```
Python 2.7.15 |Anaconda, Inc.| (default, Dec 10 2018, 21:57:18) [MSC v.1500 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 5.8.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: import numpy as np
   ...: import pandas as pd
   ...: import scipy
   ...: from scipy.sparse import csr_matrix
   ...: from scipy.sparse import csc_matrix
   ...:
   ...: from pandas.api.types import CategoricalDtype
   ...: from mlxtend.frequent_patterns import apriori
   ...:
   ...:
   ...: file1 = 'mbatest.txt'
   ...:
   ...: frame = pd.read_csv(file1, sep=",",nrows=5000000)
   ...: frame['QTY'] = 1
   ...:
   ...: trx_id  = frame['TRX_ID']
   ...: product = frame['PRODUCT']
   ...: qty     = frame['QTY']
   ...:
   ...: TRX_ID_u   = sorted(frame.TRX_ID.unique())
   ...: PRODUCT_u  = sorted(frame.PRODUCT.unique())
   ...:
   ...:
   ...: TRX_ID_c   = CategoricalDtype(sorted(frame.TRX_ID.unique()), ordered=True)
   ...: PRODUCT_c  = CategoricalDtype(sorted(frame.PRODUCT.unique()), ordered=True)
   ...:
   ...: row = frame.TRX_ID.astype(TRX_ID_c).cat.codes
   ...: col = frame.PRODUCT.astype(PRODUCT_c).cat.codes
   ...: sparse_matrix = csr_matrix((frame["QTY"], (row, col)), \
   ...:                            shape=(TRX_ID_c.categories.size, PRODUCT_c.categories.size))
   ...:
   ...: coo1 = sparse_matrix.tocoo(copy=True)
   ...:
   ...: df = pd.SparseDataFrame(coo1, \
   ...:                         index=TRX_ID_c.categories, \
   ...:                         columns=PRODUCT_c.categories, \
   ...:                         default_fill_value=0)

In [2]: type(coo1)
Out[2]: scipy.sparse.coo.coo_matrix

In [3]: type(df)
Out[3]: pandas.core.sparse.frame.SparseDataFrame

In [4]: df
Out[4]:
    1   2   3   4   5   6   7   8   9   10  18  26
1    1   1   0   0   0   0   0   0   0   0   0   0
2    0   1   0   0   0   0   0   0   0   0   0   0
3    1   0   1   1   0   0   0   0   0   0   0   0
4    0   1   0   0   0   0   0   0   0   0   0   0
5    1   0   0   1   0   0   0   0   0   0   0   1
6    0   1   1   0   0   0   0   0   0   0   1   0
7    0   0   0   1   1   1   0   0   0   0   0   0
8    1   0   1   0   0   1   1   0   1   0   0   0
9    0   0   1   0   0   0   1   1   1   0   0   0
10   1   0   1   0   0   0   1   0   0   1   0   0

In [5]: df.shape
Out[5]: (10, 12)

In [6]: df.dtypes
   ...:
Out[6]:
1     Sparse[int64, 0]
2     Sparse[int64, 0]
3     Sparse[int64, 0]
4     Sparse[int64, 0]
5     Sparse[int64, 0]
6     Sparse[int64, 0]
7     Sparse[int64, 0]
8     Sparse[int64, 0]
```

```
9     Sparse[int64, 0]
10    Sparse[int64, 0]
18    Sparse[int64, 0]
26    Sparse[int64, 0]
dtype: object

In [7]: import numpy as np
   ...: import pandas as pd
   ...: import scipy
   ...: from scipy.sparse import csr_matrix
   ...: from scipy.sparse import csc_matrix
   ...:
   ...: from pandas.api.types import CategoricalDtype
   ...: from mlxtend.frequent_patterns import apriori
   ...:
   ...:
   ...: file1 = 'mbatest.txt'
   ...:
   ...: frame = pd.read_csv(file1, sep=",",nrows=5000000)
   ...: frame['QTY'] = 1
   ...:
   ...: trx_id  = frame['TRX_ID']
   ...: product = frame['PRODUCT']
   ...: qty     = frame['QTY']
   ...:
   ...: TRX_ID_u   = sorted(frame.TRX_ID.unique())
   ...: PRODUCT_u  = sorted(frame.PRODUCT.unique())
   ...:
   ...:
   ...: TRX_ID_c   = CategoricalDtype(sorted(frame.TRX_ID.unique()), ordered=True)
   ...: PRODUCT_c  = CategoricalDtype(sorted(frame.PRODUCT.unique()), ordered=True)
   ...:
   ...: row = frame.TRX_ID.astype(TRX_ID_c).cat.codes
   ...: col = frame.PRODUCT.astype(PRODUCT_c).cat.codes
   ...: sparse_matrix = csr_matrix((frame["QTY"], (row, col)), \
   ...:                            shape=(TRX_ID_c.categories.size, PRODUCT_c.categories.size))
   ...:
   ...: coo1 = sparse_matrix.tocoo(copy=True)
   ...:
   ...: df = pd.SparseDataFrame(coo1, \
   ...:                         index=TRX_ID_c.categories, \
   ...:                         columns=PRODUCT_c.categories, \
   ...:                         default_fill_value=0)
   ...:
   ...: frequent_itemsets = apriori(df, min_support=0.5, use_colnames=True)
Traceback (most recent call last):

  File "<ipython-input-7-6187321764fa>", line 35, in <module>
    frequent_itemsets = apriori(df, min_support=0.5, use_colnames=True)

  File "C:\Users\nshah\Anaconda2\envs\mba\lib\site-packages\mlxtend\frequent_patterns\apriori.py", line 115, in apriori
    X = df.to_coo().tocsc()

  File "C:\Users\nshah\Anaconda2\envs\mba\lib\site-packages\pandas\core\sparse\frame.py", line 275, in to_coo
    dtype = find_common_type(self.dtypes)

  File "C:\Users\nshah\Anaconda2\envs\mba\lib\site-packages\pandas\core\dtypes\cast.py", line 1093, in find_common_type
    first = types[0]

  File "C:\Users\nshah\Anaconda2\envs\mba\lib\site-packages\pandas\core\series.py", line 868, in __getitem__
    result = self.index.get_value(self, key)

  File "C:\Users\nshah\Anaconda2\envs\mba\lib\site-packages\pandas\core\indexes\base.py", line 4374, in get_value
    tz=getattr(series.dtype, 'tz', None))

  File "pandas\_libs\index.pyx", line 81, in pandas._libs.index.IndexEngine.get_value

  File "pandas\_libs\index.pyx", line 89, in pandas._libs.index.IndexEngine.get_value

  File "pandas\_libs\index.pyx", line 132, in pandas._libs.index.IndexEngine.get_loc

  File "pandas\_libs\hashtable_class_helper.pxi", line 987, in pandas._libs.hashtable.Int64HashTable.get_item

  File "pandas\_libs\hashtable_class_helper.pxi", line 993, in pandas._libs.hashtable.Int64HashTable.get_item

KeyError: 0L


In [8]:

In [8]:
```