In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import sqlite3
```

In [2]:

```
connection = sqlite3.connect('D:\\AI_stuff\\dataset\\Amazon food reviews\\database.sqli
te')

food_reviews = pd.read_sql_query("select * from reviews",connection)

food_reviews.head()
```

Out[2]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	Helpfu
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0

In [3]:

```
food_reviews = pd.read_sql_query('select * from reviews where Score != 3',connection)

pos_neg = list(map(lambda x : 'Positive' if x > 3 else 'Negative',food_reviews['Score'
]))

food_reviews['Score'] = pos_neg

food_reviews.head()
```

Out[3]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	Helpfu
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0

In [4]:

```
sorted_data = food_reviews.sort_values('ProductId',axis = 0 ,inplace = False, ascending
 = True)

remove_duplicates = sorted_data.drop_duplicates(subset = {'UserId' , 'ProfileName' , 'T
ime' , 'Text'} , keep = 'first' , inplace = False)

remove_duplicates.shape
```

Out[4]:

```
(364173, 10)
```

In [5]:

```
remove_duplicates.isnull().sum()
```

Out[5]:

```
Id                        0
ProductId                 0
UserId                    0
ProfileName               0
HelpfulnessNumerator      0
HelpfulnessDenominator    0
Score                     0
Time                      0
Summary                   0
Text                      0
dtype: int64
```

In [6]:

```
final_reviews = remove_duplicates[remove_duplicates['HelpfulnessNumerator'] <= remove_d
uplicates['HelpfulnessDenominator']]

final_reviews.shape
```

Out[6]:

```
(364171, 10)
```

In [7]:

```
final_reviews_score = final_reviews['Score']

final_reviews_data  = final_reviews.drop('Score', axis=1)

final_reviews_data.shape
```

Out[7]:

```
(364171, 9)
```

In [8]:

```
from sklearn.feature_extraction.text import CountVectorizer

final_data = CountVectorizer().fit(final_reviews_data['Text'].values).transform(final_r
eviews_data['Text'].values)

final_data.shape
```

Out[8]:

```
(364171, 115281)
```

In [10]:

```
from sklearn.manifold import TSNE
import seaborn as sn
model = TSNE(n_components = 2 ,random_state = 0, perplexity = 50 , n_iter = 5000)

tsne_final = final_data[0:10000,:]

tsne_final_ma = tsne_final.toarray()

tsne_labels = final_reviews_score[0:10000]

tsne_data = model.fit_transform(tsne_final_ma)

tsne_data = np.vstack((tsne_data.T,tsne_labels)).T

tsne = pd.DataFrame(tsne_data, columns = ('dim1','dim2','labels'))

(sn.FacetGrid(tsne,hue = 'labels',size = 6).map(plt.scatter,'dim1','dim2').add_legend
())
```

```
---------------------------------------------------------------------------
-
MemoryError                               Traceback (most recent call las
t)
<ipython-input-10-3bf1fd5c5b30> in <module>()
      9 tsne_labels = final_reviews_score[0:10000]
     10 
---> 11 tsne_data = model.fit_transform(tsne_final_ma)
     12 
     13 tsne_data = np.vstack((tsne_data.T,tsne_labels)).T

D:\Anaconda\lib\site-packages\sklearn\manifold\t_sne.py in fit_transform(s
elf, X, y)
    856             Embedding of the training data in low-dimensional spac
e.
    857         """
--> 858         embedding = self._fit(X)
    859         self.embedding_ = embedding
    860         return self.embedding_

D:\Anaconda\lib\site-packages\sklearn\manifold\t_sne.py in _fit(self, X, s
kip_num_points)
    715                                     metric=self.metric)
    716             t0 = time()
--> 717             knn.fit(X)
    718             duration = time() - t0
    719             if self.verbose:

D:\Anaconda\lib\site-packages\sklearn\neighbors\base.py in fit(self, X, y)
    801             or [n_samples, n_samples] if metric='precomputed'.
    802         """
--> 803         return self._fit(X)

D:\Anaconda\lib\site-packages\sklearn\neighbors\base.py in _fit(self, X)
    246             self._tree = KDTree(X, self.leaf_size,
    247                                 metric=self.effective_metric_,
--> 248                                 **self.effective_metric_params_)
    249         elif self._fit_method == 'brute':
    250             self._tree = None

sklearn\neighbors\binary_tree.pxi in sklearn.neighbors.kd_tree.BinaryTree.
__init__()

D:\Anaconda\lib\site-packages\numpy\core\numeric.py in asarray(a, dtype, o
rder)
    490 
    491     """
--> 492     return array(a, dtype, copy=False, order=order)
    493 
    494 

MemoryError: 
```