PENGGALIAN DATA – IS184943

TUGAS GROUP PROJECT #3

Analisis Asosiasi

SULIS AVANDHY PUTRA	-	05211940000084
MUHAMMAD ZUHDI AFI ABIYYI	-	05211940000135
AFLAH ADITYA	-	05211942000001

Program Studi Sarjana
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya

Tahun 2022

Ringkasan Progress Tugas Group Project

B. Tugas

1. Setiap kelompok diharuskan mengumpulkan dua laporan berbeda untuk tugas analisis klaster dan tugas analisis asosiasi.
2. Lakukan ekplorasi data secara umum.
3. **Untuk tugas analisis klaster:**
   a. Lakukan praproses data yang diperlukan untuk keperluan analisis klaster
   b. Lakukan proses *clustering* menggunakan tiga metode yang berbeda: **partisional (K-means), metode hirarki (MIN, MAX, dan AVERAGE), dan metode berbasis densitas (DBScan).** Gunakan library python yang sesuai untuk masing-masing metode.
   c. Untuk metode DBScan, lakukan eskperimen untuk beberapa nilai *minimum points* dan *epsilon* yang berbeda untuk mendapatkan hasil terbaik.
   d. Jumlah klaster dari tugas ini sesuai dengan jumlah nilai atribut *"Kingdom"*. Untuk itu lakukan proses evaluasi menggunakan ***entropy*** dan ***purity*** baik untuk megukur kualitas masing-masing klaster yang dihasilkan maupun kualitas keseluruhan hasil klaster.
4. **Untuk tugas analisis asosiasi:**
   a. Lakukan praproses data yang diperlukan untuk keperluan analisis asosiasi, terutama praproses untuk mentrasformasikan data agar dapat diperlakukan sebagai item, sehingga dapat dilakukan analisis asosiasi. **Dalam hal ini setiap pasangan atribut dan nilainya dapat dianggap sebagai sebuah item.**
   b. Lakukan proses pembangkitan *frequent itemsets* dengan menggunakan algoritma *FP-growth*. Lakukan uji coba untuk berbagai nilai ambang batas *support* dan tentukan nilai ambang batas *support* yang pas menurut hasil uji coba. Kemudian lakukan perbandingan yang diperoleh menggunakan kedua algoritma tersebut berdasarkan waktu komputasi yang dibutuhkan oleh masing-masing algoritma.
   c. Bangkitkan sejumlah aturan asosiasi (*association rules*) yang menarik dari satu set *frequent itemsets* yang diperoleh. Salah satu aturan asosiasi yang harus dibangkiktkan adalah menjadikan atribut ***"Ncodons"*** sebagai target beserta data statistik berupa nilai rataan dari Ncodons dan juga juga simpangan baku dari aturan asosiasi yang dibangktkan. Lakukan analisis kemenarikan (*interestingness*) dari aturan yang dihasilkan menggunakan berbagai ukuran kemenarikan (selain hanya menggunakan *confidence*). Lakukan uji coba untuk berbagai nilai ambang batas ukuran kemenarikan dan buat kesimpulan dari hasil uji coba tersebut.

A. Pendahuluan

1. Dataset

Data yang digunakan pada tugas ini adalah data terkait DNA Codon yang diperoleh dari UCI Machine Learning Repository. Kodon adalah urutan trinukleotida DNA atau RNA yang sesuai dengan asam amino tertentu. Kode genetik menggambarkan hubungan antara urutan basa DNA (A, C, G, dan T) dalam gen dan urutan protein yang sesuai yang dikodekannya. Sel membaca urutan gen dalam kelompok tiga basa. Ada 64 kodon yang berbeda: 61 pertama menyatakan urutan sinyal asam amino sedangkan tiga sisanya menyatakan sinyal akhir dari urutan asam amino. Data ini terdiri dari terdiri 13.028 baris dan 69 kolom/atribut.

Berikut merupakan beberapa variabel/atribut dalam dataset yang dapat dikelompokkan menjadi empat kategori sebagai berikut:

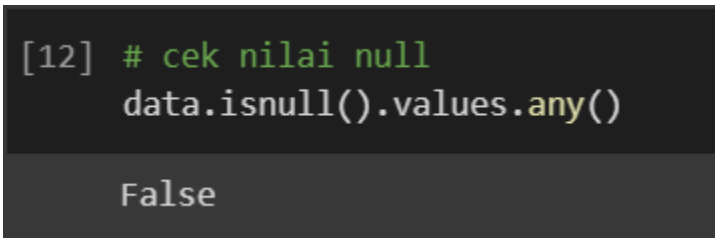
1) Kingdom, 'Kingdom' adalah kode 3 huruf yang sesuai dengan `xxx' dalam nama basis data CUTG: 'arc'(archaea), 'bct'(bacteria), 'phg'(bacteriophage), 'plm' (plasmid), ' entri urutan pln' (tanaman), 'inv' (invertebrata), 'vrt' (vertebrata), 'mam' (mamalia), 'rod' (tikus), 'pri' (primata), dan 'vrl'(virus) . Perhatikan bahwa basis data CUTG tidak mengandung 'arc' dan 'plm' (ini telah dikuratori sendiri secara manual).

2) DNAtype. 'DNAType' dilambangkan sebagai bilangan bulat untuk komposisi genom dalam spesies: 0-genomic, 1-mitochondrial, 2-chloroplast, 3-cyanelle, 4-plastid, 5-nucleomorph, 6-secondary_endosymbiont, 7-chromoplast, 8-leucoplast, 9-NA, 10-proplastid, 11-apicoplast, and 12-kinetoplast.
3) SpeciesID, 'SpeciesID' adalah bilangan bulat, yang secara unik menunjukkan entri suatu organisme. Ini adalah pengidentifikasi aksesi untuk setiap spesies berbeda dalam basis data CUTG asli, diikuti oleh item pertama yang tercantum dalam setiap genom.
4) Ncodons, jumlah kodon (`Ncodons') adalah jumlah aljabar dari angka yang terdaftar untuk kodon yang berbeda dalam entri CUTG. Frekuensi kodon dinormalisasi ke jumlah total kodon, maka jumlah kejadian dibagi dengan 'Ncodons' adalah frekuensi kodon yang tercantum dalam file data.
5) SpeciesName, nama spesies ('SpeciesName') diwakili dalam string yang dibersihkan dari `koma' (yang sekarang diganti dengan `spasi'). Ini adalah label deskriptif nama spesies untuk interpretasi data.
6) Codon, frekuensi kodon ('Codon') termasuk 'UUU', 'UUA', 'UUG', 'CUU', dll., dicatat sebagai float (dengan desimal dalam 5 digit).

B. Pra Proses Data

1. Missing Value

Missing value merupakan suatu kondisi dimana suatu atribut memiliki nilai null. Kondisi ini perlu ditangani dengan tepat agar model yang dibangun dapat melakukan klasifikasi dengan baik. Ada beberapa cara penanganannya diantaranya yaitu menghapus kolom atau baris yang memiliki missing value atau mengganti nilainya dengan nilai lain seperti median, mean, atau mode.



Berdasarkan gambar di atas, dataset data tidak memiliki nilai null pada setiap atribut sehingga tidak diperlukan penanganan missing value lebih lanjut.

2. Menghapus Atribut yang Tidak Digunakan

Tahapan ini dilakukan untuk memastikan agar hanya atribut kodon yang tersisa. Hal ini dikarenakan atribut kodon nantinya akan dijadikan sebagai item dalam analisis ini. Tahapan ini juga akan dilakukan untuk penghilangan nilai pada atribut ‘Codon’ yang berisi string. Hal ini diketahui dari info dataset bahwa atribut ini hanya berisi data bertipe float namun ditemukan ada dua atribut yang belum bertipe float.

```
[30] #menghapus atribut kodon yang memiliki data bertipe string
     data_2 = data_2.drop(data_2.index[data_2['UUC'] == '-'])
     data_2 = data_2.drop(data_2.index[data_2['UUU'] == 'non-B hepatitis virus'])

     # menghapus atribut yang tidak digunakan
     data_2.drop('SpeciesName', axis=1, inplace=True)
     data_2.drop('SpeciesID', axis=1, inplace=True)
     data_2.drop('Ncodons', axis=1, inplace=True)
     data_2.drop('Kingdom', axis=1, inplace=True)
     data_2.drop('DNAtype', axis=1, inplace=True)
```

3. Mendefinisikan Fungsi Pemetaan
   Pada proses ini dilakukan untuk mendefinisikan sebuah fungsi yang akan memetakan frekuensi kodon menjadi dua nilai saja yaitu 1 dan 0. Nilai 1 mengartikan bahwa kodon tersebut memiliki frekuensi lebih dari 0 atau juga mengartikan bahwa DNA tersebut memiliki kodon tersebut. Sementara nilai 0 mengartikan bahwa frekuensi sama dengan nol atau mengartikan bahwa DNA tersebut tidak memiliki kodon tersebut.

```
[25] #membuat fungsi untuk memetakan ada tidaknya kodon
     def encode(x):
       if x > 0:
         return 1
       else:
         return 0
```

4. Implementasi Fungsi dan Pemilihan Kodon
   Pada tahap ini, fungsi yang sudah dibuat sebelumnya akan diimplementasikan pada dataset. Namun sebelum pengimplementasiannya, dataset terlebih dahulu diubah tipe datanya menjadi float sehingga memastikan tidak ada data yang memiliki tipe selain float. Setelah itu data akan diambil 10 kodon terakhir sebagai sampel masukan pada analisis asosiasi selanjutnya.

```
#mengubah tipe data menjadi float
codon = data_2.astype('float64')

#mengaplikasikan fungsi ke dataset
codon_2 = codon.applymap(encode)

#mengambil sampel 10 kodon terakhir
codon_3 = codon_2.iloc[:,54:]
codon_3
```

   Dataset yang akan digunakan sebagai masukan adalah sebagai berikut:

	CGG	AGA	AGG	GAU	GAC	GAA	GAG	UAA	UAG	UGA
0	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	0
2	1	1	1	1	1	1	1	1	0	1
3	1	1	1	1	1	1	1	1	1	0
4	1	1	1	1	1	1	1	0	1	1
...	...	...	...	...	...	...	...	...	...	...
13023	1	1	1	1	1	1	1	1	1	1
13024	1	0	1	1	1	1	1	1	1	1
13025	0	0	0	1	1	1	1	1	1	1
13026	1	1	1	1	1	1	1	1	1	1
13027	1	1	1	1	1	1	1	1	1	1

13026 rows × 10 columns

C. Implementasi Analisis Asosiasi dengan FP Growth

1. Install library

Proses analisis asosiasi dengan FP Growth diawali dengan melakukan install dan import beberapa library yang dibutuhkan sebelum melakukan proses analisis asosiasi. Pada tabel berikut merupakan list beberapa library yang dibutuhkan dalam menjalankan model FP Growth.

Library	**Fungsi**
`!pip install mlxtend==0.18.0`	Install library untuk penerapan model FP Growth
`from mlxtend.frequent.patterns import fpgrowth`	Library ini berfungsi untuk mengimplementasikan FP-Growth untuk mengekstrak frekuensi itemsets untuk aturan asosiasi

2. Pembentukan *itemsets*

Proses analisis asosiasi diawali dengan membangun sebuah himpunan yang berisi item-item yang sering muncul. Pada proses ini ditetapkan ambang batas terendah (*treshold*) dari kemunculan menggunakan support 50%. Algoritma yang digunakan adalah fp growth. Itemset yang paling sering muncul adalah 'GAA' dengan nilai support tertinggi yaitu 0.998772.

```
n_minsupport = 0.5

# Menghasilkan association rules berdasarkan dataset
frequent_itemsets_fpgrowth = fpgrowth(codon_3, min_support= n_minsupport, use_colnames=True)
frequent_itemsets_fpgrowth.sort_values(by='support', ascending=False)
```

	support	itemsets
0	0.998772	(GAA)
1	0.991325	(GAC)
10	0.990481	(GAC, GAA)
2	0.979963	(GAU)
11	0.979426	(GAU, GAA)
...	...	...
631	0.503301	(AGG, GAA, UAG, GAG, AGA, CGG, UAA)
632	0.503301	(GAU, AGG, UAG, GAG, AGA, CGG, UAA)
633	0.503301	(AGG, GAA, UAG, GAC, AGA, CGG, UAA)
634	0.503301	(GAU, AGG, UAG, GAC, AGA, CGG, UAA)
614	0.503301	(AGG, UAG, AGA, CGG, UAA)

879 rows × 2 columns

3. Pembentukan *Association rules*

   Pembentukan *Association rules* menggunakan algoritma fp growth. Algoritma fp growth adalah algoritma untuk mengekstraksi frekuensi itemset dengan aplikasi dalam pembelajaran aturan asosiasi yang muncul sebagai alternatif populer dari algoritma 'Apriori' yang sudah ditetapkan. Itemset dianggap "sering" jika memenuhi ambang batas dukungan yang ditentukan. Misalnya, jika ambang batas dukungan diatur ke 0,5 (50%), frequent itemset didefinisikan sebagai sekumpulan item yang muncul bersama setidaknya 50% dari semua transaksi dalam database. Pada algoritma fp growth terdapat beberapa parameter yang perlu diperhatikan yaitu *min_support* dan *max_len*. *Min_support* adalah minimal support yang digunakan dalam analisa sedangkan *max_len* adalah total itemset maksimal yang akan ditampilkan. Terdapat beberapa *rule_metrics* pada algoritma fp growth diantaranya;

   a. '*support*' digunakan untuk mengukur frekuensi (sering diartikan sebagai signifikansi atau kepentingan) dari sebuah itemset dalam database
   b. '*confidence*' adalah probabilitas untuk melihat konsekuensi dalam suatu transaksi mengingat bahwa transaksi tersebut juga mengandung anteseden
   c. '*lift*' biasanya digunakan untuk mengukur seberapa sering anteseden dan konsekuensi dari aturan A->C terjadi bersamaan daripada yang kita harapkan jika mereka independen secara statistik. Jika A dan C independen, skor Lift akan tepat 1
   d. '*leverage*' menghitung perbedaan antara frekuensi teramati dari A dan C yang muncul bersamaan dan frekuensi yang diharapkan jika A dan C independen
   e. Nilai '*conviction*' yang tinggi berarti bahwa konsekuen sangat bergantung pada anteseden

- Pembangunan *association rule* dengan matriks “confidence”
  Parameter threshold yang digunakan dalam percobaan ini adalah 0.4. Matriks confidence digunakan untuk berapa kali suatu aturan muncul. Ini dapat dinyatakan secara berbeda sebagai probabilitas bersyarat dari sisi kanan yang diberikan sisi kiri. Pada *rule_metrics ‘confidence’* ini aturan yang diturunkan dari frequent itemset hanya jika tingkat kepercayaannya di atas ambang batas 40 persen. Dengan nilai confidence tertinggi yaitu 1.00000, dapat dilihat bahwa anteseden yang muncul adalah (‘GAG’ .’CGG’, ‘UAA’, ‘UAG’) sedangkan konsekuensi yang muncul adalah (‘GAC’ .’GAA’).

```
# Let's say you are interested in rules derived from the frequent itemsets only if the level of confidence is above the 4(
n_treshold = 0.4
rule_metrics = "confidence"

rules_result_fpgrowth = association_rules(frequent_itemsets_fpgrowth, metric= rule_metrics, min_threshold= n_treshold)

# urutkan berdasarkan nilai confidence
rules_result_fpgrowth.sort_values(by=[rule_metrics], ascending=False)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
16344	(GAG, CGG, UAA, UAG)	(GAC, GAA)	0.569476	0.990481	0.569476	1.00000	1.009611	0.005421	inf
23499	(AGG, GAA, UAG, AGA, UAA)	(GAU)	0.515891	0.979963	0.515891	1.00000	1.020447	0.010337	inf
25018	(AGG, UAG, AGA, CGG, UAA)	(GAG, GAA)	0.503301	0.965530	0.503301	1.00000	1.035700	0.017349	inf
28304	(GAG, UGA, AGA, UAG)	(GAA)	0.519039	0.998772	0.519039	1.00000	1.001230	0.000638	inf
23706	(AGG, UAG, GAC, AGA, UAA)	(GAU, GAA)	0.515815	0.979426	0.515815	1.00000	1.021006	0.010612	inf

- Pembangunan *association rule* dengan matriks “lift”
  Parameter threshold yang digunakan dalam percobaan ini adalah 1.2. Matriks lift digunakan untuk mengukur seberapa sering anteseden dan konsekuensi dari aturan yang ditetapkan terjadi bersamaan daripada yang diharapkan jika mereka independen secara statistik. Pada *rule_metrics ‘lift’* ini aturan yang diturunkan dari frequent itemset hanya jika tingkat kepercayaannya di atas ambang batas 120 persen. Dengan nilai lift tertinggi yaitu 1.223755, dapat dilihat bahwa anteseden yang muncul adalah (‘GAU’ .’AGG’, ‘GAA’, ‘UAG’) sedangkan konsekuensi yang muncul adalah (‘GAG’ .’GAC’, ‘AGA’, ‘CGG’, ‘UAA’).

```
# Let's say you are interested in rules derived from the frequent itemsets only if the level of confidence is above the 2(
n_treshold = 1.2
rule_metrics = "lift"

rules_result_fpgrowth_lift = association_rules(frequent_itemsets_fpgrowth, metric= rule_metrics, min_threshold= n_treshol(

# urutkan berdasarkan nilai confidence
rules_result_fpgrowth_lift.sort_values(by=[rule_metrics], ascending=False)
```

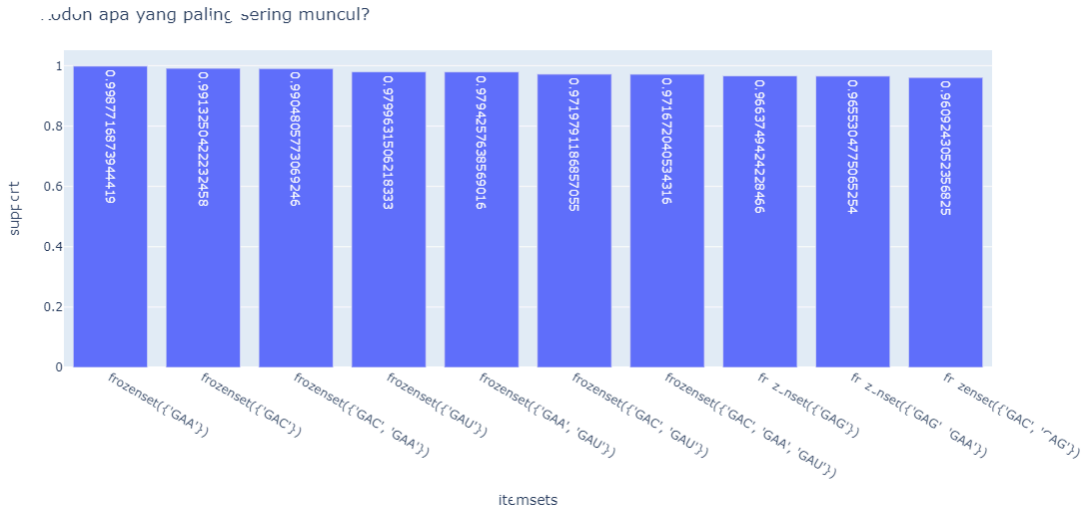
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
438	(GAU, AGG, GAA, UAG)	(GAG, GAC, AGA, CGG, UAA)	0.590204	0.696837	0.503301	0.852758	1.223755	0.092025	2.058937
437	(GAG, GAC, AGA, CGG, UAA)	(GAU, AGG, GAA, UAG)	0.696837	0.590204	0.503301	0.722265	1.223755	0.092025	1.475492
407	(GAA, GAG, GAC, AGA, CGG, UAA)	(GAU, AGG, UAG)	0.696837	0.590281	0.503301	0.722265	1.223595	0.091971	1.475216
279	(GAU, AGG, UAG)	(GAG, GAC, AGA, CGG, UAA)	0.590281	0.696837	0.503301	0.852647	1.223595	0.091971	2.057387
468	(GAU, AGG, UAG)	(GAA, GAG, GAC, AGA, CGG, UAA)	0.590281	0.696837	0.503301	0.852647	1.223595	0.091971	2.057387
...	...	...	...	...	...	...	...	...	...

4. Visualisasi Data

Setelah membangun model, langkah berikutnya yang dilakukan yaitu melakukan analisis asosiasi dari hasil pembangunan item serta pembangunan *association rule* dengan metriks confidence dan lift. Analisis asosiasi yang dilakukan yaitu:

- Kodon yang paling sering muncul

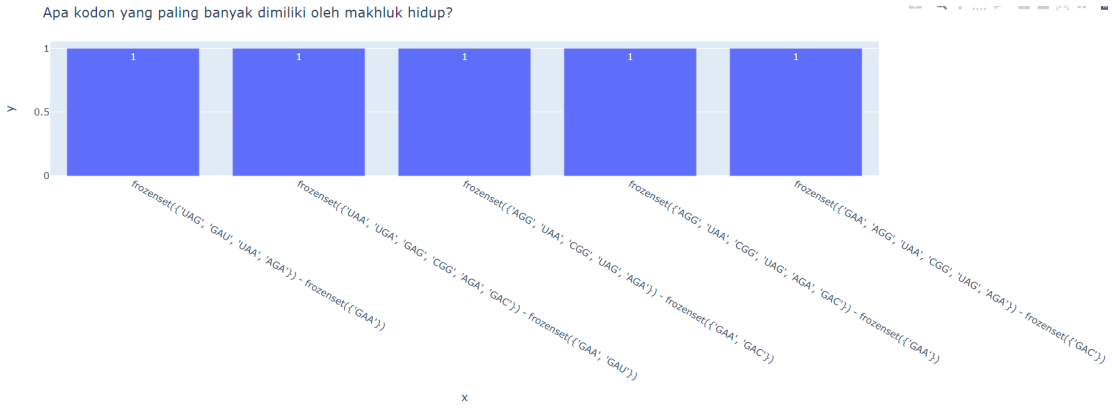
  Berdasarkan gambar dibawah, kodon yang paling sering muncul adalah {‘GAA’} dengan support 0.9987, {‘GAC’} dengan support 0.9913, {‘GAA’,‘GAC’} dengan support 0.9905, {‘GAU’} dengan support 0.9800, {‘GAA’,‘GAU’} dengan support 0.9794, {‘GAU’,‘GAC’} dengan support 0.9720, {‘GAA’, ‘GAU’,‘GAC’} dengan support 0.9717, {‘GAG’} dengan support 0.9664, {‘GAG’,‘GAA’} dengan support 0.9655, dan {‘GAG’,‘GAC’} dengan support 0.9610. Hal ini menunjukkan bahwa 99,87% DNA makhluk hidup pada dataset memiliki kodon GAA dan 99,05% dari DNA keseluruhan, kodon GAA akan muncul bersama dengan kodon GAC.



- Asosiasi kodon yang paling banyak

  Berdasarkan gambar dibawah, asosiasi kodon yang paling banyak adalah {‘UAG’, ‘GAU’, ‘UAA’, ‘AGA’} dengan {‘GAA’}, {‘UAA’, ‘UGA’, ‘GAG’, ‘CGG’, ‘AGA’, ‘GAC’} dengan {‘GAA’, ‘GAU’}, {‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘AGA’} dengan {‘GAA’,’GAC’}, {‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘AGA’, ‘GAC’} dengan {‘GAA’}, dan {‘GAA’, ‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘AGA’} dengan {‘GAC’}. Semua asosiasi yang terdapat pada gambar dibawah memiliki confidence bernilai 1. Hal ini menunjukkan bahwa jika terjadi kemunculan kodon yang berperan sebagai anticedent, 100% kemungkinan akan muncul juga kodon-kodon yang berperan sebagai consequent.

```
fig = px.bar(rules_result_fpgrowth_confidence.head(), x= rules_result_fpgrowth_confidence.head()['bundle'],
             y= rules_result_fpgrowth_confidence.head()['confidence'], text= rules_result_fpgrowth_confidence.head()['confidence'])
fig.update_layout(title_text= "Apa kodon yang paling banyak dimiliki oleh makhluk hidup?")
fig.show()
```



- Asosiasi kodon yang paling sedikit

  Berdasarkan gambar dibawah, asosiasi kodon yang paling banyak adalah {‘GAA’} dengan {‘AGG’, ‘UAA’, ‘GAG’, ‘CGG’, ‘UAG’, ‘AGA’, ‘GAC’}, {‘GAA’} dengan {‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘GAU’, ‘AGA’}, {‘GAA’} dengan {‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘AGA’}, {‘GAA’} dengan {‘AGG’, ‘UAA’, ‘CGG’, ‘UAG’, ‘AGA’, ‘GAC’}, dan {‘GAA’} dengan {‘AGG’, ‘UAA’, ‘GAG’, ‘CGG’, ‘UAG’, ‘AGA’}. Semua asosiasi yang terdapat pada gambar dibawah memiliki confidence yang bernilai sama yaitu 0.5039. Hal ini menunjukkan bahwa jika terjadi kemunculan kodon yang berperan sebagai anticedent, kemunculan kodon-kodon yang berperan sebagai consequent hanya 50,39%.

```
fig = px.bar(rules_result_fpgrowth_confidence.tail(), x= rules_result_fpgrowth_confidence.tail()['bundle'],
             y= rules_result_fpgrowth_confidence.tail()['confidence'], text= rules_result_fpgrowth_confidence.tail()['confidence'])
fig.update_layout(title_text= "Apa kodon yang paling dikit dimiliki oleh makhluk hidup?")
fig.show()
```

Apa kodon yang paling dikit dimiliki oleh makhluk hidup?