidate itemset of size k

L_k : Frequent itemset of size k

L_1 : {frequent items};

for ($k = 1$; $L_k \neq \emptyset$; $k++$) do begin

C_{k+1} = candidates generated from L_k ;

for each transaction t in database do

increment the count of all candidates in C_{k+1} that are contained in t

L_{k+1} = candidates in C_{k+1} with min_support

end

Return $\cup_k L_k$;

3.13 Bagaimana Membentuk Kandidat?

- Andaikan item dalam L_{k-1} terdaftar dalam suatu transaksi.
- Langkah 1: self-joining L_{k-1}
insert into C_k
select $p.item_1, p.item_2, \dots, p.item_{k-1}, q.item_{k-1}$
from $L_{k-1} p, L_{k-1} q$
where $p.item_1 = q.item_1, \dots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$
- Langkah 2: pruning
for all *itemsets* c in C_k do
 for all (k-1)-subsets s of c do
 if (s is not in L_{k-1}) then delete c from C_k

3.14 Contoh Pembentukan Kandidat

- $L_3 = \{ abc, acd, ace, bcd \}$
- Self-joining: $L_3 * L_3$
 - $abcd$ dari abc dan abd
 - $acde$ dari acd dan ace
- Pruning:



- $acde$ dihapus karena ade tidak dalam L_3
- $C_4 = \{abcd\}$

3.15 Perhitungan Python

Untuk lebih memahami cara apriori, kita akan menggunakan kasus dengan Python. Dataset yang akan digunakan adalah dataset Grocery_Store_Dataset.csv dari *kaggle*. Data tersebut dapat didownload di situs <https://www.kaggle.com/datasets/irfanasrullah/groceries>.

3.15.1 Import Library

```
!pip install apyori
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl
  size=5956
  sha256=7a65336ac446038b4191b4ef2652762405ebc79be6baf62321c8db3d5b0f753
  3
  Stored in directory:
  /root/.cache/pip/wheels/c4/1a/79/20f55c470a50bb3702a8cb7c94d8ada155735
  38c7f4bae2d
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

Keterangan kode:

- `!pip install apyori`, berfungsi untuk menginstall paket Python bernama “apyori” menggunakan pip (Python Package Installer). Paket “apyori” adalah implementasi dari algoritma apriori yang digunakan untuk menemukan asosiasi dalam dataset yang berisi kumpulan itemset.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Keterangan kode:

- `import numpy as np`, berfungsi untuk memudahkan operasi perhitungan type data numerik seperti penjumlahan, perkalian, pengurangan, pemangkatan dan operasi aritmatika lainnya.
- `import matplotlib.pyplot as plt`, memanggil library matplotlib untuk membuat chart atau grafik.
- `import pandas as pd`, bisa digunakan untuk mengubah dimensi data, membuat tabel, memeriksa data, membaca data dan lain sebagainya.

3.15.2 Pra Proses Data

```
# Mounted At/Content/Drive
```



```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

Keterangan kode:

- `from google.colab import drive`, digunakan untuk menghubungkan dan mengakses Google Drive dari dalam notebook Colab. Ini memungkinkan Anda untuk menyimpan dan membaca file dari Google Drive secara langsung dalam lingkungan Colab.
- `drive.mount('/content/drive')`, digunakan untuk mengakses file dan folder di Google Drive dengan path `'/content/drive/'`.

```
dataset =
pd.read_csv('/content/drive/MyDrive/KuliahUNSIA/Semester5/MachineLearning/Pertemuan11/Grocery_Store_DataSet.csv', header = None)
transactions = []
for i in range(0, 7501):
    transactions.append([str(dataset.values[i,j]) for j in range(0, 20)])
```

Keterangan kode:

- `dataset = pd.read_csv('/content/drive/MyDrive/KuliahUNSIA/Semester5/MachineLearning/Pertemuan11/Grocery_Store_DataSet.csv', header = None)`, berfungsi untuk membaca file `'data.csv'` yang ada di Google Drive.
- `transactions = []`, berfungsi untuk mendefinisikan sebuah variabel `'transaction'` sebagai daftar kosong dalam bahasa Python. "Daftar" dalam Python adalah type data yang digunakan untuk menyimpan sekumpulan elemen atau nilai, dan `'[]'` adalah representasi dari daftar kosong.
- `for i in range(0, 7501):` Ini adalah loop for yang akan berjalan dari 0 hingga 7500 (7501 tidak akan diikutsertakan). Angka 7501 mungkin merupakan jumlah baris dalam dataset yang diinginkan untuk diproses.
- `transactions.append(...)`: Ini adalah fungsi untuk menambahkan data ke dalam daftar `transactions`. Baris kode ini akan dijalankan pada setiap iterasi loop, sehingga data akan ditambahkan ke daftar `transactions` dalam setiap iterasi.
- `[str(dataset.values[i,j]) for j in range(0, 20)]`: Bagian ini adalah list comprehension, yang berarti akan menghasilkan daftar baru berdasarkan ekspresi yang diberikan. Di sini, kita sedang membuat daftar dengan melakukan iterasi melalui kolom-kolom (diasumsikan 20 kolom) dalam dataset pada baris ke-i. Fungsi `str()` digunakan



untuk mengonversi nilai menjadi string, karena item dalam daftar transactions biasanya perlu berupa string (seperti dalam implementasi algoritma Apriori).

Jadi, tujuan dari kode tersebut adalah untuk mengambil nilai dari dataset (DataFrame) pada setiap baris dan mengubahnya menjadi bentuk daftar yang kemudian ditambahkan ke dalam variabel 'transaction'. Hal ini akan berguna ketika Anda ingin menggunakan dataset ini sebagai input untuk analisa asosiasi data menggunakan algoritma seperti Apriori. Dengan menggunakan variabel 'transaction' yang dihasilkan dari kode ini, Anda dapat memproses dataset transaksi tersebut lebih lanjut untuk menemukan pola asosiasi atau itemset yang sering muncul bersama dalam transaksi.

```
print(dataset)
```

```
0          1          2          3  \
0  Item(s)          Item 1          Item 2
Item 3
1          4          citrus fruit  semi-finished bread
margarine
2          3          tropical fruit          yogurt
coffee
3          1          whole milk          NaN
NaN
4          4          pip fruit          yogurt          cream
cheese
...          ...          ...          ...
...
9831          17          sausage          chicken
beef
9832          1          cooking chocolate          NaN
NaN
9833          10          chicken          citrus fruit  other
vegetables
9834          4  semi-finished bread          bottled water
soda
9835          5          chicken          tropical fruit  other
vegetables

          4          5          6          7
\
0          Item 4          Item 5          Item 6          Item 7
1          ready soups          NaN          NaN          NaN
2          NaN          NaN          NaN          NaN
3          NaN          NaN          NaN          NaN
4          meat spreads          NaN          NaN          NaN
...          ...          ...          ...
9831  hamburger meat  citrus fruit          grapes  root vegetables
9832          NaN          NaN          NaN          NaN
9833          butter          yogurt  frozen dessert  domestic eggs
9834  bottled beer          NaN          NaN          NaN
9835          vinegar  shopping bags          NaN          NaN

          8          9  ...          23          24          25          26
27  \
0          Item 8  Item 9  ...  Item 23  Item 24  Item 25  Item 26
Item 27
```



```

1      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
2      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
3      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
4      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
...      ...      ...      ...      ...      ...      ...      ...
...
9831  whole milk  butter  ...      NaN      NaN      NaN      NaN
NaN
9832      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
9833  rolls/buns      rum      ...      NaN      NaN      NaN      NaN
NaN
9834      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN
9835      NaN      NaN      ...      NaN      NaN      NaN      NaN
NaN

```

```

      28      29      30      31      32
0  Item 28  Item 29  Item 30  Item 31  Item 32
1      NaN      NaN      NaN      NaN      NaN
2      NaN      NaN      NaN      NaN      NaN
3      NaN      NaN      NaN      NaN      NaN
4      NaN      NaN      NaN      NaN      NaN
...      ...      ...      ...      ...
9831      NaN      NaN      NaN      NaN      NaN
9832      NaN      NaN      NaN      NaN      NaN
9833      NaN      NaN      NaN      NaN      NaN
9834      NaN      NaN      NaN      NaN      NaN
9835      NaN      NaN      NaN      NaN      NaN

```

[9836 rows x 33 columns]

Keterangan kode:

- Perintah `'print (dataset)'` digunakan untuk mencetak isi dari variabel `'dataset'` ke output konsol atau layer. Saat dijalankan, perintah ini akan menampilkan isi dari objek yang disimpan dalam variabel `'dataset'` ke layar.

3.15.3 Menampilkan Sebanyak 5 Data Teratas

```

df= dataset =
pd.read_csv('/content/drive/MyDrive/KuliahUNSI/SEMESTER5/MachineLearning/Pe
rtemuan11/Grocery_Store_DataSet.csv', header = None)
df.head(5)

```

```

0  1  2  3  4  5  6  7  8  9  ..  2  2  2  2  2  2  2  3  3  3
   .  3  4  5  6  7  8  9  .  .  .  .  .  .  .  .  .  .  .  .
0  l  l  l  l  l  l  l  l  l  l  .  l  l  l  l  l  l  l  l  l  l
e  e  e  e  e  e  e  e  e  e  .  e  e  e  e  e  e  e  e  e  e
m  m  m  m  m  m  m  m  m  m  .  m  m  m  m  m  m  m  m  m  m
(s 1  2  3  4  5  6  7  8  9  .  2  2  2  2  2  2  2  3  3  3
)  )  )  )  )  )  )  )  )  )  )  .  3  4  5  6  7  8  9  0  1  2

```



0	1	2	3	4	5	6	7	8	9	..	2	2	2	2	2	2	2	3	3	3
.	3	4	5	6	7	8	9	0	1	2
1	4	citrus fruit	semi-finish bread	margarine	ready ups	N a N	N a N	N a N	N a N	N a N	.	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N
2	3	tropical fruit	yo gurt	coffee	N a N	N a N	N a N	N a N	N a N	N a N	.	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N
3	1	whole milk	N a N	NaN	N a N	N a N	N a N	N a N	N a N	N a N	.	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N
4	4	pip fruit	yo gurt	cream cheese	meat spread	N a N	N a N	N a N	N a N	N a N	.	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N	N a N

5 rows × 33 columns

Keterangan kode:

- `df=dataset=pd.read_csv('/content/drive/MyDrive/KuliahUNSIA/Semester5/MachineLearning/Pertemuan11/Grocery_Store_DataSet.csv', header = None)`, digunakan untuk mengubah dataset untuk divariabelkan menjadi 'df', sehingga jika ingin melihat isi dari dataset tersebut kita tinggal mengetikkan 'df' atau 'dataset' saja.
- `df.head(5)`, digunakan untuk menampilkan sebanyak 5 data teratas.

3.15.4 Melihat Variabel dalam Dataset

```
df.keys()
```

```
Int64Index([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32],
            dtype='int64')
```

Keterangan kode:



- `df.keys()`, digunakan untuk mendapatkan daftar (*list*) dari semua nama kolom yang ada dalam DataFrame ‘df’ pada library pandas.

3.15.5 Melihat Shape

```
df.shape
```

```
(9836, 33)
```

Keterangan kode:

- `df.shape`, digunakan untuk melihat berapa banyak baris dan kolom yang ada dalam data set tersebut, dengan kata lain yaitu `df.shape` digunakan untuk mendapatkan tuple yang berisi dimensi dari DataFrame ‘df’ pada library pandas. Tuple ini berisi dua nilai: jumlah baris dan jumlah kolom dalam DataFrame.

3.15.6 Training Apriori

```
from apyori import apriori
rules = apriori(transactions = transactions, min_support = 0.003,
min_confidence = 0.2, min_lift = 3, min_length = 2, max_length = 2)
```

Keterangan kode:

- `from apyori import apriori`, digunakan untuk mengimpor fungsi ‘apriori’ dari modul ‘apyori’. Modul ‘apyori’ adalah pustaka Python yang menyediakan implementasi algoritma Apriori untuk analisa asosiasi data.
- Pada kode ‘`rules = apriori(transactions=transactions, min_support=0.003)`’, fungsi ‘apriori’ dari modul ‘apyori’ digunakan untuk menerapkan algoritma Apriori pada dataset transaksi atau itemset yang disimpan dalam variabel ‘transactions’. Tujuan dari pemanggilan fungsi ini adalah untuk menemukan aturan asosiasi yang relevan berdasarkan kriteria support (dukungan) yang ditentukan.

Mari kita bahas argumen-argumen yang digunakan dalam pemanggilan fungsi ‘apriori’ ini:

- ‘transactions’: Ini adalah argumen yang mengharuskan Anda memberikan kumpulan data transaksi atau itemset dalam bentuk daftar dari daftar. Daftar ‘transactions’ menyimpan semua transaksi atau itemset yang akan digunakan untuk menemukan aturan asosiasi.
- ‘min_support’: Ini adalah nilai minimum dari dukungan (support) yang diperlukan untuk suatu aturan asosiasi agar dianggap relevan. Dukungan suatu aturan adalah proporsi transaksi yang mengandung semua item dalam aturan tersebut. Nilai ‘min_support’ adalah ambang batas untuk memfilter aturan yang jarang muncul dan hanya mempertahankan aturan yang memiliki dukungan yang lebih tinggi.



- 'min_confidence', 'min_lift', 'min_length', dan 'max_length' digunakan untuk memberikan lebih banyak kriteria dan batasan dalam mencari aturan asosiasi yang relevan dalam dataset transaksi.

3.15.7 Menampilkan Hasil Training

```
results = list(rules)
for rule in results:
    print(rule)
```

```
RelationRecord(items=frozenset({'10', 'fruit/vegetable juice'}),
support=0.0061325156645780565,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'10'}),
items_add=frozenset({'fruit/vegetable juice'}),
confidence=0.25136612021857924, lift=3.46598762455802)])
RelationRecord(items=frozenset({'10', 'margarine'}),
support=0.005465937874950006,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'10'}),
items_add=frozenset({'margarine'}), confidence=0.2240437158469945,
lift=3.734559805707346)])
RelationRecord(items=frozenset({'10', 'root vegetables'}),
support=0.008132249033462205,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'10'}),
items_add=frozenset({'root vegetables'}),
confidence=0.3333333333333333, lift=3.0160836349014875)])
RelationRecord(items=frozenset({'domestic eggs', '11'}),
support=0.0038661511798426876,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'11'}),
items_add=frozenset({'domestic eggs'}),
confidence=0.21804511278195488, lift=3.379248741688933)])
RelationRecord(items=frozenset({'margarine', '11'}),
support=0.0038661511798426876,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'11'}),
items_add=frozenset({'margarine'}), confidence=0.21804511278195488,
lift=3.634569757727652)])
RelationRecord(items=frozenset({'domestic eggs', '12'}),
support=0.0034662045060658577,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'domestic eggs'}), confidence=0.2765957446808511,
lift=4.286662563741868)])
RelationRecord(items=frozenset({'fruit/vegetable juice', '12'}),
support=0.003199573390214638,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'fruit/vegetable juice'}),
confidence=0.25531914893617025, lift=3.5204943679599507)])
RelationRecord(items=frozenset({'12', 'other vegetables'}),
support=0.007598986801759766,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'other vegetables'}),
confidence=0.6063829787234043, lift=3.1564737844581927)])
RelationRecord(items=frozenset({'pip fruit', '12'}),
support=0.003332888948140248,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'pip fruit'}), confidence=0.26595744680851063,
lift=3.4455039870650053)])
RelationRecord(items=frozenset({'12', 'root vegetables'}),
support=0.004399413411545127,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
```




```
items_add=frozenset({'root vegetables'}),
confidence=0.35106382978723405, lift=3.1765136154813542))
RelationRecord(items=frozenset({'tropical fruit', '12'}),
support=0.0041327822956939075,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'tropical fruit'}),
confidence=0.3297872340425532, lift=3.1392563991791773)])
RelationRecord(items=frozenset({'whipped/sour cream', '12'}),
support=0.003332888948140248,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'12'}),
items_add=frozenset({'whipped/sour cream'}),
confidence=0.26595744680851063, lift=3.6404138841435008)])
RelationRecord(items=frozenset({'13', 'whipped/sour cream'}),
support=0.0030662578322890282,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'13'}),
items_add=frozenset({'whipped/sour cream'}),
confidence=0.3770491803278689, lift=5.161032667225081)])
RelationRecord(items=frozenset({'13', 'yogurt'}),
support=0.0037328356219170776,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'13'}),
items_add=frozenset({'yogurt'}), confidence=0.45901639344262296,
lift=3.3427980264204997)])
RelationRecord(items=frozenset({'other vegetables', '14'}),
support=0.004932675643247567,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'14'}),
items_add=frozenset({'other vegetables'}),
confidence=0.6491228070175439, lift=3.378952238333516)])
RelationRecord(items=frozenset({'14', 'root vegetables'}),
support=0.0034662045060658577,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'14'}),
items_add=frozenset({'root vegetables'}),
confidence=0.45614035087719296, lift=4.127272342496773)])
RelationRecord(items=frozenset({'yogurt', '14'}),
support=0.003332888948140248,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'14'}),
items_add=frozenset({'yogurt'}), confidence=0.4385964912280702,
lift=3.1940895929143247)])
RelationRecord(items=frozenset({'baking powder', 'sugar'}),
support=0.0037328356219170776,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'baking
powder'}), items_add=frozenset({'sugar'}),
confidence=0.20895522388059698, lift=6.269492537313432)])
RelationRecord(items=frozenset({'baking powder', 'whipped/sour
cream'}), support=0.005065991201173177,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'baking
powder'}), items_add=frozenset({'whipped/sour cream'}),
confidence=0.2835820895522388, lift=3.8816592221374875)])
RelationRecord(items=frozenset({'beef', 'root vegetables'}),
support=0.01759765364618051,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'beef'}),
items_add=frozenset({'root vegetables'}),
confidence=0.3464566929133858, lift=3.1348270850944595)])
RelationRecord(items=frozenset({'berries', 'whipped/sour cream'}),
support=0.007732302359685375,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'berries'}),
items_add=frozenset({'whipped/sour cream'}), confidence=0.25,
lift=3.4219890510948905)])
RelationRecord(items=frozenset({'bottled beer', 'liquor'}),
support=0.004532728969470737,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'liquor'}),
```



```
items_add=frozenset({'bottled beer'}), confidence=0.40963855421686746,
lift=5.087249660895236)])
RelationRecord(items=frozenset({'bottled beer', 'red/blush wine'}),
support=0.004932675643247567,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'red/blush
wine'}), items_add=frozenset({'bottled beer'}),
confidence=0.2624113475177305, lift=3.258853506176319)])
RelationRecord(items=frozenset({'margarine', 'flour'}),
support=0.0041327822956939075,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'flour'}),
items_add=frozenset({'margarine'}), confidence=0.23484848484848483,
lift=3.9146632996632995)])
RelationRecord(items=frozenset({'flour', 'sugar'}),
support=0.005065991201173177,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'flour'}),
items_add=frozenset({'sugar'}), confidence=0.28787878787878785,
lift=8.637515151515151)])
RelationRecord(items=frozenset({'flour', 'whipped/sour cream'}),
support=0.0041327822956939075,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'flour'}),
items_add=frozenset({'whipped/sour cream'}),
confidence=0.23484848484848483, lift=3.2145957752709577)])
RelationRecord(items=frozenset({'frankfurter', 'mustard'}),
support=0.0030662578322890282,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'mustard'}),
items_add=frozenset({'frankfurter'}),
confidence=0.22999999999999998, lift=3.9209772727272725)])
RelationRecord(items=frozenset({'processed cheese', 'fruit/vegetable
juice'}), support=0.0034662045060658577,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'processed
cheese'}), items_add=frozenset({'fruit/vegetable juice'}),
confidence=0.21848739495798317, lift=3.012635936727632)])
RelationRecord(items=frozenset({'tropical fruit', 'grapes'}),
support=0.005999200106652446,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'grapes'}),
items_add=frozenset({'tropical fruit'}),
confidence=0.32608695652173914, lift=3.1040333259766055)])
RelationRecord(items=frozenset({'processed cheese', 'ham'}),
support=0.003332888948140248,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'processed
cheese'}), items_add=frozenset({'ham'}),
confidence=0.21008403361344538, lift=8.20750175070028)])
RelationRecord(items=frozenset({'white bread', 'ham'}),
support=0.005332622317024397,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'ham'}),
items_add=frozenset({'white bread'}), confidence=0.20833333333333331,
lift=4.8231738683127565)])
RelationRecord(items=frozenset({'herbs', 'root vegetables'}),
support=0.006532462338354886,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'herbs'}),
items_add=frozenset({'root vegetables'}),
confidence=0.40495867768595045, lift=3.6641677217398243)])
RelationRecord(items=frozenset({'processed cheese', 'white bread'}),
support=0.004799360085321957,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'processed
cheese'}), items_add=frozenset({'white bread'}),
confidence=0.3025210084033613, lift=7.003734827264238)])
RelationRecord(items=frozenset({'rice', 'root vegetables'}),
support=0.003199573390214638,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'rice'}),
```



```
items_add=frozenset({'root vegetables'}),
confidence=0.4067796610169492, lift=3.680644435811985)])
RelationRecord(items=frozenset({'sliced cheese', 'sausage'}),
support=0.006932409012131715,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'sliced
cheese'}), items_add=frozenset({'sausage'}),
confidence=0.30057803468208094, lift=3.239419307687197)])
RelationRecord(items=frozenset({'tropical fruit', 'turkey'}),
support=0.0030662578322890282,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'turkey'}),
items_add=frozenset({'tropical fruit'}),
confidence=0.3484848484848485, lift=3.3172396554376253)])
```

Keterangan kode:

- Kode `results = list(rules)` digunakan untuk mengonversi hasil aturan asosiasi yang dihasilkan oleh algoritma Apriori (yang disimpan dalam bentuk generator) menjadi list yang lebih mudah digunakan dan diakses. Ketika algoritma Apriori dijalankan, hasilnya berupa objek generator, yang merupakan struktur data yang menghasilkan nilai-nilai secara lazy, artinya nilai-nilai baru dihasilkan saat kita mengaksesnya dan tidak semua nilai dihasilkan sekaligus pada awal.

Ketika kita ingin menyimpan hasil aturan asosiasi untuk digunakan atau ditampilkan, seringkali lebih nyaman dan lebih mudah bekerja dengan list daripada generator. Dengan mengonversi hasil dari generator menjadi list menggunakan `list(rules)`, kita bisa dengan mudah mengakses dan menggunakan hasil aturan asosiasi secara keseluruhan tanpa harus mengkhawatirkan tentang cara kerja generator.

- Kode `for rule in results`, digunakan untuk melakukan iterasi (pengulangan) melalui setiap aturan asosiasi yang ada dalam `results`. Setiap aturan asosiasi akan menjadi elemen dari `rule` pada setiap iterasi, sehingga Anda dapat mengakses dan melakukan tindakan terhadap setiap aturan secara terpisah.
- `print(rule)` digunakan untuk mencetak atau menampilkan satu aturan asosiasi (dalam bentuk `RelationRecord`) ke dalam output konsol atau tampilan layar. Ketika kita melakukan iterasi melalui hasil algoritma Apriori, setiap aturan asosiasi akan menjadi nilai dari variabel `rule`, sehingga kita dapat mencetak atau menampilkan satu aturan asosiasi pada setiap iterasi.

3.15.8 Merubah Dataset Ke Dalam Bentuk Kolom

```
def inspect(results):
    lhs          = [tuple(result[2][0][0])[0] for result in results]
    rhs          = [tuple(result[2][0][1])[0] for result in results]
    supports     = [result[1] for result in results]
    confidences  = [result[2][0][2] for result in results]
    lifts        = [result[2][0][3] for result in results]
```



```
return list(zip(lhs, rhs, supports, confidences, lifts))
resultsinDataFrame = pd.DataFrame(inspect(results), columns = ['Left Hand Side', 'Right Hand Side', 'Support', 'Confidence', 'Lift'])
```

Keterangan kode:

- Fungsi 'inspect(results)' merupakan sebuah fungsi yang digunakan untuk memeriksa dan mengeksplorasi hasil dari analisis aturan asosiasi yang dihasilkan oleh algoritma Apriori. Fungsi ini membantu Anda dalam melakukan tugas-tugas tertentu terhadap hasil aturan asosiasi yang berguna untuk analisis lebih lanjut atau pemahaman pola asosiasi dalam dataset transaksi.
- Pada kode 'lhs = [tuple(result[2][0][0])[0] for result in results]', itu digunakan untuk mendapatkan item di sebelah kiri (left-hand side) dari setiap aturan asosiasi yang ada dalam hasil 'results'. Aturan asosiasi terdiri dari itemset $X \rightarrow Y$, di mana X adalah left-hand side dan Y adalah right-hand side dari aturan tersebut. Jadi, 'lhs' akan berisi daftar item di sebelah kiri dari setiap aturan asosiasi dalam 'result'.
- Pada kode 'rhs = [tuple(result[2][0][0])[0] for result in results]', itu digunakan untuk mendapatkan item di sebelah kanan (right-hand side) dari setiap aturan asosiasi yang ada dalam hasil 'results'. Aturan asosiasi terdiri dari itemset $X \rightarrow Y$, di mana X adalah left-hand side dan Y adalah right-hand side dari aturan tersebut. Jadi, 'lhs' akan berisi daftar item di sebelah kiri dari setiap aturan asosiasi dalam 'result'.
- Kode 'supports = [result[1] for result in results]' digunakan untuk mengumpulkan nilai support dari setiap aturan asosiasi yang ada dalam hasil 'results' ke dalam sebuah list bernama 'supports'. Jadi, 'supports' akan berisi daftar nilai support dari setiap aturan asosiasi dalam 'results'.
- Kode 'confidences = [result[2][0][2] for result in results]' digunakan untuk mengumpulkan nilai confidence dari setiap aturan asosiasi yang ada dalam hasil 'results' ke dalam sebuah list bernama 'confidences'. Jadi, 'confidences' akan berisi daftar nilai confidence dari setiap aturan asosiasi dalam 'results'.
- Kode 'lifts = [result[2][0][3] for result in results]' digunakan untuk mengumpulkan nilai lift dari setiap aturan asosiasi yang ada dalam hasil 'results' ke dalam sebuah list bernama 'lifts'. Jadi, 'lifts' akan berisi daftar nilai lift dari setiap aturan asosiasi dalam 'result'.
- Kode 'return list(zip(lhs, rhs, supports, confidences, lifts))' digunakan untuk menggabungkan atau mengelompokkan beberapa list 'lhs', 'rhs', 'supports', 'confidences', dan 'lifts' menjadi satu list berisi tupel-tupel yang berisi informasi yang



relevan dari setiap aturan asosiasi dalam bentuk '(lhs, rhs, support, confidence, lift)'. Jadi, fungsi tersebut menggabungkan informasi dari 'lhs', 'rhs', 'supports', 'confidences', dan 'lifts' untuk setiap aturan asosiasi menjadi satu list yang berisi tupel-tupel, sehingga Anda dapat dengan mudah menyimpan dan mengelola informasi tersebut untuk analisis lebih lanjut atau tujuan lain yang relevan.

- Kode `resultsinDataFrame = pd.DataFrame(inspect(results), columns=['Left Hand Side', 'Right Hand Side', 'Support', 'Confidence', 'Lift'])` digunakan untuk mengubah hasil analisis aturan asosiasi dari list `inspect(results)` menjadi suatu DataFrame yang terstruktur dengan kolom-kolom yang sesuai. Jadi, `resultsinDataFrame` akan berisi DataFrame yang mengandung informasi lengkap dari setiap aturan asosiasi dalam results, dengan kolom-kolom yang sesuai, yaitu 'Left Hand Side', 'Right Hand Side', 'Support', 'Confidence', dan 'Lift'.

3.15.9 Melihat Hasil

```
print(resultsinDataFrame)
```

	Left Hand Side	Right Hand Side	Support	Confidence	Lift
0	10	fruit/vegetable juice	0.006133	0.251366	3.465988
1	10	margarine	0.005466	0.224044	3.734560
2	10	root vegetables	0.008132	0.333333	3.016084
3	11	domestic eggs	0.003866	0.218045	3.379249
4	11	margarine	0.003866	0.218045	3.634570
5	12	domestic eggs	0.003466	0.276596	4.286663
6	12	fruit/vegetable juice	0.003200	0.255319	3.520494
7	12	other vegetables	0.007599	0.606383	3.156474
8	12	pip fruit	0.003333	0.265957	3.445504
9	12	root vegetables	0.004399	0.351064	3.176514
10	12	tropical fruit	0.004133	0.329787	3.139256
11	12	whipped/sour cream	0.003333	0.265957	3.640414
12	13	whipped/sour cream	0.003066	0.377049	5.161033
13	13	yogurt	0.003733	0.459016	3.342798
14	14	other vegetables	0.004933	0.649123	3.378952
15	14	root vegetables	0.003466	0.456140	4.127272
16	14	yogurt	0.003333	0.438596	3.194090
17	baking powder	sugar	0.003733	0.208955	6.269493
18	baking powder	whipped/sour cream	0.005066	0.283582	3.881659
19	beef	root vegetables	0.017598	0.346457	3.134827
20	berries	whipped/sour cream	0.007732	0.250000	3.421989
21	liquor	bottled beer	0.004533	0.409639	5.087250



22	red/blush wine	bottled beer	0.004933	0.262411	3.258854
23	flour	margarine	0.004133	0.234848	3.914663
24	flour	sugar	0.005066	0.287879	8.637515
25	flour	whipped/sour cream	0.004133	0.234848	3.214596
26	mustard	frankfurter	0.003066	0.230000	3.920977
27	processed cheese	fruit/vegetable juice	0.003466	0.218487	3.012636
28	grapes	tropical fruit	0.005999	0.326087	3.104033
29	processed cheese	ham	0.003333	0.210084	8.207502
30	ham	white bread	0.005333	0.208333	4.823174
31	herbs	root vegetables	0.006532	0.404959	3.664168
32	processed cheese	white bread	0.004799	0.302521	7.003735
33	rice	root vegetables	0.003200	0.406780	3.680644
34	sliced cheese	sausage	0.006932	0.300578	3.239419
35	turkey	tropical fruit	0.003066	0.348485	3.317240

Keterangan kode:

- `'print(resultsinDataFrame)'` digunakan untuk mencetak atau menampilkan DataFrame `'resultsinDataFrame'` ke dalam output konsol atau tampilan layar. Dengan menggunakan fungsi `'print()'`, Anda dapat melihat isi dari DataFrame yang telah dibuat atau hasil analisis aturan asosiasi dalam bentuk tabular.

3.16 Kelebihan dan Kekurangan Association Rule Learning

Beberapa kelebihan Association rule learning:

1. Mudah dipahami.
2. Mendukung penambahan data tidak terarah
3. Bekerja pada catatan data panjang variabel dan perhitungan sederhana.
4. Pemahaman pola hubungan : Association Rule Learning dapat membantu dalam pemahaman pola hubungan antara item atau variabel dalam data. Hal ini dapat mengungkapkan hubungan yang mungkin tidak terlihat secara intuitif atau terdapat dalam data yang kompleks.
5. Pengambilan keputusan: Aturan asosiasi dapat digunakan untuk pengambilan keputusan yang lebih baik dalam berbagai bidang seperti pemasaran, manajemen rantai pasokan, dan rekomendasi produk. Dengan memahami hubungan antara item, organisasi atau bisnis dapat mengoptimalkan strategi mereka dan meningkatkan kinerja.
6. Identifikasi Peluang Bisnis: Dengan menggunakan Association Rule Learning, bisnis dapat mengidentifikasi peluang baru, peningkatan penjualan lintas produk, dan peluang lintas penjualan. Ini memungkinkan mereka untuk mengembangkan strategi pemasaran yang lebih efektif dan meningkatkan pendapatan.



7. Efisiensi Operasional: Dengan pemahaman yang lebih baik tentang pola pembelian pelanggan, bisnis dapat mengatur stok dan rantai pasokan dengan lebih efisien. Ini dapat mengurangi biaya operasional, menghindari kelebihan persediaan, dan memastikan ketersediaan produk yang tepat pada waktu yang tepat.

Kekurangan Association rule learning:

1. Peningkatan eksponensial dalam komputasi dengan sejumlah item (algoritma Apriori)
2. Keterbatasan pada Data Terstruktur: Association Rule Learning umumnya bekerja lebih baik dengan data terstruktur yang memiliki atribut diskrit atau kategorik. Data yang kontinu atau memiliki atribut berkelanjutan mungkin membutuhkan pra-pemrosesan atau pendekatan lain sebelum dapat dianalisis dengan baik menggunakan aturan asosiasi.
3. Masalah Spuriousity: Aturan asosiasi dapat menghasilkan aturan yang terlihat signifikan tetapi sebenarnya tidak bermakna atau hanya kebetulan. Ini dapat terjadi ketika terdapat hubungan yang tidak terduga dalam data yang tidak relevan secara praktis.
4. Masalah skala dan efisiensi: Analisis aturan asosiasi dapat menjadi sulit untuk data yang sangat besar dan kompleks. Pencarian aturan yang valid dapat memakan waktu dan membutuhkan sumber daya komputasi yang signifikan.
5. Tidak memberikan penjelasan Sebab-akibat: Aturan asosiasi hanya menunjukkan hubungan antara item atau variabel, tetapi tidak memberikan penjelasan sebab akibat mengapa hubungan tersebut ada. Oleh karena itu, informasi yang lebih mendalam mungkin diperlukan untuk memahami hubungan yang terungkap oleh aturan asosiasi.

Penting untuk mempertimbangkan kelebihan dan kekurangan ini ketika menggunakan Association Rule Learning dan memastikan bahwa pendekatan ini sesuai dengan konteks dan tujuan analisis data yang dilakukan.

BAB IV

PENUTUP

4.1 Kesimpulan

Berdasarkan hasil perhitungan menggunakan python di atas dapat disimpulkan bahwa: pembeli yang membeli baking powder (bubuk pengembang) kemungkinan akan membeli sugar (gula), pembeli yang membeli baking powder (bubuk pengembang) kemungkinan juga akan membeli (krim kue), pembeli yang membeli beef (daging sapi) kemungkinan akan membeli root vegetables (sayuran berakar), pembeli yang membeli berries (buah beri) kemungkinan akan membeli whipped/sour cream (krim kue), pembeli yang membeli liquor (minuman beralkohol) kemungkinan akan membeli bottled beer (bir botol), pembeli yang membeli red/blushwine (anggur merah) kemungkinan akan membeli bottled beer (bir botol), pembeli yang membeli flour (tepung) kemungkinan akan membeli margarine (mentega),



pembeli yang membeli flour (tepung) kemungkinan akan membeli sugar (gula), pembeli yang membeli flour (tepung) kemungkinan akan membeli whipped/sour cream (krim kue), pembeli yang membeli mustard (saus moster) kemungkinan akan membeli frankfurter (sisis), pembeli yang membeli processed cheese (keju) kemungkinan akan membeli fruit/vegetable juice (jus sayur/buah), pembeli yang membeli grapes (anggur) kemungkinan akan membeli tropical fruit (buah tropis), pembeli yang membeli processed cheese (keju) kemungkinan akan membeli ham (daging babi yang dikeringkan), pembeli yang membeli ham (daging babi yang dikeringkan) kemungkinan akan membeli white bread (roti putih), pembeli yang membeli herbs (tanaman herba) kemungkinan akan membeli root vegetables (sayuran akar), pembeli yang membeli rice (beras) kemungkinan akan membeli root vegetables (sayuran berakar), pembeli yang membeli sliced cheese (irisasi keju) kemungkinan akan membeli sausage (sisis), dan pembeli yang membeli turkey (ayam kalkun) kemungkinan akan membeli tropical fruit (buah tropis).

4.2 Saran

Dalam mempelajari aturan asosiasi (Algoritma Apriori) ada beberapa hal yang harus kita perhatikan, yaitu diantaranya adalah:

1. Pahami Konsep Aturan Asosiasi: Mulailah dengan memahami konsep dasar aturan asosiasi, seperti support, confidence, dan lift. Ketahui bagaimana aturan asosiasi dapat membantu mengidentifikasi hubungan antara item dalam data transaksi dan bagaimana menginterpretasi hasilnya.
2. Pelajari Algoritma Apriori: Pelajari dengan baik cara kerja algoritma Apriori. Pahami bagaimana algoritma tersebut bekerja untuk menemukan aturan asosiasi dengan berbagai tingkat support dan confidence.
3. Kenali Dataset Transaksi: Pastikan Anda mengerti tentang dataset transaksi yang akan digunakan dalam analisis. Ketahui struktur data dan formatnya serta bagaimana cara mengolahnya untuk digunakan dalam algoritma Apriori.
4. Implementasi Algoritma: Cobalah untuk mengimplementasikan algoritma Apriori secara manual dalam bahasa pemrograman yang Anda kuasai (misalnya Python). Ini akan membantu Anda memahami prosesnya dengan lebih baik dan meningkatkan pemahaman Anda tentang algoritma tersebut.
5. Gunakan Library atau Package yang Tersedia: Manfaatkan library atau package yang menyediakan implementasi algoritma Apriori. Contohnya, di Python, Anda dapat menggunakan library seperti apyori atau mlxtend untuk menerapkan algoritma Apriori dengan mudah.



6. Eksplorasi Parameter: Eksplorasi berbagai parameter dalam algoritma Apriori seperti `min_support`, `min_confidence`, dan lainnya. Pahami bagaimana nilai-nilai parameter ini mempengaruhi hasil aturan asosiasi yang dihasilkan.
 7. Analisis Hasil: Setelah mendapatkan hasil aturan asosiasi, lakukan analisis lebih lanjut terhadap hasil tersebut. Identifikasi aturan yang paling relevan dan berarti dalam konteks dataset Anda.
 8. Pelajari Aturan Asosiasi Lainnya: Selain algoritma Apriori, ada juga metode lain untuk menemukan aturan asosiasi seperti FP-Growth dan Eclat. Pelajari perbedaan dan kelebihan dari setiap metode untuk memahami kapan dan bagaimana cara menggunakannya.
 9. Studi Kasus: Coba terapkan algoritma Apriori pada berbagai studi kasus dan dataset yang berbeda. Dengan menghadapi berbagai masalah dan data, Anda akan mendapatkan pengalaman yang berharga dalam memahami aturan asosiasi.
 10. Sumber Belajar: Manfaatkan berbagai sumber belajar seperti buku, tutorial online, video, dan kursus terstruktur yang membahas aturan asosiasi dan algoritma Apriori.
- Ingatlah bahwa mempelajari aturan asosiasi adalah proses yang memerlukan latihan dan pemahaman mendalam tentang konsep dan algoritma yang terlibat. Dengan berlatih dan eksplorasi lebih lanjut, Anda akan menjadi lebih mahir dalam menerapkan dan memahami aturan asosiasi untuk analisis data dan penemuan pola dalam transaksi.

UCAPAN TERIMA KASIH

Terima kasih saya ucapkan kepada Bapak Syahid Abdullah, S.Si., M.Kom. yang telah memberikan pengalamannya dalam mengajarkan ilmunya kepada kami terutama dalam hal mata kuliah Machine Learning selama semester lima ini di Universitas Siber Asia (UNSI). Saya sangat bangga dengan cara menyampaikan materi pembelajaran baik secara sinkron maupun asinkron. Di sini kami telah mempelajari materi yang telah diajarkan dari pertemuan ke-1 yaitu mempelajari tentang Konsep Machine Learning, pada pertemuan ke-2 membahas tentang Supervised dan Unsupervised Learning, kemudian pada sesi ke-3 membahas tentang Praproses Data, pada sesi ke-4 belajar tentang Supervised Learning: Regresi Linear Sederhana, lanjut pada pertemuan ke-5 membahas tentang Supervised Learning: Naïve Bayes, kemudian pada sesi ke-6 kita belajar tentang Supervised Learning: KNN (K-Nearest Neighbor), pada pertemuan ke-7 membahas tentang Supervised Learning: SVM (Support Vector Machine), kemudian kita lanjut pada Ujian Tengah Semester (UTS) yang diadakan pada pertemuan ke delapan. Pada pertemuan ke-9 kita lanjut ke sesi berikutnya yaitu membahas tentang Unsupervised Learning: K-Means Clustering, pada sesi berikutnya yaitu pertemuan ke-10 membahas tentang Unsupervised Learning: Hierarchical Clustering, kemudian melangkah ke sesi-11 membahas tentang Aturan asosiasi (Algoritma Apriori), pada



sesi ke-12 ada Artificial Neural Network (ANN)/Jaringan Syaraf Tiruan, selanjutnya pada sesi ke-13 belajar tentang Reinforcement Learning, Principal Component Analysis (PCA) diajarkan pada sesi ke-14, dan pada pertemuan terakhir yaitu sesi ke-15 kita belajar mengenai Studi Kasus Machine Learning. Namun masih ada lagi pertemuan yang paling kita nanti yaitu pada sesi ke-16 kita mengadakan Ujian Akhir semester (UAS). Tidak lupa juga saya ucapkan rasa terima kasih yang banyak kepada Bapak Ikhwan Saputra, S.Kom., M.Kom. selaku dosen pembimbing dan semua teman-teman yang telah membantu dalam menyelesaikan laporan penelitian ini. Kami sangat senang dengan adanya diskusi yang diadakan pada setiap sesinya beserta praktikum-praktikum yang telah dilakukan menggunakan aplikasi Google Collab, semoga kita bisa bertemu pada semester berikutnya.

DAFTAR PUSTAKA

1. Syahid Abdullah, S. M. (2023, Juni 28). *Machine Learning*. Diambil kembali dari Edlink Universitas Siber Asia: <https://api.edlink.id/api/v1.4/media/download/eyJpdil6IIBDc0MwYkZoUm0ram40cnRRQUZrRlE9PSIsInZhbnVlIjoia1hrN0NlQUFvakM1aTFLejZNVVJ2dz09IiwibWFjIjojZjYyZjZiZGQxODM2Yjk0MTdhMzkxMjRlMzNkM2ZiNWNiNGFiMjM3MDMyNTczZjViNzU1NDM0YzU4MDU0YzUyMSJ9>. Diakses pada 21 Juni 2023.
2. Irwansyah Saputra, D. A. (2021). *Machine Learning Untuk Pemula*. Bandung: Informatika.
3. Nasrullah, I. (2019, Juli 16). *Groceries Market Basket Dataset*. Diambil kembali dari kaggle: <https://www.kaggle.com/datasets/irfanarullah/groceries>. Diakses pada 21 Juli 2023.
4. Trivusi. (2022, September 17). *Algoritma Apriori: Pengertian, Cara Kerja, Kelebihan, dan Kekurangannya*. Diambil kembali dari Trivusi: <https://www.trivusi.web.id/2022/08/algoritma-apriori.html>. Diakses pada 21 Juli 2023.
5. ilmuskripsi. (2016, Juni 30). *Association Rules*. Diambil kembali dari ilmuskripsi.com: <https://www.ilmuskripsi.com/2016/06/association-rules.html>. Diakses pada 23 Juli 2023.



Link Google Collab


<https://colab.research.google.com/drive/1-TVlKpxRLrLZmb9iyX0maOCX56Y-4E3x>

Link Google Drive

<https://drive.google.com/drive/folders/129K0AivSWFxUhwYxF1BiCICzaBCneXhU>

Link GitHub

https://github.com/Hendro10/TUGAS2_MACHINE_LEARNING_NIM-200401072103_NAMA_HENDRO_GUNAWAN_KELAS_IT-602/tree/main

Nilai	Tanda Tangan Dosen Pengampu	Tanda Tangan Mahasiswa
		
	(Syahid Abdullah, S.Si., M.Kom)	(Hendro Gunawan)
Diserahkan pada Tanggal :		Tanggal Mengumpulkan :
		23/07/2023