In [1]:

```
import pandas as pd
import numpy as np
import google.cloud.logging
from google.cloud import bigquery
from argparse import ArgumentParser
from time import sleep
from datetime import datetime
from pytz import timezone
from random import randint
import random
from tqdm import tqdm
from gensim.models import Word2Vec
import matplotlib.pyplot as plt
%matplotlib inline
import warnings;
warnings.filterwarnings('ignore')
```

In [2]:

```
%%bigquery df
SELECT  query_v2.query_key as CustomerID,   query_v2.raw,  query_v2.ordered,  query_v2.ordered_stem_wo_stop_words as stem_stop,
        item.id as StockCode, count(*) as freq , item.title

FROM relatedsearch.dim_cat_trimmed
inner join  relatedsearch.item on dim_cat_trimmed.category_code = item.cat
inner join  relatedsearch.query_source_mapping_v2 on query_source_mapping_v2.source_key = item.key
inner join  relatedsearch.query_v2 on query_v2.query_key = query_source_mapping_v2.query_key

where fourth_sub_category  = "Tablet" and
    DATE(dim_cat_trimmed.source_time) between "2018-01-20" and "2020-07-28" and
    DATE(item.source_time) between "2018-01-20" and "2020-07-28" and
    DATE(query_source_mapping_v2.source_time) between "2018-01-20" and "2020-07-28" and
    DATE(query_v2.source_time) between "2018-01-20" and "2020-07-28"

group by  query_v2.query_key,  item.id, query_v2.raw,  query_v2.ordered,  query_v2.ordered_stem_wo_stop_words, item.title

order by item.id
```

In [3]:

```
df.head()
```

Out[3]:

	CustomerID	raw	ordered	stem_stop	StockCode	freq	title
0	2UEnXzvG18mt6DUwaa7ecmlw1AqZ7L0ZkXr2mk/eBxl=	ucuz tablet	ucuz tablet	ucuz tablet	165786711	2	CODEGEN+QBIX+7KAPASİTİF+EKRAN+ARM+CORTEX+A10+1...
1	k8NdVRWT3FgGtLj4ErfH9mcocsWzmQHU6kb1fkMuPog=	en ucuz tabletler	en ucuz tabletler	en ucuz tablet	165786711	1	CODEGEN+QBIX+7KAPASİTİF+EKRAN+ARM+CORTEX+A10+1...
2	W+2FvWmlUCqPZFpYoF6Xn5S963e5Vxmc6EKeD/vAElc=	tablet	tablet	tablet	165786711	7	CODEGEN+QBIX+7KAPASİTİF+EKRAN+ARM+CORTEX+A10+1...
3	W+2FvWmlUCqPZFpYoF6Xn5S963e5Vxmc6EKeD/vAElc=	tablet	tablet	tablet	166429944	1	Dji Phantom 3 standard için Taşıma Çantası
4	niW5+dzcTvXhZF49NpFskturkVlwHfM55nuLYCMK/G8=	Çaycı	çaycı	çaycı	181456675	1	ÇAYCI+TELSİZLERİ+1+MERKEZ+16+ŞUBE+HAZIR+SİSTEM

In [4]:

```
df['StockCode']= df['StockCode'].astype(str)
```

In [5]:

```
# customer ID'ler
customers = df["CustomerID"].unique().tolist()
len(customers)
```

Out[5]:

```
18274
```

In [6]:

```
# ID lerin %90 nini train olarak olabiliriz %10 unu da validation olarak alalım.
customers_train = [customers[i] for i in range(round(0.9*len(customers)))]

# data yı train and validation set olarak split edelim
train_df = df[df['CustomerID'].isin(customers_train)]
validation_df = df[~df['CustomerID'].isin(customers_train)]
```

In [7]:

```
purchases_train = []

# ürün train
for i in tqdm(customers_train):
    temp = train_df[train_df["CustomerID"] == i]["StockCode"].tolist()
    purchases_train.append(temp)
```

```
100%|██████████| 16447/16447 [02:08<00:00, 128.15it/s]
```

In [8]:

```
customers_train[0]
```

Out[8]:

```
'2UEnXzvG18mt6DUwaa7ecmlw1AqZ7L0ZkXr2mk/eBxI='
```

In [9]:

```
purchases_train[0][0]
```

Out[9]:

```
'165786711'
```

In [10]:

```
purchases_val = []

# validation grubunu oluşturalım (test/kontrol gurubu)
for i in tqdm(validation_df['CustomerID'].unique()):
    temp = validation_df[validation_df["CustomerID"] == i]["StockCode"].tolist()
    purchases_val.append(temp)
```

```
100%|██████████| 1827/1827 [00:01<00:00, 1335.95it/s]
```

In [11]:

```
#  word2vec model i purchases_train ile eğitelim
model = Word2Vec(window = 10, sg = 1, hs = 0,
                 negative = 10, # for negative sampling
                 alpha=0.03, min_alpha=0.0007,
                 seed = 14)

model.build_vocab(purchases_train, progress_per=200)

model.train(purchases_train, total_examples = model.corpus_count,
            epochs=10, report_delay=1)
```

Out[11]:

```
(1033067, 1161980)
```

In [12]:

```
# word2vec modelini kaydedelim
model.save("word2vec_2.model")
```

In [13]:

```
model.init_sims(replace=True)
```

In [14]:

```
print(model)
```

```
Word2Vec(vocab=6960, size=100, alpha=0.03)
```

In [15]:

```
X = model[model.wv.vocab]

X.shape
```

Out[15]:

```
(6960, 100)
```

In [16]:

```
X
```

Out[16]:

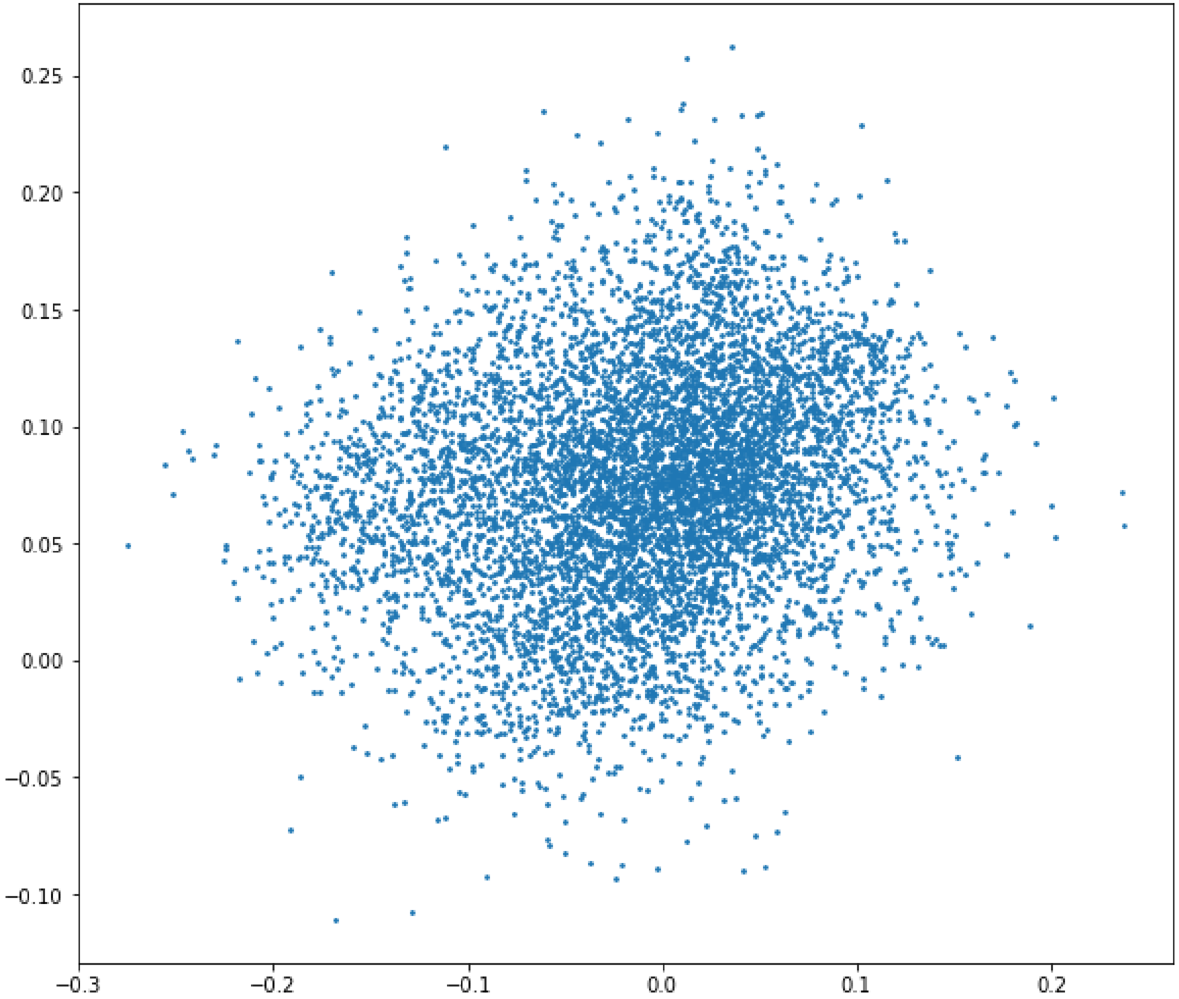
```
array([[-0.0404036 ,  0.10059489,  0.16308855, ..., -0.13459189,
         0.05625929,  0.05255076],
       [-0.01316117,  0.07043511,  0.08151367, ..., -0.0891676 ,
         0.12591262,  0.0561089 ],
       [-0.1083677 ,  0.07628665,  0.17567293, ..., -0.17106038,
         0.10151844,  0.10466572],
       ...,
       [ 0.01919419,  0.0482697 ,  0.08211095, ..., -0.04563933,
         0.09890724,  0.03009988],
       [ 0.03389419,  0.16034812,  0.03468767, ..., -0.05956305,
         0.23232919, -0.0017047 ],
       [-0.0463897 ,  0.05040234,  0.0408004 , ..., -0.15277074,
        -0.12558283,  0.15816087]], dtype=float32)
```

In [17]:

```
plt.figure(figsize=(10,9))
plt.scatter(X[:, 0], X[:, 1], s=3, cmap='Spectral')
```

```
Out[17]:

<matplotlib.collections.PathCollection at 0x7f13e0484850>
```


0.25
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.3
−0.2
−0.1
0.0
0.1
0.2

In [18]:

```
products = train_df[["StockCode", "ordered"]]

products_dict = products.groupby('StockCode')['ordered'].apply(list).to_dict()
# groupby ile StockCode un frekans kelimelerini oluşturdum
```

In [19]:

```
products
```

Out[19]:

	StockCode	ordered
0	165786711	ucuz tablet
1	165786711	en ucuz tabletler
2	165786711	tablet
3	166429944	tablet
4	181456675	çaycı
...	...	...
118578	580673686	general mobile e tab 5 ekran
118579	580673686	general mobile e tab 5
118580	580705948	ipad 2 el
118581	580705948	ipad air 2
118582	580714545	2 el tablet

116198 rows × 2 columns

In [20]:

```
products_dict['165786711']
```

Out[20]:

```
['ucuz tablet', 'en ucuz tabletler', 'tablet']
```

In [21]:

```
def similar_products(v, n = 9):

    # benzer ürünleri modelden al
    ms = model.similar_by_vector(v, topn= n+1)[0:]

    # ürünlerin similarity score al
    new_ms = []

    for j in ms:
        pair = (j[0], j[1],  products_dict[j[0]][0] )
        new_ms.append(pair)

    return new_ms
```

In [22]:

```
similar_products(model['555864992'])
```

Out[22]:

```
[('555864992', 1.0, 'galaxy tab'),
 ('556782532', 0.9728291034698486, 'samsung t510'),
 ('554374345', 0.9635255932807922, 'samsung t510'),
 ('554374351', 0.9539943337440491, 'samsung galaxy tab a 10 1 2019'),
 ('557320325', 0.9528002738952637, 'tablet bilgisayar'),
 ('557894635', 0.9515163898468018, 'samsung galaxy tab sm t510 32 gb 10 1'),
 ('557320330', 0.9500331878662109, 'samsung galaxy tab a sm t510 10 1'),
 ('553226927', 0.9476763010025024, 'tablet'),
 ('555963946', 0.945946216583252, 'samsung tablet'),
 ('554374350', 0.9435567855834961, 'samsung galaxy tab a sm t510')]
```

In [23]:

```
# kullanıcı benzerliği için yeni bir fonksiyon oluşturalım
def aggregate_vectors(products):
    product_vec = []
    for i in products:
        try:
            product_vec.append(model[i])
        except KeyError:
            continue

    return np.mean(product_vec, axis=0)
```

In [24]:

```
aggregate_vectors(purchases_val[0]).shape
```

Out[24]:

```
(100,)
```

In [25]:

```
products_dict['554262381']
```

Out[25]:

```
['ipad',
 'tablet',
 'ipad 7',
 'apple',
 'ipad',
 'ipad',
 'apple ipad',
 'apple tablet',
 'ipad 2',
 'ipad 32 gb']
```

In [26]:

```
similar_products(aggregate_vectors(['554262381']))
```

Out[26]:

```
[('554262381', 1.0000001192092896, 'ipad'),
 ('554374340', 0.9805550575256348, 'ipad'),
 ('554093212', 0.980064332485199, 'ipad 7 nesil'),
 ('554374334', 0.9762858152389526, 'apple ipad 7 nesil tablet'),
 ('554349803', 0.9748654961585999, 'apple ipad pro'),
 ('553481120', 0.9735358953475952, 'ipad 32 gb'),
 ('553610297', 0.9711045026779175, 'apple ipad 7 10 2'),
 ('555001334', 0.9691473841667175, 'teşhir tablet'),
 ('554319160', 0.9682694673538208, 'apple ipad 7'),
 ('554309988', 0.966841459274292, 'apple ipad')]
```