MODUL DATA MINING

ASSOSIATION RULES ANALYSIS

Pada modul ini dijelaskan mengenai proses assosiation rules analysis dengan menggunakan algoritma apriori dan menerapkannya dalam bahasa pemrograman python.

Diharapkan setelah mempelajari modul ini, mahasiswa mampu memahami manfaat assosiation rules analysis pada suatu kasus.

EPS 7

ASSOSIATION RULES ANALYSIS

Analisis aturan asossiasi sering digunakan untuk analisis transaksi penjualan. Analisis ini digunakan untuk mengetahui hubungan/asosiasi antar item yang dijual. Hasil dari asosiasi ini dapat digunakan dalam pembuatan strategi.

```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
import networkx as nx
import warnings
warnings.filterwarnings('ignore')
```

```
In [18]: data=pd.read_csv('D:/Data/BreadBasket_DMS.csv')
         data.head()
```

Out[18]:

	Date	Time	Transaction	Item
0	2016-10-30	09:58:11	1	Bread
1	2016-10-30	10:05:34	2	Scandinavian
2	2016-10-30	10:05:34	2	Scandinavian
3	2016-10-30	10:07:57	3	Hot chocolate
4	2016-10-30	10:07:57	3	Jam

Eksplorasi data

Tahap awal adalah melakukan eksplorasi data. Hal ini digunakan untuk analisis data deskriptif dan mencari apakah terdapat data yang kosong.

```
In [19]: data.shape
```

```
Out[19]: (21293, 4)
```

```
In [20]: data.describe()
```

Out[20]:

	Transaction
count	21293.000000
mean	4951.990889
std	2787.758400
min	1.000000
25%	2548.000000
50%	5067.000000
75%	7329.000000
max	9684.000000

```
In [21]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21293 entries, 0 to 21292
Data columns (total 4 columns):
Date           21293 non-null object
Time           21293 non-null object
Transaction    21293 non-null int64
Item           21293 non-null object
dtypes: int64(1), object(3)
memory usage: 665.5+ KB
```

```
In [22]: data.isnull().sum()
```

```
Out[22]: Date           0
         Time           0
         Transaction    0
         Item           0
         dtype: int64
```

Dari keterangan di atas dapat dinyatakan bahwa data tidak memiliki nilai null. Namun terdapat beberapa item yang berlabel ‘NONE’ (786). Item ini sebaiknya dihapus.

```
In [23]: data.loc[data['Item']=='NONE',:].count()
```

```
Out[23]: Date           786
         Time           786
         Transaction    786
         Item           786
         dtype: int64
```

In [24]:
```
data=data.drop(data.loc[data['Item']=='NONE'].index)
```

In [25]:
```
data.loc[data['Item']=='NONE',:].count()
```

Out[25]:
```
Date           0
Time           0
Transaction    0
Item           0
dtype: int64
```

Tahap berikutnya adalah mencari jumlah jenis item dari bakery menu yang dijual.

```
In [26]: data['Item'].nunique()
```

```
Out[26]: 94
```

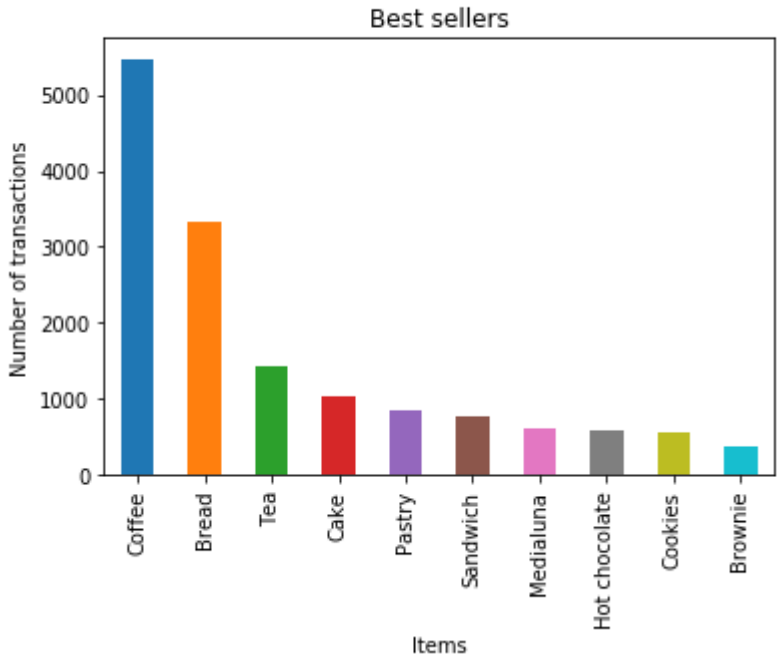
Dari 94 item yang dijual tersebut, jika diurutkan dari yang paling populer adalah sebagai berikut.

```
In [27]: data['Item'].value_counts().sort_values(ascending=False).head(10)
```

```
Out[27]: Coffee            5471
         Bread             3325
         Tea               1435
         Cake              1025
         Pastry             856
         Sandwich           771
         Medialuna          616
         Hot chocolate      590
         Cookies            540
         Brownie            379
         Name: Item, dtype: int64
```

```
In [28]: fig, ax=plt.subplots(figsize=(6,4))
         data['Item'].value_counts().sort_values(ascending=False).head(10).plot(kind='bar')
         plt.ylabel('Number of transactions')
         plt.xlabel('Items')
         ax.get_yaxis().get_major_formatter().set_scientific(False)
         plt.title('Best sellers')
```

```
Out[28]: Text(0.5, 1.0, 'Best sellers')
```

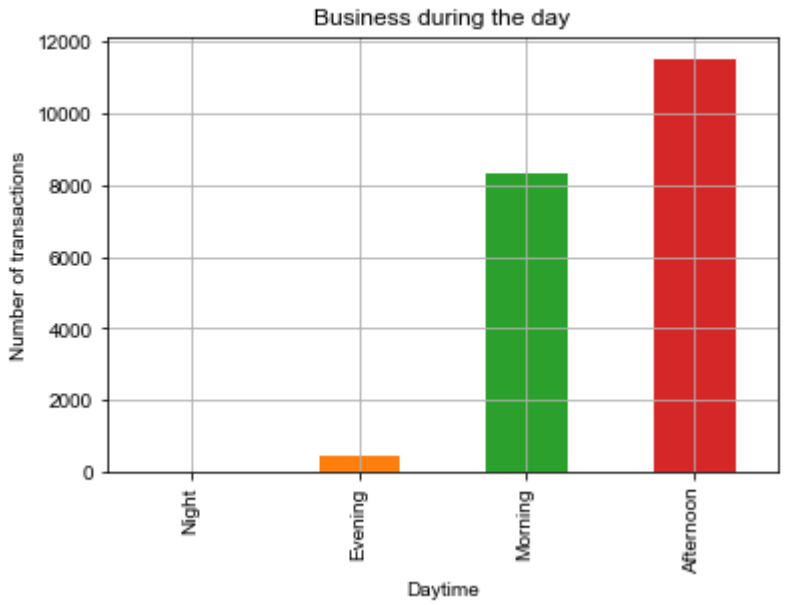


Tahap berikutnya adalah mencari waktu paling sibuk (paling banyak terjadi transaksi) dalam satu hari.

```
In [29]: data.loc[(data['Time']<'12:00:00'),'Daytime']='Morning'
         data.loc[(data['Time']>='12:00:00')&(data['Time']<'17:00:00'),'Daytime']='Afternoon'
         data.loc[(data['Time']>='17:00:00')&(data['Time']<'21:00:00'),'Daytime']='Evening'
         data.loc[(data['Time']>='21:00:00')&(data['Time']<'23:50:00'),'Daytime']='Night'
```

```
In [30]: fig, ax=plt.subplots(figsize=(6,4))
         sns.set_style('darkgrid')
         data.groupby('Daytime')['Item'].count().sort_values().plot(kind='bar')
         plt.ylabel('Number of transactions')
         ax.get_yaxis().get_major_formatter().set_scientific(False)
         plt.title('Business during the day')
```

```
Out[30]: Text(0.5, 1.0, 'Business during the day')
```



Dari grafik tersebut maka kapan waktu tersibuk?

Sedangkan untuk mengetahui item yang yang terjual pada waktu operasi toko roti adalah sebagai berikut.

```
In [31]: data.groupby('Daytime')['Item'].count().sort_values(ascending=False)
```

```
Out[31]: Daytime
         Afternoon    11569
         Morning       8404
         Evening        520
         Night           14
         Name: Item, dtype: int64
```

Untuk mengetahui jumlah transaksi per bulan maka dilakukan ekstraksi bulan dari atribut date.

In [32]:
```
data['Date_Time']=pd.to_datetime(data['Date']+' '+data['Time'])
data['Day']=data['Date_Time'].dt.day_name()
data['Month']=data['Date_Time'].dt.month
data['Month_name']=data['Date_Time'].dt.month_name()
data['Year']=data['Date_Time'].dt.year
data['Year_Month']=data['Year'].apply(str)+' '+data['Month_name'].apply(str)
data.drop(['Date','Time'], axis=1, inplace=True)

data.index=data['Date_Time']
data.index.name='Date'
data.drop(['Date_Time'],axis=1,inplace=True)
data.head()
```

Out[32]:

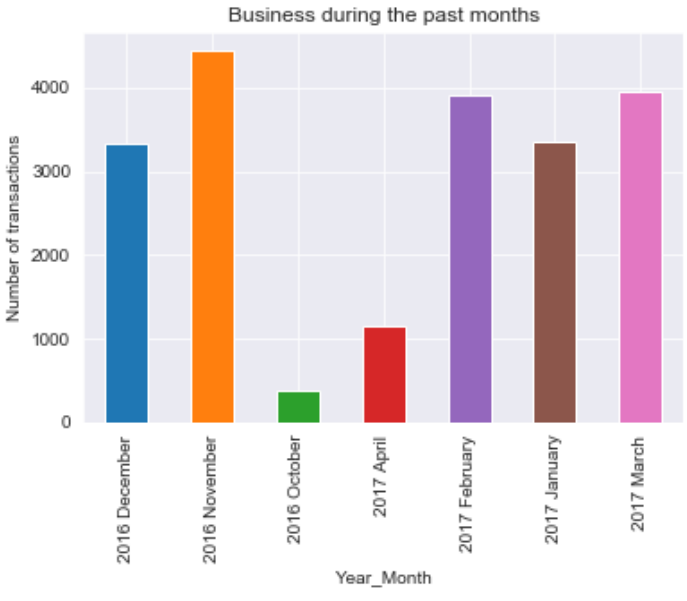
Date	Transaction	Item	Daytime	Day	Month	Month_name	Year	Year_Month
2016-10-30 09:58:11	1	Bread	Morning	Sunday	10	October	2016	2016 October
2016-10-30 10:05:34	2	Scandinavian	Morning	Sunday	10	October	2016	2016 October
2016-10-30 10:05:34	2	Scandinavian	Morning	Sunday	10	October	2016	2016 October
2016-10-30 10:07:57	3	Hot chocolate	Morning	Sunday	10	October	2016	2016 October
2016-10-30 10:07:57	3	Jam	Morning	Sunday	10	October	2016	2016 October

Berikut ini adalah grafik total transaksi yang terjadi untuk setiap bulannya.

Menurut anda, mengapa transaksi sangat sedikit pada bulan Oktober 2016 dan april 2017?

In [33]:
```
data.groupby('Year_Month')['Item'].count().plot(kind='bar')
plt.ylabel('Number of transactions')
plt.title('Business during the past months')
```

Out[33]:
```
Text(0.5, 1.0, 'Business during the past months')
```



Sedangkan untuk mengetahui best seller item pada setiap bulannya adalah sebagai berikut. Pada matrix berikut juga ditampilkan jumlah dari masing-masing item terjual pada setiap bulannya.

Bagaimana menampilkan item yang paling tidak populer?

```
In [34]: data2=data.pivot_table(index='Month_name',columns='Item', aggfunc={'Item':'count'}).fillna(0)
         data2['Max']=data2.idxmax(axis=1)
         data2
```

Out[34]:

	Item											Max
Item	Adjustment	Afternoon with the baker	Alfajores	Argentina Night	Art Tray	Bacon	Baguette	...	Valentine's card	Vegan Feast	Vegan mincepie	Victorian Sponge	
Month_name													
April	0.0	2.0	24.0	5.0	1.0	0.0	11.0		0.0	7.0	0.0	0.0	(Item, Coffee)
December	0.0	0.0	45.0	0.0	6.0	0.0	0.0		0.0	0.0	33.0	0.0	(Item, Coffee)
February	0.0	13.0	112.0	0.0	5.0	0.0	60.0		12.0	3.0	5.0	0.0	(Item, Coffee)
January	0.0	15.0	39.0	0.0	5.0	0.0	20.0		1.0	0.0	13.0	0.0	(Item, Coffee)
March	0.0	14.0	8.0	2.0	3.0	0.0	61.0		0.0	6.0	3.0	0.0	(Item, Coffee)
November	1.0	0.0	141.0	0.0	18.0	1.0	0.0		0.0	0.0	0.0	4.0	(Item, Coffee)
October	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	3.0	(Item, Coffee)

7 rows × 95 columns

Sedangkan berikut ini adalah menampilkan bestseller dalam satu harinya.

Item apa saja yang populer pada waktu operasi toko?

```
In [35]: data3=data.pivot_table(index='Daytime',columns='Item', aggfunc={'Item':'count'}).fillna(0)
         data3['Max']=data3.idxmax(axis=1)
         data3
```

Out[35]:

	Item												Max
Item	Adjustment	Afternoon with the baker	Alfajores	Argentina Night	Art Tray	Bacon	Baguette	...	Tshirt	Valentine's card	Vegan Feast	Vegan mincepie	Victorian Sponge	
Daytime														
Afternoon	0.0	19.0	245.0	3.0	31.0	1.0	67.0		0.0	4.0	4.0	30.0	5.0	(Item, Coffee)
Evening	1.0	15.0	17.0	0.0	2.0	0.0	1.0		21.0	4.0	3.0	2.0	0.0	(Item, Coffee)
Morning	0.0	10.0	107.0	4.0	5.0	0.0	84.0		0.0	3.0	2.0	22.0	2.0	(Item, Coffee)
Night	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	2.0	7.0	0.0	0.0	(Item, Vegan Feast)

4 rows × 95 columns

Berikut ini adalah untuk mencari bestseller item untuk setiap harinya.

```
In [37]: data4=data.pivot_table(index='Day',columns='Item', aggfunc={'Item':'count'}).fillna(0)
         data4['Max']=data4.idxmax(axis=1)
         data4
```

Out[37]:

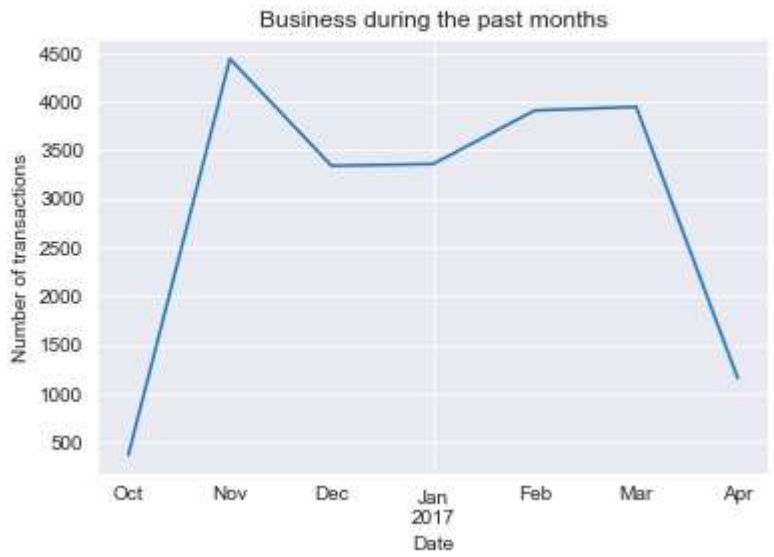
	Item											Max
Item	Adjustment	Afternoon with the baker	Alfajores	Argentina Night	Art Tray	Bacon	Baguette	...	Valentine's card	Vegan Feast	Vegan mincepie	Victorian Sponge	
Day													
Friday	0.0	7.0	59.0	0.0	4.0	0.0	21.0		1.0	1.0	9.0	0.0	(Item, Coffee)
Monday	0.0	5.0	32.0	0.0	2.0	0.0	10.0		0.0	0.0	5.0	0.0	(Item, Coffee)
Saturday	0.0	12.0	67.0	5.0	5.0	0.0	33.0		7.0	10.0	11.0	1.0	(Item, Coffee)
Sunday	0.0	14.0	70.0	1.0	7.0	1.0	21.0		0.0	2.0	9.0	4.0	(Item, Coffee)
Thursday	0.0	2.0	57.0	1.0	8.0	0.0	21.0		3.0	0.0	7.0	0.0	(Item, Coffee)
Tuesday	0.0	2.0	43.0	0.0	6.0	0.0	23.0		2.0	2.0	8.0	1.0	(Item, Coffee)
Wednesday	1.0	2.0	41.0	0.0	6.0	0.0	23.0		0.0	1.0	5.0	1.0	(Item, Coffee)

7 rows × 95 columns

Untuk menampilkan grafik jumlah transaksi yang terjadi adalah sebagai berikut.

```
In [38]: data['Item'].resample('M').count().plot()
         plt.ylabel('Number of transactions')
         plt.title('Business during the past months')
```

```
Out[38]: Text(0.5, 1.0, 'Business during the past months')
```



Assosiation Rules with Apriori Algorithm

Pada tahap ini, dilakukan analisis aturan asosiasi untuk menunjukkan hubungan (asosiasi) antar item. Hubungan antar item ini dapat membantu dalam pengambilan keputusan berdasarkan hubungan yang ada, misalnya untuk membuat strategi marketing.

Metric yang digunakan dalam aturan asosiasi adalah sebagai berikut:

- Support : It is the measure of frequency or abundance of an item in a dataset. It can be '**antecedent support**', '**consequent support**', and '**support**'.
  'antecedent support' contains proportion of transactions done for the antecedent while 'consequent support' involves those for consequent.
  'Support' is computed for both antecedent and consequent in question.
- Confidence : This gives the probability of consequent in a transaction given the presence of antecedent.
- Lift : Given that antecedents and consequents are independent, how often do they come together/bought together.
- Leverage : It is the difference between frequency of antecedent and consequent together in transactions to frequency of both in independent transactions.
- Conviction : A higher conviction score means that consequent is highly dependent on antecedent.

Algoritma apriori digunakan untuk ekstraksi itemset yang sering muncul (frequent) untuk digunakan dalam analisis aturan asosiasi. Pada algoritma ini, user memerlukan untuk mendefiniskan nilai minimum untuk support. Nilai ini adalah batas minimal itemset dinyatakan sebagai "frequent".

Pada python telah disediakan package/library yang mendukung proses analisis association rule.

```
In [39]: from mlxtend.preprocessing import TransactionEncoder
         from mlxtend.frequent_patterns import apriori, association_rules
```

***Catatan: jika tidak terdapat module "mlxtend", lihat catatan di akhir modul ini.**

Tahap pertama adalah membuat dataset yang berisi daftar item-item yang dibeli secara bersamaan.

```
In [46]: lst=[]
         for item in data['Transaction'].unique():
             lst2=list(set(data[data['Transaction']==item]['Item']))
             if len(lst2)>0:
                 lst.append(lst2)
         print(lst[0:3])
         print(len(lst))
```

```
[['Bread'], ['Scandinavian'], ['Cookies', 'Jam', 'Hot chocolate']]
9465
```

Agar dapat diproses oleh algoritma apriori, maka dataset tersebut perlu di-encoded. Hal ini bisa dilakukan dengan menggunakan TransactionEncoder. Diikuti oleh algoritma apriori untuk mendapatkan frequent itemset. Kemudian dianalisis dengan metric yang digunakan pada assosiation rule analysis.

In [47]:
```
te=TransactionEncoder()
te_data=te.fit(lst).transform(lst)
data_x=pd.DataFrame(te_data,columns=te.columns_)
print(data_x.head())

frequent_items= apriori(data_x, use_colnames=True, min_support=0.03)
print(frequent_items.head())

rules = association_rules(frequent_items, metric="lift", min_threshold=1)
rules.antecedents = rules.antecedents.apply(lambda x: next(iter(x)))
rules.consequents = rules.consequents.apply(lambda x: next(iter(x)))
rules
```

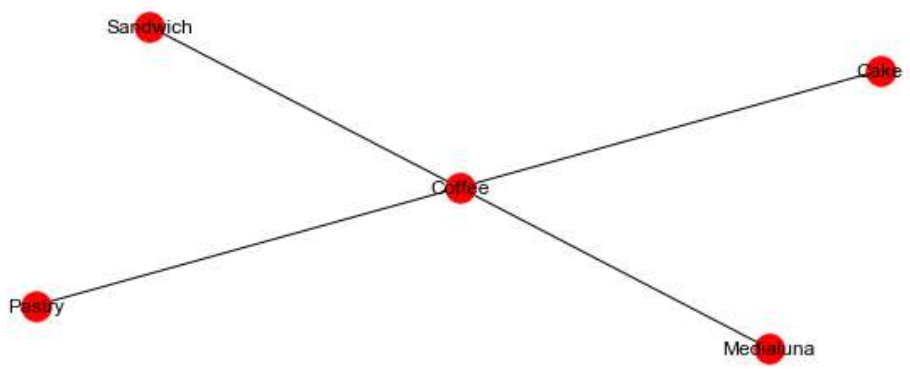
```
[5 rows x 94 columns]
    support     itemsets
0  0.036344  (Alfajores)
1  0.327205      (Bread)
2  0.040042    (Brownie)
3  0.103856       (Cake)
4  0.478394     (Coffee)
```

Out[47]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	Coffee	Cake	0.478394	0.103856	0.054728	0.114399	1.101515	0.005044	1.011905
1	Cake	Coffee	0.103856	0.478394	0.054728	0.526958	1.101515	0.005044	1.102664
2	Coffee	Medialuna	0.478394	0.061807	0.035182	0.073542	1.189878	0.005614	1.012667
3	Medialuna	Coffee	0.061807	0.478394	0.035182	0.569231	1.189878	0.005614	1.210871
4	Coffee	Pastry	0.478394	0.086107	0.047544	0.099382	1.154168	0.006351	1.014740
5	Pastry	Coffee	0.086107	0.478394	0.047544	0.552147	1.154168	0.006351	1.164682
6	Coffee	Sandwich	0.478394	0.071844	0.038246	0.079947	1.112792	0.003877	1.008807
7	Sandwich	Coffee	0.071844	0.478394	0.038246	0.532353	1.112792	0.003877	1.115384

Hubungan antar item tersebut dapat digambarkan dalam bentuk jaringan (network). Visualisasi network ini dapat menggunakan NetworkX (Python package yang digunakan untuk menghasilkan complex network). Pada network tersebut dapat dilihat hubungan antara antecedents dan consequents yang didapat dari aturan asosiasi.

In [48]:
```
fig, ax=plt.subplots(figsize=(10,4))
GA=nx.from_pandas_edgelist(rules,source='antecedents',target='consequents')
nx.draw(GA,with_labels=True)
plt.show()
```



Dari network tersebut dapat dilihat bahwa coffee adalah item yang memiliki hubungan dengan pastry, cake, medialuna dan sandwich. Sehingga dapat dinyatakan bahwa jika seseorang membeli salah satu atau beberapa dari 4 item tersebut maka kemungkinan besar seseorang tersebut akan membeli coffee. Apakah hal ini akan berlaku sebaliknya? Belum tentu, perlu dicek kembali confidence-nya.

Kesimpulan

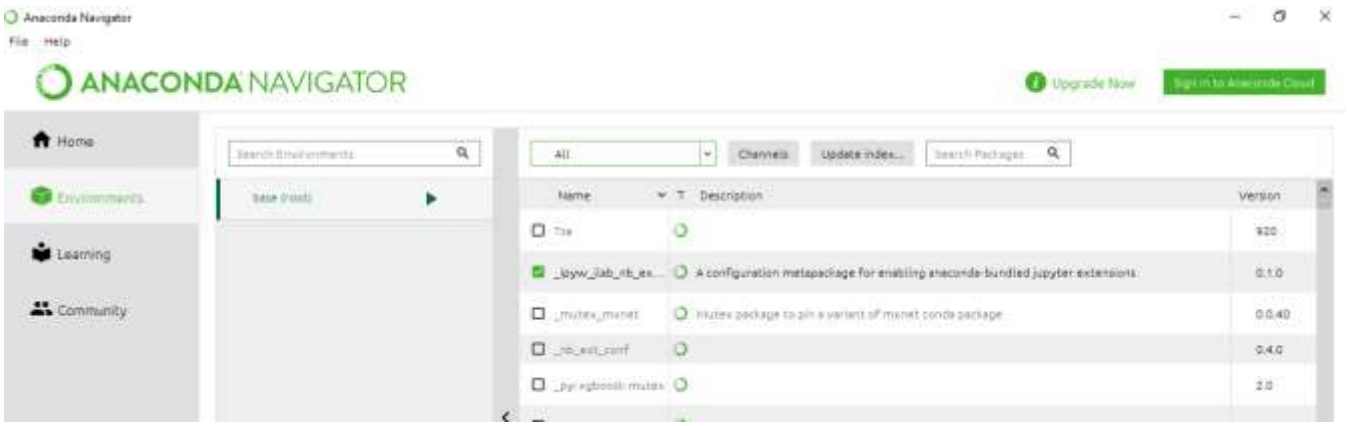
Pada toko ini, coffee merupakan item paling populer dan item ini berasosiasi dengan 4 item: pastry, cake, medialuna and sandwich. Dari hubungan ini maka dapat dibuat beberapa strategi yang membantu meningkatkan penjualan toko, antara lain sebagai berikut:

- Untuk setiap pembelian coffee, maka dapat diberikan diskon/promo untuk pembelian item kedua (sebaiknya item yang memiliki kemungkinan kuat dengan coffee).
- Menempatkan 4 item tersebut dekat dengan counter pemesanan coffee dapat menarik pelanggan untuk memesan item tersebut.
- Itemsets dengan lift lebih dari 1, misalnya {Hot chocolate, Cookies}, {Sandwhich, Sandwhich}, {Hot Chocolate, Cake}, {Coffee, Toast}. Pemberian diskon/promo untuk bundling itemset tersebut dapat membantu menarik minat pelanggan.

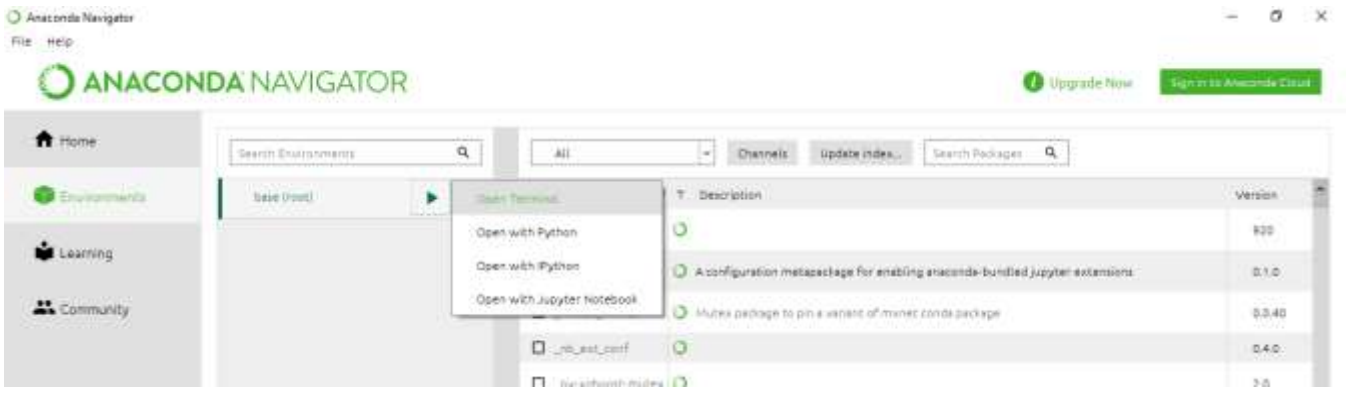
Module Not Found

Jika tidak ditemukan package yang dibutuhkan (*error module not found*), maka lakukan instalasi package yang dibutuhkan. Salah satu caranya adalah dengan melakukan instalasi melalui anaconda navigator, yang tahapannya sebagai berikut:

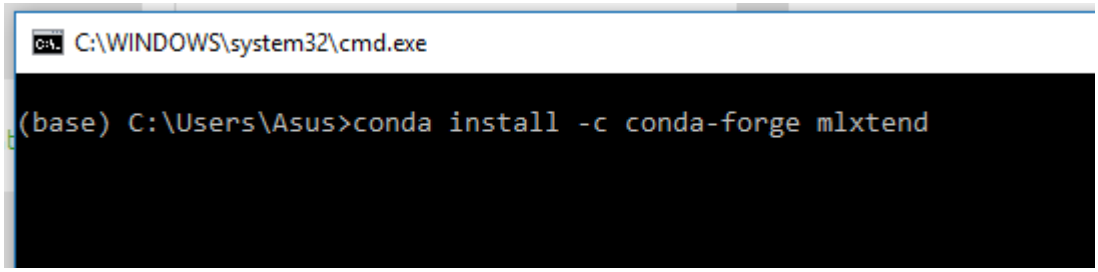
Buka anaconda navigator, dan lakukan pencarian pada field search untuk semua (All) module -> terinstall maupun tidak.



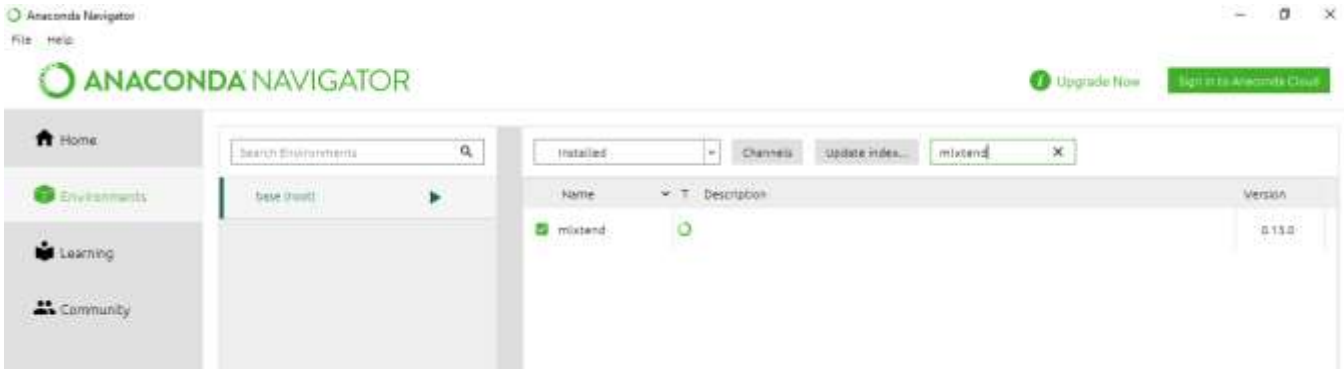
Jika tidak ditemukan maka lakukan instalasi melalui command prompt, dengan cara klik environment yang aktif. Kemudian pilih Open Terminal.



Jika terbuka command prompt, maka lakukan instalasi dengan perintah conda. Perintah berikut untuk melakukan instalasi package “mlxtend” yang digunakan untuk algoritma apriori – assosiation rules.



Jika telah berhasil terinstall, maka akan muncul pada list package yang telah terinstall.



Namun jika tidak muncul, maka pilih Update Index yang berada di sebelah field pencarian.