Project Description (Topic model for review data):

“””

DESCRIPTION

Help a leading mobile brand understand the voice of the customer by analyzing the reviews of their product on Amazon and the topics that customers are talking about. You will perform topic modeling on specific parts of speech. You’ll finally interpret the emerging topics.

Problem Statement:

A popular mobile phone brand, Lenovo has launched their budget smartphone in the Indian market. The client wants to understand the VOC (voice of the customer) on the product. This will be useful to not just evaluate the current product, but to also get some direction for developing the product pipeline. The client is particularly interested in the different aspects that customers care about. Product reviews by customers on a leading e-commerce site should provide a good view.

Domain: Amazon reviews for a leading phone brand

Analysis to be done: POS tagging, topic modeling using LDA, and topic interpretation

Content:

Dataset: ‘K8 Reviews v0.2.csv’

Columns:

Sentiment: The sentiment against the review (4,5 star reviews are positive, 1,2 are negative)

Reviews: The main text of the review

Steps to perform:

Discover the topics in the reviews and present it to business in a consumable format. Employ techniques in syntactic processing and topic modeling.

Perform specific cleanup, POS tagging, and restricting to relevant POS tags, then, perform topic modeling using LDA. Finally, give business-friendly names to the topics and make a table for business.

Tasks:

1. Read the .csv file using Pandas. Take a look at the top few records.
2. Normalize casings for the review text and extract the text into a list for easier manipulation.
3. Tokenize the reviews using NLTKs word_tokenize function.
4. Perform parts-of-speech tagging on each sentence using the NLTK POS tagger.

5. For the topic model, we should want to include only nouns.
    1. Find out all the POS tags that correspond to nouns.
    2. Limit the data to only terms with these tags.
6. Lemmatize.
    1. Different forms of the terms need to be treated as one.
    2. No need to provide POS tag to lemmatizer for now.
7. Remove stopwords and punctuation (if there are any).
8. Create a topic model using LDA on the cleaned-up data with 12 topics.
    1. Print out the top terms for each topic.
    2. What is the coherence of the model with the c_v metric?
9. Analyze the topics through the business lens.
    1. Determine which of the topics can be combined.
10. Create a topic model using LDA with what you think is the optimal number of topics
    1. What is the coherence of the model?
11. The business should be able to interpret the topics.
    1. Name each of the identified topics.
    2. Create a table with the topic name and the top 10 terms in each to present to the business.

“””

Source Code:

```
# Import required libraries, functions and classes


#Numpy and pandas for dataframes

import numpy as np

import pandas as pd


# nltk library for tokenization, lemmatizer, stopwords, pos tags and FreqDist

# import string for punctuation and str manipulations

import nltk

from nltk.tokenize import word_tokenize,TweetTokenizer

from nltk.tag import pos_tag

from nltk.stem import WordNetLemmatizer

from nltk import FreqDist

from nltk.corpus import stopwords

import string


#Gensim library for LDA model creation . Corpora in gensim to create the id2word Dictionary and corpus of terms

import gensim

import gensim.corpora as corpora
```

```
from gensim.models import CoherenceModel


#visualization using matplotlib and pyLDAvis for the LDA model viz

import matplotlib.pyplot as plt

import pyLDAvis

import pyLDAvis.gensim_models

from pprint import pprint

#import warnings to ignore deprecation warning

import warnings

warnings.filterwarnings("ignore", category = DeprecationWarning)


if __name__=='__main__':

#Task 1 : Read the .csv file using Pandas. Take a look at the top few records.

  print("reading csv file:")

  file_path=input("enter path for the loan data file to load:")

  df_path=file_path.replace("\\",'/')


  reviews_df = pd.read_csv(df_path)


  print(reviews_df.head()) #look at the top few records

  print("---------------------------------------------------------------------")
```

#Task 2: Normalize casings for the review text and extract the text into a list for easier manipulation

```
print("Normalize the text - reduce to lower case")

review_list = [review.lower() for review in reviews_df["review"]]

print(review_list[:5])

print("-------------------------------------------------------------------------")
```

#Task 3:Tokenize the reviews using NLTKs word_tokenize function.

```
print("Tokenize the reviews:")


rev_words = [word_tokenize(review) for review in review_list]

print(rev_words[:5])

print("------------------------------------------------------------------------")
```

#Task 4: Perform parts-of-speech tagging on each sentence using the NLTK POS tagger.

```
print("POS tagging using NLTK pos tagger:")

pos_tagged_review = [pos_tag(review) for review in rev_words]

print(len(pos_tagged_review))

print(pos_tagged_review[:5])

print("------------------------------------------------------------------------")
```

#Task 5: For the topic model, we should want to include only nouns.

#1. Find out all the POS tags that correspond to nouns.

#2. Limit the data to only terms with these tags.

```
print("Find out all the POS tags that correspond to nouns - Since pos_tag function in NLTK library uses the Penn Treebank tagset.")

pos_noun_reviews = []
for review in pos_tagged_review:
  nouns=[]
  for word_tuple in review:
    if "NN" in word_tuple[1]:
      nouns.append(word_tuple)
  pos_noun_reviews.append(nouns)

print(pos_noun_reviews[:50])
```

Exclude any reviews that did not have any nouns as these reviews will be blank or empty sublists []

```
print("Limit the data to only terms with noun tags")
pos_noun_reviews=[review for review in pos_noun_reviews if len(review)>=1]
print(len(pos_noun_reviews), pos_noun_reviews[:50])
```

```
  print("------------------------------------------------------------------------")

#Task 6: Lemmatize.

#1. Different forms of the terms need to be treated as one.
#2. No need to provide POS tag to lemmatizer for now

  print("Lemmatize the different forms of the nouns")
# POS tags not passed to lemmatizer

  wnl = WordNetLemmatizer()
  lemmatized_words =[]
  for review in pos_noun_reviews:
    lemma_word=[]
    for word in review:
      lemma_word.append(wnl.lemmatize(word[0]))
    lemmatized_words.append(lemma_word)

  print(lemmatized_words[:50])

#Task 7: Remove stopwords and punctuation (if there are any).
```

```
  print("Remove stopwords and punctuation (if there are any)")

# The o/p from lemmatizer still has many composite words that still contain emojis , special characters etc.

# Using tweet tokenizer for isolating them better.

  tweet_tokenize = TweetTokenizer()



#Create list of stopwords with punctuations.Manually added token ['\s'] as this is usually seperated in tokenize

  stop_words = stopwords.words("english")

  stop_words = stop_words+list(string.punctuation)+["\'s"]



  filtered_rev_words=[]



  for review in lemmatized_words:

    filter_words=[]

    for words in review:

      rev_words = []

      rev_words = tweet_tokenize.tokenize(words)

      for word in rev_words:

        if word not in stop_words:

          filter_words.append(word)

    filtered_rev_words.append(filter_words)
```

```
print("----------------------------------------------------------------")
```

Exclude any reviews that contained only stopwords as these reviews will be blank or empty sublists []

```
print("filtered reviews:")

filtered_rev_words=[review for review in filtered_rev_words if len(review)>=1]

print(len(filtered_rev_words),filtered_rev_words[:100])

print("--------------------------------------------------------------------------")
```

Barplot to visualize the 100 most common words using FreqDist and barplots

```
list_of_words = [word for review in filtered_rev_words for word in review]

common_word_freq=FreqDist(list_of_words).most_common(100)

word_list = common_word_freq[::-1]


words,freq = [],[]

for word in word_list:

    words.append(word[0])

    freq.append(word[1])

x=np.array(words)

y=np.array(freq)
```

```
  plt.figure(figsize=(20,22))

  plt.barh(x,y,color="lightblue")

  plt.show()



  print(common_word_freq)


# Revising the stopwords based of above analysis

  stop_words_inclusions =
["...","..",'phone','good','bad','lenovo','k8','note','product',

                 'mobile','hai','please','pls','star','hi','ho','ok','superb','handset']

  stop_words = stop_words + stop_words_inclusions




#isalnum() to remove emoji an isnumeric() to remove only number tokens present
in the list

#len(word)!=1 will eliminate all one letter tokens such as 'u','i' etc.

  final_rev_words = []

  for review in filtered_rev_words:

    stopwords_removed_review=[]

    for word in review:

      if word not in stop_words and word.isalnum() and (not word.isnumeric())
and len(word)!=1:
```

```
        stopwords_removed_review.append(word)
    final_rev_words.append(stopwords_removed_review)
```

```
# Clearing any reviews which are now empty lists after removal of revised stop words
  final_rev_words=[review for review in final_rev_words if len(review)>=1]
  print(len(final_rev_words),final_rev_words[:50])
```

```
# Barplot to visualize the 100 most common words using FreqDist and barplots

  list_of_words = [word for review in final_rev_words for word in review]
  word_freq=FreqDist(list_of_words).most_common(100)
  word_list_2 = word_freq[::-1]

  words,freq = [],[]
  for word in word_list_2:
    words.append(word[0])
    freq.append(word[1])
  x=np.array(words)
  y=np.array(freq)
```

```
plt.figure(figsize=(20,22))

plt.barh(x,y,color="plum")

plt.show()
```

#Task 8: Create a topic model using LDA on the cleaned-up data with 12 topics.

#1. Print out the top terms for each topic.

#2. What is the coherence of the model with the c_v metric?

```
print("First creating the id2word Dictionary and corpus of words required for
the LDA topic model")

id2word = corpora.Dictionary(final_rev_words)

corpus =[]

for review in final_rev_words:

  new = id2word.doc2bow(review)

  corpus.append(new)

print(corpus[:20],"\n")

print("No of reviews:",len(corpus),"\n")
```

```
  print("No of unique words:",len(id2word),"\n")
  print("-----------------------------------------------------------")

  print("create LDA Model")

# Create a topic model using LDA on the cleaned-up data with 12 topics
  lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                    id2word=id2word,
                                    num_topics=12,
                                    random_state=47,
                                    update_every=1,
                                    chunksize=100,
                                    passes=10,
                                    alpha="auto")

  pprint(lda_model.print_topics())

#Print out the top terms for each topic.
  print("--------------------------------------------------------------")
  print("Top terms for each topic.")
```

```
  topics=[]

  topic_terms=[]

  for idx in range(12):

    topics.append("Topic "+ str(idx+1))

    terms=[]

    for term in lda_model.get_topic_terms(idx,topn=10):

      terms.append(id2word[term[0]])

    topic_terms.append(terms)


  for idx in range(12):

    print(idx,topic_terms[idx])


  df_topics = pd.DataFrame(topic_terms).transpose()

  df_topics.columns = topics

  print(df_topics)

  print("--------------------------------------------------------")

#What is the coherence of the model with the c_v metric?

# coherence of the model with the c_v metric?


  coherence_model_lda = CoherenceModel(model=lda_model,
texts=final_rev_words, dictionary=id2word, coherence='c_v')
```

coherence_lda = coherence_model_lda.get_coherence()

print('\nCoherence Score: ', coherence_lda)

print("---")

#Task 9 : Analyze the topics through the business lens.

#1. Determine which of the topics can be combined.

print("---")

vis=pyLDAvis.gensim_models.prepare(lda_model,corpus,id2word,mds='mmds',R=10)

pyLDAvis.show(vis)

print("--")

print("As per the LDA model with 12 topics many of these can be combined as per below . The ideal number of topics would be 4\n")

print("New Topics\t\t\t\t\tCurrent LDA model Topics\t\t\tKey Words for new topic classification\n")

print("Sale and Customer support\t\t\t\t3,8\t\t\tAmazon, service, support, replacement , refund, purchase, expectation, gorilla , glass, button, power, range, software, game\n")

print("Daily usage experience\t\t\t\t\t2,9\t\t\tCamera, quality, day, time, use, usage, time, network, call, signal, volta, music, speaker, processor, app, charging\n")

print("Phone features and performance\t\t\t\t\t4,7,11,12\t\t\tFeature, performance, speed, ram, price, sim, sound, experience, display, screen, video, stock, android, user, interface, apps, response, contact, gallery, photo, flash, mp, sensor, clarity\n")

print("Problems/issues and Pricing\t\t\t\t\t1,5,6,10\t\t\t\tIssue, problem, waste, update, bug, function, battery, backup, hour, hr, life, charger, charge, heat, heating, money, value, worth, cost, budget")

print("---")

#Task 10: Create topic model using LDA with what you think is the optimal number of topics

#1. What is the coherence of the model?

print("Creating LDA model with 4 topics")

lda_model_2 = gensim.models.ldamodel.LdaModel(corpus=corpus,

id2word=id2word,

num_topics=4,

random_state=47,

update_every=1,

chunksize=100,

passes=10,

alpha="auto")

#What is the coherence of the model?

```
# Coherence of the new model

  coherence_model_lda_2 = CoherenceModel(model=lda_model_2, texts=final_rev_words, dictionary=id2word, coherence='c_v')
  coherence_lda_2 = coherence_model_lda_2.get_coherence()
  print('\nCoherence Score: ', coherence_lda_2)
  print("--------------------------------------------------------------------------------------------")
  vis2 =pyLDAvis.gensim_models.prepare(lda_model_2,corpus,id2word,mds='mmds',R=25)
  pyLDAvis.show(vis2)
  print(lda_model_2.print_topics())
  print("--------------------------------------------------------------------------------------------")
  topics_model2=[]
  topic_terms_model2=[]
  for idx in range(4):
    topics_model2.append("Topic "+ str(idx+1))
    terms=[]
    for term in lda_model_2.get_topic_terms(idx,topn=10):
      terms.append(id2word[term[0]])
    topic_terms_model2.append(terms)
```

```
  for idx in range(4):
    print(idx,topic_terms_model2[idx])


  df_topics_model_2 = pd.DataFrame(topic_terms_model2).transpose()
  df_topics_model_2.columns=topics_model2
  print(df_topics_model_2)
  print("-----------------------------------------------------------------------------------------------
")
```

#Task 11: The business should be able to interpret the topics.

#1. Name each of the identified topics.

#2. Create a table with the topic name and the top 10 terms in each to present to the business.

```
  print("Create a table with the topic name and the top 10 terms in each to
present to the  business.")


  topics_model2 = ["Problems and Issues"," Key features for user", "Sales and
customer service", "Hardware specs and value features"]
  df_topics_model_2.columns=topics_model2
  print(df_topics_model_2) # Import required libraries, functions and classes
```

```
#Numpy and pandas for dataframes

import numpy as np

import pandas as pd


# nltk library for tokenization, lemmatizer, stopwords, pos tags and FreqDist

# import string for punctuation and str manipulations

import nltk

from nltk.tokenize import word_tokenize,TweetTokenizer

from nltk.tag import pos_tag

from nltk.stem import WordNetLemmatizer

from nltk import FreqDist

from nltk.corpus import stopwords

import string


#Gensim library for LDA model creation . Corpora in gensim to create the id2word
Dictionary and corpus of terms

import gensim

import gensim.corpora as corpora

from gensim.models import CoherenceModel


#visualization using matplotlib and pyLDAvis for the LDA model viz
```

```
import matplotlib.pyplot as plt

import pyLDAvis

import pyLDAvis.gensim_models

from pprint import pprint

#import warnings to ignore deprecation warning

import warnings

warnings.filterwarnings("ignore", category = DeprecationWarning)



if __name__=='__main__':

#Task 1 : Read the .csv file using Pandas. Take a look at the top few records.

  print("reading csv file:")

  file_path=input("enter path for the loan data file to load:")

  df_path=file_path.replace("\\",'/')



  reviews_df = pd.read_csv(df_path)



  print(reviews_df.head()) #look at the top few records

  print("-----------------------------------------------------------------------")

#Task 2: Normalize casings for the review text and extract the text into a list for easier manipulation
```

```
print("Normalize the text - reduce to lower case")

review_list = [review.lower() for review in reviews_df["review"]]

print(review_list[:5])

print("----------------------------------------------------------------------")
```

#Task 3:Tokenize the reviews using NLTKs word_tokenize function.

```
print("Tokenize the reviews:")



rev_words = [word_tokenize(review) for review in review_list]

print(rev_words[:5])

print("----------------------------------------------------------------------")
```

#Task 4: Perform parts-of-speech tagging on each sentence using the NLTK POS tagger.

```
print("POS tagging using NLTK pos tagger:")

pos_tagged_review = [pos_tag(review) for review in rev_words]

print(len(pos_tagged_review))

print(pos_tagged_review[:5])

print("----------------------------------------------------------------------")
```

#Task 5: For the topic model, we should want to include only nouns.

#1. Find out all the POS tags that correspond to nouns.

#2. Limit the data to only terms with these tags.

```
    print("Find out all the POS tags that correspond to nouns - Since pos_tag
function in NLTK library uses the Penn Treebank tagset.")


    pos_noun_reviews = []

    for review in pos_tagged_review:

      nouns=[]

      for word_tuple in review:

        if "NN" in word_tuple[1]:

          nouns.append(word_tuple)

      pos_noun_reviews.append(nouns)


    print(pos_noun_reviews[:50])


# Exclude any reviews that did not have any nouns as these reviews will be blank
or empty sublists []


    print("Limit the data to only terms with noun tags")

    pos_noun_reviews=[review for review in pos_noun_reviews if len(review)>=1]

    print(len(pos_noun_reviews), pos_noun_reviews[:50])

    print("------------------------------------------------------------------------")
```

#Task 6: Lemmatize.

#1. Different forms of the terms need to be treated as one.

#2. No need to provide POS tag to lemmatizer for now

```
  print("Lemmatize the different forms of the nouns")
# POS tags not passed to lemmatizer

  wnl = WordNetLemmatizer()
  lemmatized_words =[]
  for review in pos_noun_reviews:
    lemma_word=[]
    for word in review:
      lemma_word.append(wnl.lemmatize(word[0]))
    lemmatized_words.append(lemma_word)

  print(lemmatized_words[:50])
```

#Task 7: Remove stopwords and punctuation (if there are any).

```
  print("Remove stopwords and punctuation (if there are any)")
```

The o/p from lemmatizer still has many composite words that still contain emojis , special characters etc.

Using tweet tokenizer for isolating them better.

```
 tweet_tokenize = TweetTokenizer()
```

#Create list of stopwords with punctuations.Manually added token ['\s'] as this is usually seperated in tokenize

```
 stop_words = stopwords.words("english")

 stop_words = stop_words+list(string.punctuation)+["\'s"]


 filtered_rev_words=[]


 for review in lemmatized_words:
   filter_words=[]
   for words in review:
     rev_words = []
     rev_words = tweet_tokenize.tokenize(words)
     for word in rev_words:
       if word not in stop_words:
         filter_words.append(word)
   filtered_rev_words.append(filter_words)
```

```
  print("--------------------------------------------------------------------")

# Exclude any reviews that contained only stopwords as these reviews will be blank or empty sublists []

  print("filtered reviews:")

  filtered_rev_words=[review for review in filtered_rev_words if len(review)>=1]

  print(len(filtered_rev_words),filtered_rev_words[:100])

  print("-------------------------------------------------------------------------------")

# Barplot to visualize the 100 most common words using FreqDist and barplots


  list_of_words = [word for review in filtered_rev_words for word in review]

  common_word_freq=FreqDist(list_of_words).most_common(100)

  word_list = common_word_freq[::-1]


  words,freq = [],[]

  for word in word_list:

    words.append(word[0])

    freq.append(word[1])

  x=np.array(words)

  y=np.array(freq)


  plt.figure(figsize=(20,22))
```

```
  plt.barh(x,y,color="lightblue")

  plt.show()


  print(common_word_freq)


# Revising the stopwords based of above analysis

  stop_words_inclusions = 
["...","..",'phone','good','bad','lenovo','k8','note','product',

                  'mobile','hai','please','pls','star','hi','ho','ok','superb','handset']

  stop_words = stop_words + stop_words_inclusions




#isalnum() to remove emoji an isnumeric() to remove only number tokens present 
in the list

#len(word)!=1 will eliminate all one letter tokens such as 'u','i' etc.

  final_rev_words = []

  for review in filtered_rev_words:

    stopwords_removed_review=[]

    for word in review:

      if word not in stop_words and word.isalnum() and (not word.isnumeric()) 
and len(word)!=1:

        stopwords_removed_review.append(word)
```

```
    final_rev_words.append(stopwords_removed_review)


# Clearing any reviews which are now empty lists after removal of revised stop words

  final_rev_words=[review for review in final_rev_words if len(review)>=1]

  print(len(final_rev_words),final_rev_words[:50])


# Barplot to visualize the 100 most common words using FreqDist and barplots


  list_of_words = [word for review in final_rev_words for word in review]

  word_freq=FreqDist(list_of_words).most_common(100)

  word_list_2 = word_freq[::-1]


  words,freq = [],[]

  for word in word_list_2:

    words.append(word[0])

    freq.append(word[1])

  x=np.array(words)

  y=np.array(freq)


  plt.figure(figsize=(20,22))
```

```
  plt.barh(x,y,color="plum")

  plt.show()
```

#Task 8: Create a topic model using LDA on the cleaned-up data with 12 topics.

#1. Print out the top terms for each topic.

#2. What is the coherence of the model with the c_v metric?

```
  print("First creating the id2word Dictionary and corpus of words required for
the LDA topic model")

  id2word = corpora.Dictionary(final_rev_words)

  corpus =[]

  for review in final_rev_words:

    new = id2word.doc2bow(review)

    corpus.append(new)

  print(corpus[:20],"\n")

  print("No of reviews:",len(corpus),"\n")

  print("No of unique words:",len(id2word),"\n")
```

```
  print("----------------------------------------------------------")


  print("create LDA Model")


# Create a topic model using LDA on the cleaned-up data with 12 topics
  lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                           id2word=id2word,
                                           num_topics=12,
                                           random_state=47,
                                           update_every=1,
                                           chunksize=100,
                                           passes=10,
                                           alpha="auto")


  pprint(lda_model.print_topics())


#Print out the top terms for each topic.
  print("-------------------------------------------------------------")
  print("Top terms for each topic.")


  topics=[]
```

```
  topic_terms=[]

  for idx in range(12):

    topics.append("Topic "+ str(idx+1))

    terms=[]

    for term in lda_model.get_topic_terms(idx,topn=10):

      terms.append(id2word[term[0]])

    topic_terms.append(terms)


  for idx in range(12):

    print(idx,topic_terms[idx])


  df_topics = pd.DataFrame(topic_terms).transpose()

  df_topics.columns = topics

  print(df_topics)

  print("------------------------------------------------------------")
#What is the coherence of the model with the c_v metric?

# coherence of the model with the c_v metric?


  coherence_model_lda = CoherenceModel(model=lda_model, 
texts=final_rev_words, dictionary=id2word, coherence='c_v')

  coherence_lda = coherence_model_lda.get_coherence()
```

print('\nCoherence Score: ', coherence_lda)

print("--")

#Task 9 : Analyze the topics through the business lens.

#1. Determine which of the topics can be combined.

print("---")

vis=pyLDAvis.gensim_models.prepare(lda_model,corpus,id2word,mds='mmds',R=10)

pyLDAvis.show(vis)

print("--")

print("As per the LDA model with 12 topics many of these can be combined as per below . The ideal number of topics would be 4\n")

print("New Topics\t\t\t\t\tCurrent LDA model Topics\t\t\tKey Words for new topic classification\n")

print("Sale and Customer support\t\t\t\t3,8\t\t\tAmazon, service, support, replacement , refund, purchase, expectation, gorilla , glass, button, power, range, software, game\n")

print("Daily usage experience\t\t\t\t\t2,9\t\t\tCamera, quality, day, time, use, usage, time, network, call, signal, volta, music, speaker, processor, app, charging\n")

print("Phone features and performance\t\t\t\t\t4,7,11,12\t\t\tFeature, performance, speed, ram, price, sim, sound, experience, display, screen, video, stock, android, user, interface, apps, response, contact, gallery, photo, flash, mp, sensor, clarity\n") print("Problems/issues and Pricing\t\t\t\t\t1,5,6,10\t\t\t\tIssue, problem, waste, update, bug, function, battery, backup, hour, hr, life, charger, charge, heat, heating, money, value, worth, cost, budget")

print("---")

#Task 10: Create topic model using LDA with what you think is the optimal number of topics

#1. What is the coherence of the model?

print("Creating LDA model with 4 topics")

lda_model_2 = gensim.models.ldamodel.LdaModel(corpus=corpus,

id2word=id2word,

num_topics=4,

random_state=47,

update_every=1,

chunksize=100,

passes=10,

alpha="auto")

#What is the coherence of the model?

```
# Coherence of the new model

  coherence_model_lda_2 = CoherenceModel(model=lda_model_2, texts=final_rev_words, dictionary=id2word, coherence='c_v')

  coherence_lda_2 = coherence_model_lda_2.get_coherence()

  print('\nCoherence Score: ', coherence_lda_2)

  print("-----------------------------------------------------------------------------------------")

  vis2 =pyLDAvis.gensim_models.prepare(lda_model_2,corpus,id2word,mds='mmds',R=25)

  pyLDAvis.show(vis2)

  print(lda_model_2.print_topics())

  print("----------------------------------------------------------------------------------------")

  topics_model2=[]

  topic_terms_model2=[]

  for idx in range(4):

    topics_model2.append("Topic "+ str(idx+1))

    terms=[]

    for term in lda_model_2.get_topic_terms(idx,topn=10):

      terms.append(id2word[term[0]])

    topic_terms_model2.append(terms)
```

```
for idx in range(4):
    print(idx,topic_terms_model2[idx])


df_topics_model_2 = pd.DataFrame(topic_terms_model2).transpose()

df_topics_model_2.columns=topics_model2

print(df_topics_model_2)

print("---------------------------------------------------------------------------------------------
")
```

#Task 11: The business should be able to interpret the topics.

#1. Name each of the identified topics.

#2. Create a table with the topic name and the top 10 terms in each to present to the business.

```
print("Create a table with the topic name and the top 10 terms in each to
present to the  business.")


topics_model2 = ["Problems and Issues"," Key features for user", "Sales and
customer service", "Hardware specs and value features"]

df_topics_model_2.columns=topics_model2

print(df_topics_model_2)
```

Screenshot of the output:

Task 1 : Read the .csv file using Pandas. Take a look at the top few records.:

```
reading csv file:
enter path for the loan data file to load:C:\Users\sweth\Downloads\K8 Reviews v0.2.csv
   sentiment                                             review
0          1              Good but need updates and improvements
1          0  Worst mobile i have bought ever, Battery is dr...
2          1  when I will get my 10% cash back.... its alrea...
3          1                                               Good
4          0  The worst phone everThey have changed the last...
----------------------------------------------------------------------
```

Task 2: Normalize casings for the review text and extract the text into a list for easier manipulation:

```
Normalize the text - reduce to lower case
['good but need updates and improvements', "worst mobile i have bought ever, battery is draining like hell, backup is only 6 to 7 hours with internet uses, even if i put mobile idle its getting discharged.this i
s biggest lie from amazon & lenove which is not at all expected, they are making full by saying that battery is 4000mah & booster charger is fake, it takes at least 4 to 5 hours to be fully charged.don't know ho
w lenovo will survive by making full of us.please don;t go for this else you will regret like me.", 'when i will get my 10% cash back.... its already 15 january..', 'good', 'the worst phone everthey have changed
 the last phone but the problem is still same and the amazon is not returning the phone .highly disappointing of amazon']
----------------------------------------------------------------------
```

Task 3:Tokenize the reviews using NLTKs word_tokenize function.:

```
Tokenize the reviews:
[['good', 'but', 'need', 'updates', 'and', 'improvements'], ['worst', 'mobile', 'i', 'have', 'bought', 'ever', ',', 'battery', 'is', 'draining', 'like', 'hell', ',', 'backup', 'is', 'only', '6', 'to', '7', 'hour
s', 'with', 'internet', 'uses', ',', 'even', 'if', 'i', 'put', 'mobile', 'idle', 'its', 'getting', 'discharged.this', 'is', 'biggest', 'lie', 'from', 'amazon', '&', 'lenove', 'which', 'is', 'not', 'at', 'all', '
expected', ',', 'they', 'are', 'making', 'full', 'by', 'saying', 'that', 'battery', 'is', '4000mah', '&', 'booster', 'charger', 'is', 'fake', ',', 'it', 'takes', 'at', 'least', '4', 'to', '5', 'hours', 'to', 'be
', 'fully', 'charged.do', "n't", 'know', 'how', 'lenovo', 'will', 'survive', 'by', 'making', 'full', 'of', 'us.please', 'don', ';', 't', 'go', 'for', 'this', 'else', 'you', 'will', 'regret', 'like', 'me', '.'],
['when', 'i', 'will', 'get', 'my', '10', '%', 'cash', 'back', '....', 'its', 'already', '15', 'january', '..'], ['good'], ['the', 'worst', 'phone', 'everthey', 'have', 'changed', 'the', 'last', 'phone', 'but', '
the', 'problem', 'is', 'still', 'same', 'and', 'the', 'amazon', 'is', 'not', 'returning', 'the', 'phone', '.highly', 'disappointing', 'of', 'amazon']]
----------------------------------------------------------------------
```

Task 4: Perform parts-of-speech tagging on each sentence using the NLTK POS tagger.:

```
POS tagging using NLTK pos tagger:
14675
[[('good', 'JJ'), ('but', 'CC'), ('need', 'VBP'), ('updates', 'NNS'), ('and', 'CC'), ('improvements', 'NNS')], [('worst', 'JJS'), ('mobile', 'NN'), ('i', 'NN'), ('have', 'VBP'), ('bought', 'VBN'), ('ever', 'RB')
, (',', ','), ('battery', 'NN'), ('is', 'VBZ'), ('draining', 'VBG'), ('like', 'IN'), ('hell', 'NN'), (',', ','), ('backup', 'NN'), ('is', 'VBZ'), ('only', 'RB'), ('6', 'CD'), ('to', 'TO'), ('7', 'CD'), ('hours',
 'NNS'), ('with', 'IN'), ('internet', 'JJ'), ('uses', 'NNS'), (',', ','), ('even', 'RB'), ('if', 'IN'), ('i', 'JJ'), ('put', 'VBP'), ('mobile', 'JJ'), ('idle', 'NN'), ('its', 'PRP$'), ('getting', 'VBG'), ('disch
arged.this', 'NN'), ('is', 'VBZ'), ('biggest', 'JJS'), ('lie', 'NN'), ('from', 'IN'), ('amazon', 'NN'), ('&', 'CC'), ('lenove', 'NN'), ('which', 'WDT'), ('is', 'VBZ'), ('not', 'RB'), ('at', 'IN'), ('all', 'DT'),
 ('expected', 'VBN'), (',', ','), ('they', 'PRP'), ('are', 'VBP'), ('making', 'VBG'), ('full', 'JJ'), ('by', 'IN'), ('saying', 'VBG'), ('that', 'DT'), ('battery', 'NN'), ('is', 'VBZ'), ('4000mah', 'CD'), ('&', '
CC'), ('booster', 'JJR'), ('charger', 'NN'), ('is', 'VBZ'), ('fake', 'JJ'), (',', ','), ('it', 'PRP'), ('takes', 'VBZ'), ('at', 'IN'), ('least', 'JJS'), ('4', 'CD'), ('to', 'TO'), ('5', 'CD'), ('hours', 'NNS'),
('to', 'TO'), ('be', 'VB'), ('fully', 'RB'), ('charged.do', 'VBP'), ("n't", 'RB'), ('know', 'VB'), ('how', 'WRB'), ('lenovo', 'JJ'), ('will', 'MD'), ('survive', 'VB'), ('by', 'IN'), ('making', 'VBG'), ('full', '
JJ'), ('of', 'IN'), ('us.please', 'JJ'), ('don', 'NN'), (';', ':'), ('t', 'CC'), ('go', 'VB'), ('for', 'IN'), ('this', 'DT'), ('else', 'JJ'), ('you', 'PRP'), ('will', 'MD'), ('regret', 'VB'), ('like', 'IN'), ('m
e', 'PRP'), ('.', '.')], [('when', 'WRB'), ('i', 'NN'), ('will', 'MD'), ('get', 'VB'), ('my', 'PRP$'), ('10', 'CD'), ('%', 'NN'), ('cash', 'NN'), ('back', 'RB'), ('....', 'VBZ'), ('its', 'PRP$'), ('already', 'RB
'), ('15', 'CD'), ('january', 'JJ'), ('..', 'NN')], [('good', 'JJ')], [('the', 'DT'), ('worst', 'JJS'), ('phone', 'NN'), ('everthey', 'NN'), ('have', 'VBP'), ('changed', 'VBN'), ('the', 'DT'), ('last', 'JJ'), ('
phone', 'NN'), ('but', 'CC'), ('the', 'DT'), ('problem', 'NN'), ('is', 'VBZ'), ('still', 'RB'), ('same', 'JJ'), ('and', 'CC'), ('the', 'DT'), ('amazon', 'NN'), ('is', 'VBZ'), ('not', 'RB'), ('returning', 'VBG'),
 ('the', 'DT'), ('phone', 'NN'), ('.highly', 'RB'), ('disappointing', 'JJ'), ('of', 'IN'), ('amazon', 'NN')]]
```

Task 5: For the topic model, we should want to include only nouns.:


```
----------------------------------------------------------------------
Find out all the POS tags that correspond to nouns - Since pos_tag function in NLTK library uses the Penn Treebank tagset.
[[('updates', 'NNS'), ('improvements', 'NNS')], [('mobile', 'NN'), ('i', 'NN'), ('battery', 'NN'), ('hell', 'NN'), ('backup', 'NN'), ('hours', 'NNS'), ('uses', 'NNS'), ('idle', 'NN'), ('discharged.this', 'NN'),
('lie', 'NN'), ('amazon', 'NN'), ('lenove', 'NN'), ('battery', 'NN'), ('charger', 'NN'), ('hours', 'NNS'), ('don', 'NN')], [('i', 'NN'), ('%', 'NN'), ('cash', 'NN'), ('..', 'NN')], [], [('phone', 'NN'), ('everth
ey', 'NN'), ('phone', 'NN'), ('problem', 'NN'), ('amazon', 'NN'), ('phone', 'NN'), ('amazon', 'NN')], [('camerawaste', 'NN'), ('money', 'NN')], [('phone', 'NN'), ('allot', 'NN'), ('..', 'NNP'), ('reason', 'NN'),
 ('k8', 'NNS')], [('battery', 'NN'), ('level', 'NN')], [('problems', 'NNS'), ('phone', 'NN'), ('hanging', 'NN'), ('problems', 'NNS'), ('note', 'NN'), ('station', 'NN'), ('ahmedabad', 'NN'), ('years', 'NNS'), ('p
hone', 'NN'), ('lenovo', 'NN')], [('lot', 'NN'), ('glitches', 'NNS'), ('thing', 'NN'), ('options', 'NNS')], [('wrost', 'NN')], [('phone', 'NN'), ('charger', 'NN'), ('damage', 'NN'), ('months', 'NNS')], [('item',
 'NN'), ('battery', 'NN'), ('life', 'NN')], [('i', 'NNS'), ('battery', 'NN'), ('problem', 'NN'), ('motherboard', 'NN'), ('problem', 'NN'), ('months', 'NNS'), ('mobile', 'NN'), ('life', 'NN')], [('phone', 'NN'),
('slim', 'NN'), ('battry', 'NN'), ('backup', 'NN'), ('screen', 'NN')], [('headset', 'NN')], [('time', 'NN'), ('i', 'NN')], [('product', 'NN'), ('prize', 'NN'), ('range', 'NN'), ('specification', 'NN'), ('compari
son', 'NN'), ('mobile', 'NN'), ('range', 'NN'), ('i', 'NN'), ('phone', 'NN'), ('seal', 'NN'), ('i', 'NNS'), ('credit', 'NN'), ('card', 'NN'), ('i', 'NN'), ('..', 'NNP'), ('..', 'NNP'), ('deal', 'NN'), ('amazon',
 'NN'), ('..', 'NN')], [('battery', 'NN'), ('..', 'NNP'), ('solutions', 'NNS'), ('battery', 'NN'), ('life', 'NN')], [('smartphone', 'NN')], [], [('galery', 'NN'), ('problem', 'NN'), ('speaker', 'NN'), ('phone',
'NN')], [('camera', 'NN'), ('speed.excellent', 'NN'), ('features.excelent', 'NN'), ('battery', 'NN')], [('product', 'NN')], [('product', 'NN'), ('camera', 'NN'), ('os', 'NN'), ('battery', 'NN'), ('phone', 'NN'),
 ('product', 'NN'), ('..', 'NN')], [('options', 'NNS'), ('cast', 'NN'), ('screen', 'NN'), ('wifi', 'NN'), ('call', 'NN'), ('option', 'NN'), ('mobile', 'NN'), ('hotspot', 'NN')], [('phone', 'NN'), ('usb', 'NN'),
('cable', 'NN')], [('phone', 'NN'), ('price', 'NN'), ('mobile', 'NN'), ('lenovo', 'NN'), ('display', 'NN')], [('specifications', 'NNS'), ('functions', 'NNS'), ('phone', 'NN'), ('any1', 'NN'), ('..', 'NNS')], [('
i', 'NN'), ('fon', 'NN'), ('fon', 'NN'), ('speekars', 'NNS'), ('i', 'NN')], [('phone', 'NN'), ('issue', 'NN'), ('colors', 'NNS'), ('screen', 'NN')], [('update', 'NN'), ('oreo', 'NN'), ('battery', 'NN'), ('back-u
p', 'NN'), ('heating', 'NN'), ('problem', 'NN'), ('update', 'NN'), ('phone', 'NN'), ('battery', 'NN'), ('update', 'NN'), ('oreo', 'NN'), ('□ □ □□', 'NN')], [('one', 'NN'), ('sim', 'NN'), ('customer', 'NN'), ('se
rvice', 'NN')], [('performance', 'NN'), ('battery', 'NN')], [('camera', 'NN'), ('backup', 'NN'), ('pricefull', 'NN'), ('passa', 'NN'), ('wasole', 'NN'), ('phone', 'NN')], [('phone', 'NN')], [('performance', 'NN'
), ('..', 'NNP'), ('signal', 'NN'), ('restarts', 'NNS'), ('phone', 'NN'), ('bcoms', 'NNS'), ('..', 'NNP'), ('plzz', 'NN'), ('dont', 'NN')], [('round', 'NN'), ('performance', 'NN')], [('rs.3000', 'NN'), ('span',
'NN'), ('days', 'NNS'), ('trust', 'NN'), ('deals', 'NNS'), ('amazon', 'NN')], [('disappointment', 'NN'), ('signal', 'NN'), ('problems', 'NNS'), ('headache', 'NN'), ('problems', 'NNS'), ('calls', 'NNS'), ('range'
, 'NN')], [('phone', 'NN'), ('rate', 'NN'), ('camera', 'NN'), ('quality', 'NN')], [], [('product', 'NN'), ('mobile', 'NN'), ('price', 'NN'), ('features', 'NNS')], [('mobile', 'NN'), ('price', 'NN'), ('quality',
'NN'), ('camera', 'NN')], [('mins', 'NNS'), ('quality', 'NN')], [('phone', 'NN'), ('i', 'NN'), ('issue', 'NN')], [], [('i', 'NN'), ('k8', 'NN'), ('sell', 'NN'), ('phone', 'NN'), ('take', 'NN'), ('hr', 'NN'), ('c
harge', 'NN'), ('.....', 'NN')], [('ekdam', 'NN'), ('network', 'NN'), ('problem', 'NN'), ('k8', 'NN'), ('camera', 'NN'), ('quality', 'NN'), ('performance', 'NN')], [('phone', 'NN')]]
Limit the data to only terms with noun tags
13487 [[('updates', 'NNS'), ('improvements', 'NNS')], [('mobile', 'NN'), ('i', 'NN'), ('battery', 'NN'), ('hell', 'NN'), ('backup', 'NN'), ('hours', 'NNS'), ('uses', 'NNS'), ('idle', 'NN'), ('discharged.this', '
NN'), ('lie', 'NN'), ('amazon', 'NN'), ('lenove', 'NN'), ('battery', 'NN'), ('charger', 'NN'), ('hours', 'NNS'), ('don', 'NN')], [('i', 'NN'), ('%', 'NN'), ('cash', 'NN'), ('..', 'NN')], [('phone', 'NN'), ('ever
they', 'NN'), ('phone', 'NN'), ('problem', 'NN'), ('amazon', 'NN'), ('phone', 'NN'), ('amazon', 'NN')], [('camerawaste', 'NN'), ('money', 'NN')], [('phone', 'NN'), ('allot', 'NN'), ('..', 'NNP'), ('reason', 'NN'
), ('k8', 'NNS')], [('battery', 'NN'), ('level', 'NN')], [('problems', 'NNS'), ('phone', 'NN'), ('hanging', 'NN'), ('problems', 'NNS'), ('note', 'NN'), ('station', 'NN'), ('ahmedabad', 'NN'), ('years', 'NNS'), (
'phone', 'NN'), ('lenovo', 'NN')], [('lot', 'NN'), ('glitches', 'NNS'), ('thing', 'NN'), ('options', 'NNS')], [('wrost', 'NN')], [('phone', 'NN'), ('charger', 'NN'), ('damage', 'NN'), ('months', 'NNS')], [('item
', 'NN'), ('battery', 'NN'), ('life', 'NN')], [('i', 'NNS'), ('battery', 'NN'), ('problem', 'NN'), ('motherboard', 'NN'), ('problem', 'NN'), ('months', 'NNS'), ('mobile', 'NN'), ('life', 'NN')], [('phone', 'NN')
, ('slim', 'NN'), ('battry', 'NN'), ('backup', 'NN'), ('screen', 'NN')], [('headset', 'NN')], [('time', 'NN'), ('i', 'NN')], [('product', 'NN'), ('prize', 'NN'), ('range', 'NN'), ('specification', 'NN'), ('compa
rison', 'NN'), ('mobile', 'NN'), ('range', 'NN'), ('i', 'NN'), ('phone', 'NN'), ('seal', 'NN'), ('i', 'NNS'), ('credit', 'NN'), ('card', 'NN'), ('i', 'NN'), ('..', 'NNP'), ('..', 'NNP'), ('deal', 'NN'), ('amazon
', 'NN'), ('..', 'NN')], [('battery', 'NN'), ('..', 'NNP'), ('solutions', 'NNS'), ('battery', 'NN'), ('life', 'NN')], [('smartphone', 'NN')], [('galery', 'NN'), ('problem', 'NN'), ('speaker', 'NN'), ('phone', 'N
N')], [('camera', 'NN'), ('speed.excellent', 'NN'), ('features.excelent', 'NN'), ('battery', 'NN')], [('product', 'NN')], [('product', 'NN'), ('camera', 'NN'), ('os', 'NN'), ('battery', 'NN'), ('phone', 'NN'), (
'product', 'NN'), ('..', 'NN')], [('options', 'NNS'), ('cast', 'NN'), ('screen', 'NN'), ('wifi', 'NN'), ('call', 'NN'), ('option', 'NN'), ('mobile', 'NN'), ('hotspot', 'NN')], [('phone', 'NN'), ('usb', 'NN'), ('
cable', 'NN')], [('phone', 'NN'), ('price', 'NN'), ('mobile', 'NN'), ('lenovo', 'NN'), ('display', 'NN')], [('specifications', 'NNS'), ('functions', 'NNS'), ('phone', 'NN'), ('any1', 'NN'), ('..', 'NNS')], [('i'
, 'NN'), ('fon', 'NN'), ('fon', 'NN'), ('speekars', 'NNS'), ('i', 'NN')], [('phone', 'NN'), ('issue', 'NN'), ('colors', 'NNS'), ('screen', 'NN')], [('update', 'NN'), ('oreo', 'NN'), ('battery', 'NN'), ('back-up'
, 'NN'), ('heating', 'NN'), ('problem', 'NN'), ('update', 'NN'), ('phone', 'NN'), ('battery', 'NN'), ('update', 'NN'), ('oreo', 'NN'), ('□ □ □□', 'NN')], [('one', 'NN'), ('sim', 'NN'), ('customer', 'NN'), ('serv
ice', 'NN')], [('performance', 'NN'), ('battery', 'NN')], [('camera', 'NN'), ('backup', 'NN'), ('pricefull', 'NN'), ('passa', 'NN'), ('wasole', 'NN'), ('phone', 'NN')], [('phone', 'NN')], [('performance', 'NN'),
 ('..', 'NNP'), ('signal', 'NN'), ('restarts', 'NNS'), ('phone', 'NN'), ('bcoms', 'NNS'), ('..', 'NNP'), ('plzz', 'NN'), ('dont', 'NN')], [('round', 'NN'), ('performance', 'NN')], [('rs.3000', 'NN'), ('span', 'N
N'), ('days', 'NNS'), ('trust', 'NN'), ('deals', 'NNS'), ('amazon', 'NN')], [('disappointment', 'NN'), ('signal', 'NN'), ('problems', 'NNS'), ('headache', 'NN'), ('problems', 'NNS'), ('calls', 'NNS'), ('range',
'NN')], [('phone', 'NN'), ('rate', 'NN'), ('camera', 'NN'), ('quality', 'NN')], [('product', 'NN'), ('mobile', 'NN'), ('price', 'NN'), ('features', 'NNS')], [('mobile', 'NN'), ('price', 'NN'), ('quality', 'NN'),
 ('camera', 'NN')], [('mins', 'NNS'), ('quality', 'NN')], [('phone', 'NN'), ('i', 'NN'), ('issue', 'NN')], [('i', 'NN'), ('k8', 'NN'), ('sell', 'NN'), ('phone', 'NN'), ('take', 'NN'), ('hr', 'NN'), ('charge', 'N
N'), ('.....', 'NN')], [('ekdam', 'NN'), ('network', 'NN'), ('problem', 'NN'), ('k8', 'NN'), ('camera', 'NN'), ('quality', 'NN'), ('performance', 'NN')], [('phone', 'NN')], [('performance', 'NN'), ('rm', 'NNS'),
 ('memory', 'NN')], [('superb', 'NN'), ('featurs', 'NNS'), ('battery', 'NN'), ('phone□ □ ', 'NN')], [('value', 'NN'), ('money', 'NN')], [('lenovo', 'NN'), ('k8', 'NN')]]
```

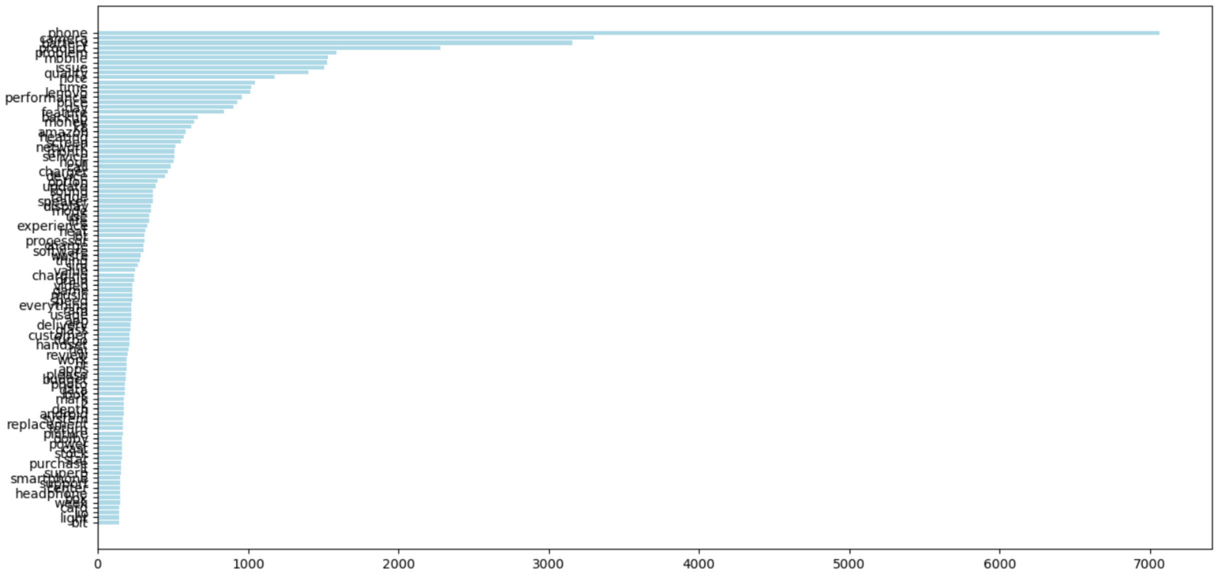

Task 6: Lemmatize:


```
Lemmatize the different forms of the nouns
[['update', 'improvement'], ['mobile', 'i', 'battery', 'hell', 'backup', 'hour', 'us', 'idle', 'discharged.this', 'lie', 'amazon', 'lenove', 'battery', 'charger', 'hour', 'don'], ['i', '%', 'cash', '..'], ['phon
e', 'everthey', 'phone', 'problem', 'amazon', 'phone', 'amazon'], ['camerawaste', 'money'], ['phone', 'allot', '..', 'reason', 'k8'], ['battery', 'level'], ['problem', 'phone', 'hanging', 'problem', 'note', 'sta
tion', 'ahmedabad', 'year', 'phone', 'lenovo'], ['lot', 'glitch', 'thing', 'option'], ['wrost'], ['phone', 'charger', 'damage', 'month'], ['item', 'battery', 'life'], ['i', 'battery', 'problem', 'motherboard', '
problem', 'month', 'mobile', 'life'], ['phone', 'slim', 'battry', 'backup', 'screen'], ['headset'], ['time', 'i'], ['product', 'prize', 'range', 'specification', 'comparison', 'mobile', 'range', 'i', 'phone', 's
eal', 'i', 'credit', 'card', 'i', '..', '..', 'deal', 'amazon', '..'], ['battery', '..', 'solution', 'battery', 'life'], ['smartphone'], ['galery', 'problem', 'speaker', 'phone'], ['camera', 'speed.excellent', '
features.excelent', 'battery'], ['product'], ['product', 'camera', 'o', 'battery', 'phone', 'product', '..'], ['option', 'cast', 'screen', 'wifi', 'call', 'option', 'mobile', 'hotspot'], ['phone', 'usb', 'cable'
], ['phone', 'price', 'mobile', 'lenovo', 'display'], ['specification', 'function', 'phone', 'any1', '..'], ['i', 'fon', 'fon', 'speekars', 'i'], ['phone', 'issue', 'color', 'screen'], ['update', 'oreo', 'batter
y', 'back-up', 'heating', 'problem', 'update', 'phone', 'battery', 'update', 'oreo', '□ □ □□'], ['one', 'sim', 'customer', 'service'], ['performance', 'battery'], ['camera', 'backup', 'pricefull', 'passa', 'waso
le', 'phone'], ['phone'], ['performance', '..', 'signal', 'restarts', 'phone', 'bcoms', '..', 'plzz', 'dont'], ['round', 'performance'], ['rs.3000', 'span', 'day', 'trust', 'deal', 'amazon'], ['disappointment',
'signal', 'problem', 'headache', 'problem', 'call', 'range'], ['phone', 'rate', 'camera', 'quality'], ['product', 'mobile', 'price', 'feature'], ['mobile', 'price', 'quality', 'camera'], ['min', 'quality'], ['ph
one', 'i', 'issue'], ['i', 'k8', 'sell', 'phone', 'take', 'hr', 'charge', '.....'], ['ekdam', 'network', 'problem', 'k8', 'camera', 'quality', 'performance'], ['phone'], ['performance', 'rm', 'memory'], ['superb
', 'featurs', 'battery', 'phone□ □ '], ['value', 'money'], ['lenovo', 'k8']]
```


Task 7: Remove stopwords and punctuation (if there are any).:


```
filtered reviews:
13453 [['update', 'improvement'], ['mobile', 'battery', 'hell', 'backup', 'hour', 'us', 'idle', 'discharged.this', 'lie', 'amazon', 'lenove', 'battery', 'charger', 'hour'], ['cash', '..'], ['phone', 'everthey', '
phone', 'problem', 'amazon', 'phone', 'amazon'], ['camerawaste', 'money'], ['phone', 'allot', '..', 'reason', 'k8'], ['battery', 'level'], ['problem', 'phone', 'hanging', 'problem', 'note', 'station', 'ahmedaba
d', 'year', 'phone', 'lenovo'], ['lot', 'glitch', 'thing', 'option'], ['wrost'], ['phone', 'charger', 'damage', 'month'], ['item', 'battery', 'life'], ['battery', 'problem', 'motherboard', 'problem', 'month', 'm
obile', 'life'], ['phone', 'slim', 'battry', 'backup', 'screen'], ['headset'], ['time'], ['product', 'prize', 'range', 'specification', 'comparison', 'mobile', 'range', 'phone', 'seal', 'credit', 'card', '..', '
..', 'deal', 'amazon', '..'], ['battery', '..', 'solution', 'battery', 'life'], ['smartphone'], ['galery', 'problem', 'speaker', 'phone'], ['camera', 'speed.excellent', 'features.excelent', 'battery'], ['product
'], ['product', 'camera', 'battery', 'phone', 'product', '..'], ['option', 'cast', 'screen', 'wifi', 'call', 'option', 'mobile', 'hotspot'], ['phone', 'usb', 'cable'], ['phone', 'price', 'mobile', 'lenovo', 'dis
play'], ['specification', 'function', 'phone', '1', '..'], ['fon', 'fon', 'speekars'], ['phone', 'issue', 'color', 'screen'], ['update', 'oreo', 'battery', 'back-up', 'heating', 'problem', 'update', 'phone', 'ba
ttery', 'update', 'oreo', '□', '□', '□', '□'], ['one', 'sim', 'customer', 'service'], ['performance', 'battery'], ['camera', 'backup', 'pricefull', 'passa', 'wasole', 'phone'], ['phone'], ['performance', '..',
'signal', 'restarts', 'phone', 'bcoms', '..', 'plzz', 'dont'], ['round', 'performance'], ['rs', '3000', 'span', 'day', 'trust', 'deal', 'amazon'], ['disappointment', 'signal', 'problem', 'headache', 'problem',
'call', 'range'], ['phone', 'rate', 'camera', 'quality'], ['product', 'mobile', 'price', 'feature'], ['mobile', 'price', 'quality', 'camera'], ['min', 'quality'], ['phone', 'issue'], ['k8', 'sell', 'phone', 'tak
e', 'hr', 'charge', '...'], ['ekdam', 'network', 'problem', 'k8', 'camera', 'quality', 'performance'], ['phone'], ['performance', 'rm', 'memory'], ['superb', 'featurs', 'battery', 'phone', '□ ', '□ '], ['value',
 'money'], ['lenovo', 'k8'], ['problem', 'speaker', 'breakup', 'problem', 'hour'], ['phone', 'heating', 'problem', 'choice'], ['battery', 'standby', 'problem'], ['battery', 'camera', 'jet', 'speed', 'apps', '□ '
, '□ '], ['product'], ['ok'], ['screen', 'cast', 'tv', 'miracast', 'charging', 'hour', 'contact', 'file', 'google', 'card', 'total'], ['device', 'usage', 'battery', 'day', 'con', 'phone', 'battery', 'mah', 'batt
ery', 'camera', 'quality', 'depth', 'mode', 'focus', 'thing', 'device', 'display'], ['india', 'ka', 'battery', 'offer', 'hour', 'network', 'signal', 'call', 'coverage', 'area.lenovo', 'people'], ['mobile'], ['mo
bile', 'range', 'battery', 'backup', 'mark', 'weight', '...'], ['price', 'point'], ['camera', 'mo', 'issue', 'system', 'system', 'update'], ['superb'], ['mobile', 'phone'], ['product'], ['delivery', 'rs', '10000
', 'gesture', 'feature', 'issue', 'brand'], ['mobile.great', 'specification', 'camera', 'focus', 'thanks', 'amazon', 'lenovo', 'buy', 'guy'], ['verigood', 'verigood.best'], ['product', 'time', 'naugat', 'product
', 'cost', 'phone', 'feel', 'nerve', 'radiation', 'need', 'fix', 'matter', 'safety', 'security', 'customer.fifthly', 'issue', 'hardware', 'issue', 'service', 'center', 'location', 'availability', 'time', 'custom
er', 'product', 'feature', 'issue', 'recording', 'option', 'product', 'expectations.camera', 'quality', 'snap'], ['product'], ['device'], ['product', 'amazon', 'phone'], ['hello', '..', 'performance', 'device',
'picture', 'use', 'hour', 'data', 'battery', 'mode', 'performanc', 'turbo', 'charger', 'waste', 'charger', 'return', 'request.pros', 'rest', 'good', 'cameradesignperformance', 'etc'], ['phone', 'problem'], ['pro
duct', '0', '□', '□ '], ['product', '..', 'lot', 'malfunction', 'donot', 'product'], ['look', 'sleekness', 'phone', 'feature', 'phone', 'phone', 'feature', 'sound', 'controler', 'option', 'shortcut', 'shutter',
'voice', 'call', 'recorder', 'camera', 'quality', 'performance'], ['class', 'phone', 'time', 'lenovo', 'note', 'note', 'kind', 'thinking', 'load', 'battery', 'class', 'defuces', 'time', 'charge', 'time', 'charge
r.gallery', 'othet', 'apps', 'stock', 'android', 'thing', 'process.android.com', '..'], ['ok', 'product', 'problem'], ['lenovo', 'k8', 'note', 'handset', 'time', 'charger', 'service', 'centre', 'month', 'fix', '
u'], ['feature'], ['gift', 'raping', 'poorwords', 'money', 'work'], ['product', 'anyone', 'tv', 'cast', 'screen', 'option', 'option', 'device', 'chromecast', 'casting', 'model', 'phone'], ['range', 'phone', 'u',
 'feature', 'problem', 'arises', 'money'], ['sim', 'slot', 'delivery', 'star', 'bcz', 'lenovo', 'fan', 'mobile', 'year', 'time', 'guess', 'phone', 'lot', 'feature', 'look'], ['gaming', 'music', '..', 'deca', 'co
re', 'processor', 'quad', 'core', 'phone', 'speaker', 'speaker', 'hole', 'buyer', 'camera', 'display', 'battery', 'keep', 'phone', 'use'], ['battery', 'backup', 'problem'], ['phone', '...'], ['product', 'cash',
'way', 'fool', 'cash', 'back.lol'], ['mobile'], ['mobile', 'heating', 'problem', 'thing', 'popularity'], ['segment', 'productcamera', 'greatsound', 'superb'], ['mobile', 'process', 'drawback', 'turbo', 'charger'
, 'mobile'], ['experience', 'lenovo', 'note', 'problem', 'battery', 'problem', 'camera', 'problem', 'plz', 'buy', 'phone'], ['phone'], ['..'], ['mo'], ['expectation', '□ ', 'camera', 'mark', '...', 'issue', 'nee
d', '...'], ['lenovo', 'sale', 'dolby', 'feature', 'lenovo', 'switch', 'k6', 'sofwares', 'phone', 'please']]
```

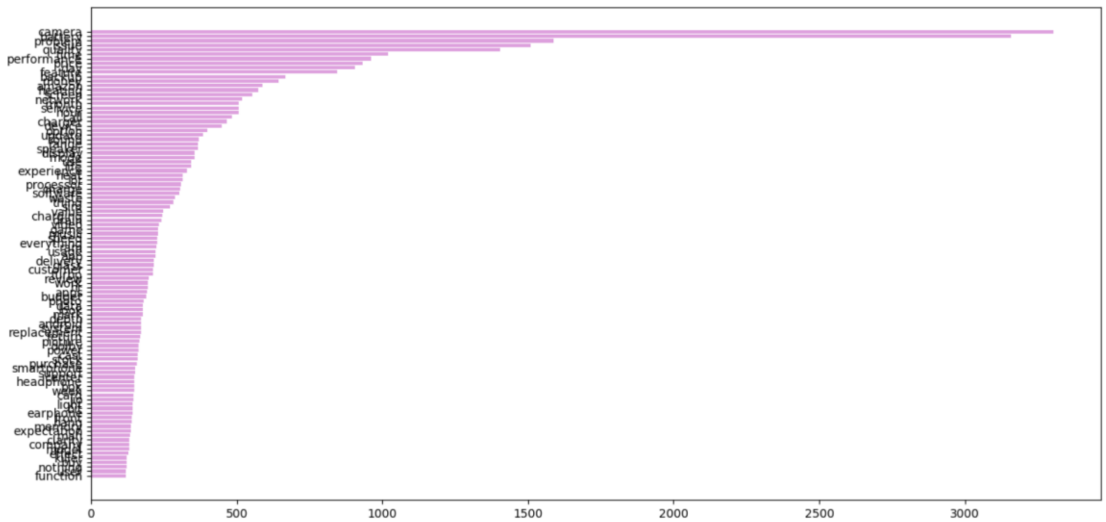
Barplot to visualize the 100 most common words using FreqDist and barplots:



Clearing any reviews which are now empty lists after removal of revised stop words

```
[('phone', 7062), ('camera', 3303), ('battery', 3157), ('product', 2279), ('problem', 1589), ('mobile', 1530), ('..', 1527), ('issue', 1509), ('quality', 1404), ('note', 1177), ('...', 1047), ('time', 1019), ('
enovo', 1013), ('performance', 961), ('price', 931), ('day', 905), ('feature', 844), ('backup', 667), ('money', 644), ('k8', 626), ('amazon', 588), ('heating', 575), ('screen', 555), ('network', 519), ('month',
508), ('service', 508), ('hour', 506), ('call', 483), ('charger', 467), ('device', 449), ('option', 399), ('update', 385), ('sound', 369), ('range', 368), ('speaker', 366), ('display', 356), ('mode', 354), ('us
', 344), ('life', 343), ('experience', 328), ('heat', 316), ('lot', 314), ('processor', 309), ('charge', 307), ('software', 303), ('waste', 289), ('thing', 282), ('sim', 270), ('value', 247), ('charging', 246),
('drain', 241), ('video', 233), ('game', 231), ('music', 230), ('speed', 228), ('everything', 226), ('ram', 224), ('usage', 222), ('app', 222), ('delivery', 217), ('glass', 217), ('customer', 214), ('turbo', 2
), ('handset', 212), ('hai', 204), ('review', 198), ('work', 196), ('hr', 194), ('apps', 193), ('please', 188), ('budget', 188), ('photo', 181), ('data', 179), ('look', 178), ('mark', 177), ('2', 177), ('depth'
 173), ('android', 172), ('system', 171), ('replacement', 171), ('return', 169), ('picture', 167), ('dolby', 163), ('power', 162), ('cast', 159), ('stock', 159), ('star', 159), ('purchase', 158), ('1', 155), ('
uperb', 153), ('smartphone', 152), ('support', 152), ('center', 149), ('headphone', 148), ('box', 148), ('week', 148), ('card', 146), ('jio', 146), ('light', 144), ('bit', 143)]
11858 [['update', 'improvement'], ['battery', 'hell', 'backup', 'hour', 'us', 'idle', 'lie', 'amazon', 'lenove', 'battery', 'charger', 'hour'], ['cash'], ['everthey', 'problem', 'amazon', 'amazon'], ['camerawas
e', 'money'], ['allot', 'reason'], ['battery', 'level'], ['problem', 'hanging', 'problem', 'station', 'ahmedabad', 'year'], ['lot', 'glitch', 'thing', 'option'], ['wrost'], ['charger', 'damage', 'month'], ['ite
', 'battery', 'life'], ['battery', 'problem', 'motherboard', 'problem', 'month', 'life'], ['slim', 'battry', 'backup', 'screen'], ['headset'], ['time'], ['prize', 'range', 'specification', 'comparison', 'range'
 'seal', 'credit', 'card', 'deal', 'amazon'], ['battery', 'solution', 'battery', 'life'], ['smartphone'], ['galery', 'problem', 'speaker'], ['camera', 'battery'], ['camera', 'battery'], ['option', 'cast', 'scre
n', 'wifi', 'call', 'option', 'hotspot'], ['usb', 'cable'], ['price', 'display'], ['specification', 'function'], ['fon', 'fon', 'speekars'], ['issue', 'color', 'screen'], ['update', 'oreo', 'battery', 'heating'
 'problem', 'update', 'battery', 'update', 'oreo'], ['one', 'sim', 'customer', 'service'], ['performance', 'battery'], ['camera', 'backup', 'pricefull', 'passa', 'wasole'], ['performance', 'signal', 'restarts',
'bcoms', 'plzz', 'dont'], ['round', 'performance'], ['rs', 'span', 'day', 'trust', 'deal', 'amazon'], ['disappointment', 'signal', 'problem', 'headache', 'problem', 'call', 'range'], ['rate', 'camera', 'quality
], ['price', 'feature'], ['price', 'quality', 'camera'], ['min', 'quality'], ['issue'], ['sell', 'take', 'hr', 'charge'], ['ekdam', 'network', 'problem', 'camera', 'quality', 'performance'], ['performance', 'rm
, 'memory'], ['featurs', 'battery'], ['value', 'money'], ['problem', 'speaker', 'breakup', 'problem', 'hour'], ['heating', 'problem', 'choice'], ['battery', 'standby', 'problem'], ['battery', 'camera', 'jet', '
peed', 'apps']]
First creating the id2word Dictionary and corpus of words required for the LDA topic model
[[(0, 1), (1, 1)], [(2, 1), (3, 1), (4, 2), (5, 1), (6, 1), (7, 2), (8, 1), (9, 1), (10, 1), (11, 1)], [(12, 1)], [(2, 2), (13, 1), (14, 1)], [(15, 1), (16, 1)], [(17, 1), (18, 1)], [(4, 1), (19, 1)], [(14, 2),
(20, 1), (21, 1), (22, 1), (23, 1)], [(24, 1), (25, 1), (26, 1), (27, 1)], [(28, 1)], [(5, 1), (29, 1), (30, 1)], [(4, 1), (31, 1), (32, 1)], [(4, 1), (14, 2), (30, 1), (32, 1), (33, 1)], [(3, 1), (34, 1), (35,
1), (36, 1)], [(37, 1)], [(38, 1)], [(2, 1), (39, 1), (40, 1), (41, 1), (42, 1), (43, 1), (44, 2), (45, 1), (46, 1)], [(4, 2), (32, 1), (47, 1)], [(48, 1)], [(14, 1), (49, 1), (50, 1)]]
```

Barplot to visualize the 100 most common words using FreqDist and barplots:



Task 8: Create a topic model using LDA on the cleaned-up data with 12 topics.:

```
create LDA Model
[(0,
  '0.157*"update" + 0.106*"waste" + 0.046*"smartphone" + 0.033*"bug" + '
  '0.032*"hand" + 0.029*"function" + 0.029*"cost" + 0.028*"class" + '
  '0.025*"connection" + 0.024*"mi"'),
 (1,
  '0.273*"camera" + 0.108*"quality" + 0.073*"day" + 0.030*"mode" + '
  '0.028*"processor" + 0.027*"speaker" + 0.021*"use" + 0.021*"music" + '
  '0.019*"usage" + 0.015*"thing"'),
 (2,
  '0.123*"amazon" + 0.080*"range" + 0.077*"software" + 0.042*"support" + '
  '0.042*"mark" + 0.040*"game" + 0.039*"dolby" + 0.030*"refund" + '
  '0.026*"power" + 0.026*"expectation"'),
 (3,
  '0.149*"screen" + 0.128*"device" + 0.089*"option" + 0.044*"sensor" + '
  '0.041*"model" + 0.040*"user" + 0.034*"cast" + 0.028*"set" + 0.018*"someone" '
  '+ 0.016*"interface"'),
 (4,
  '0.103*"charger" + 0.100*"heat" + 0.077*"charge" + 0.066*"lot" + 0.058*"bit" '
  '+ 0.056*"turbo" + 0.048*"hr" + 0.046*"budget" + 0.042*"system" + '
  '0.038*"slot"'),
 (5,
  '0.333*"issue" + 0.150*"money" + 0.069*"delivery" + 0.051*"value" + '
  '0.038*"light" + 0.026*"application" + 0.025*"week" + 0.023*"worth" + '
  '0.021*"brand" + 0.019*"color"'),
 (6,
  '0.259*"feature" + 0.085*"sim" + 0.076*"speed" + 0.055*"ram" + 0.050*"apps" '
  '+ 0.039*"contact" + 0.029*"gallery" + 0.026*"mp" + 0.020*"response" + '
  '0.017*"one"'),
 (7,
  '0.125*"service" + 0.051*"replacement" + 0.050*"glass" + 0.035*"button" + '
  '0.035*"purchase" + 0.033*"touch" + 0.033*"piece" + 0.028*"number" + '
  '0.027*"gorilla" + 0.024*"wifi"'),
 (8,
  '0.138*"time" + 0.087*"network" + 0.074*"call" + 0.034*"customer" + '
  '0.033*"app" + 0.032*"charging" + 0.025*"card" + 0.022*"min" + '
  '0.022*"signal" + 0.021*"volta"'),
 (9,
  '0.302*"battery" + 0.160*"problem" + 0.060*"heating" + 0.053*"month" + '
  '0.053*"backup" + 0.042*"hour" + 0.033*"life" + 0.020*"return" + '
  '0.019*"work" + 0.018*"data"'),
 (10,
  '0.254*"price" + 0.104*"display" + 0.077*"video" + 0.076*"everything" + '
  '0.041*"box" + 0.037*"flash" + 0.036*"headphone" + 0.031*"killer" + '
  '0.019*"nice" + 0.018*"night"'),
 (11,
  '0.144*"performance" + 0.059*"sound" + 0.034*"experience" + 0.034*"drain" + '
  '0.032*"stock" + 0.030*"android" + 0.029*"review" + 0.028*"photo" + '
  '0.028*"key" + 0.027*"clarity"')]
```

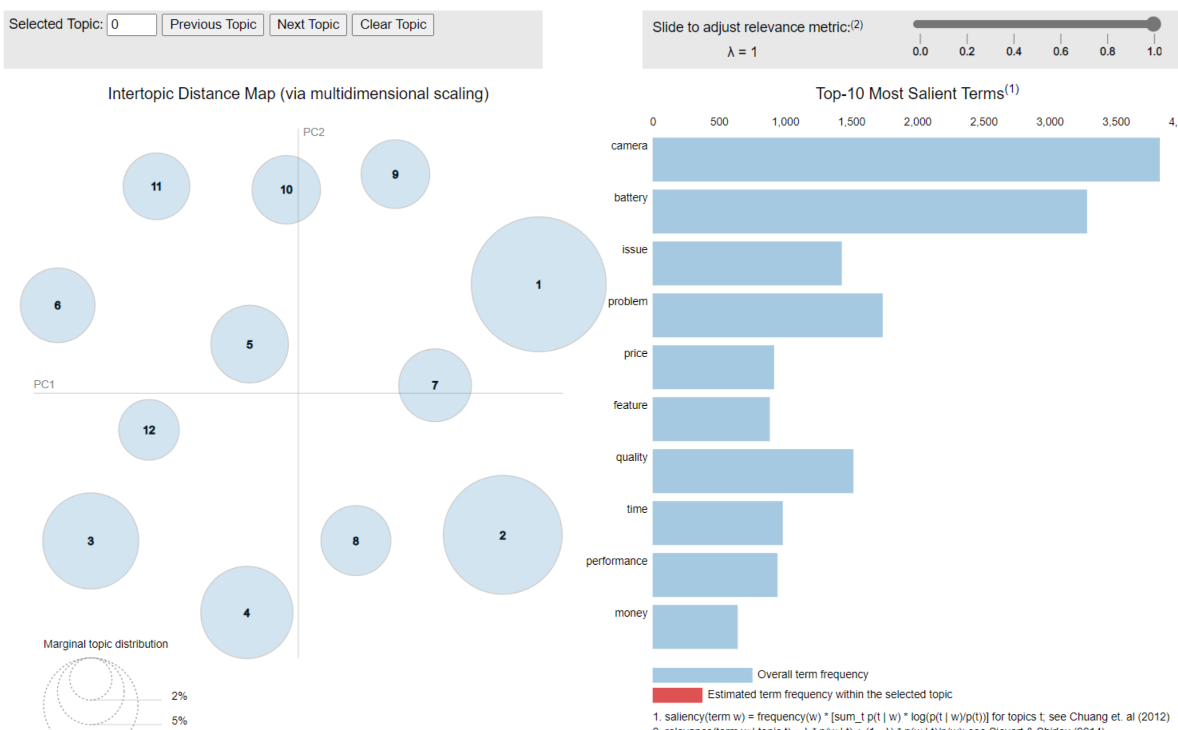
Print out the top terms for each topic.:

```
Top terms for each topic.
0 ['update', 'waste', 'smartphone', 'bug', 'hand', 'function', 'cost', 'class', 'connection', 'mi']
1 ['camera', 'quality', 'day', 'mode', 'processor', 'speaker', 'use', 'music', 'usage', 'thing']
2 ['amazon', 'range', 'software', 'support', 'mark', 'game', 'dolby', 'refund', 'power', 'expectation']
3 ['screen', 'device', 'option', 'sensor', 'model', 'user', 'cast', 'set', 'someone', 'interface']
4 ['charger', 'heat', 'charge', 'lot', 'bit', 'turbo', 'hr', 'budget', 'system', 'slot']
5 ['issue', 'money', 'delivery', 'value', 'light', 'application', 'week', 'worth', 'brand', 'color']
6 ['feature', 'sim', 'speed', 'ram', 'apps', 'contact', 'gallery', 'mp', 'response', 'one']
7 ['service', 'replacement', 'glass', 'button', 'purchase', 'touch', 'piece', 'number', 'gorilla', 'wifi']
8 ['time', 'network', 'call', 'customer', 'app', 'charging', 'card', 'min', 'signal', 'volta']
9 ['battery', 'problem', 'heating', 'month', 'backup', 'hour', 'life', 'return', 'work', 'data']
10 ['price', 'display', 'video', 'everything', 'box', 'flash', 'headphone', 'killer', 'nice', 'night']
11 ['performance', 'sound', 'experience', 'drain', 'stock', 'android', 'review', 'photo', 'key', 'clarity']
```

coherence of the model with the c_v metric?:

```
Coherence Score:  0.40989000751047 98
```

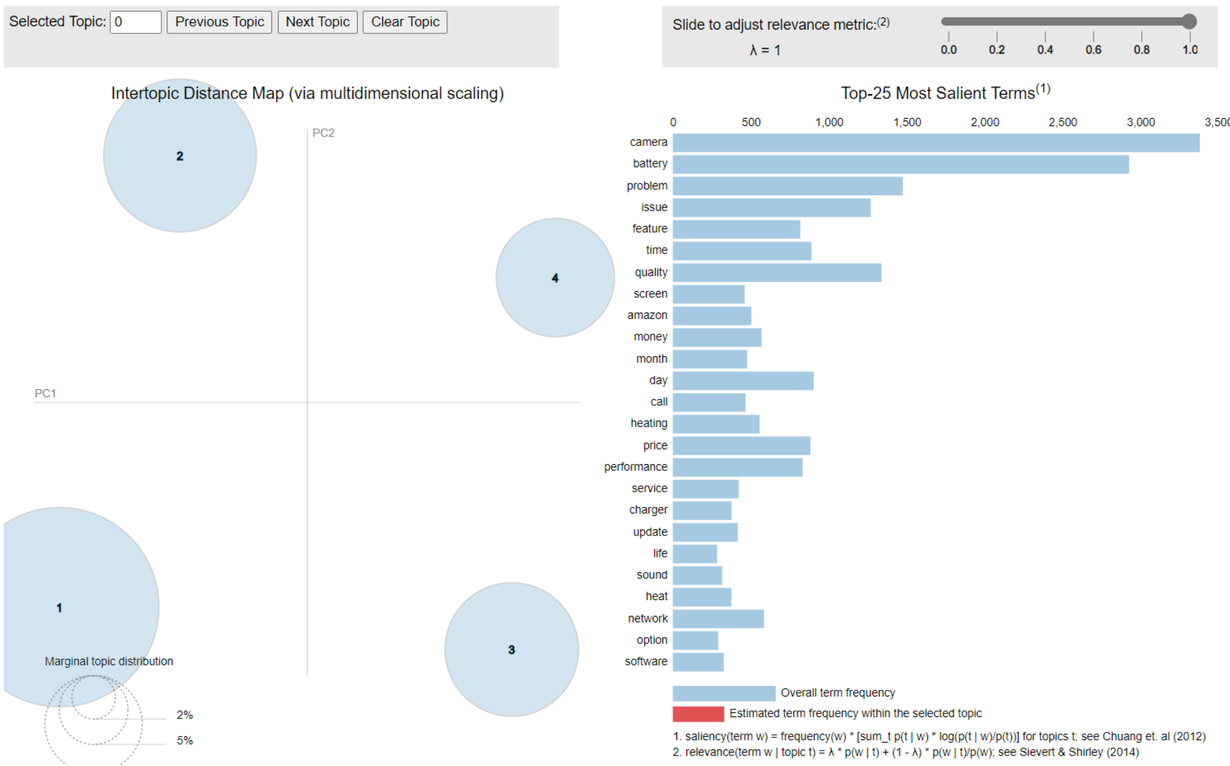
Task 9 : Analyze the topics through the business lens.:



Determine which of the topics can be combined:

```
---------------------------------------------------------------------------------------
As per the LDA model with 12 topics many of these can be combined as per below . The ideal number of topics would be 4

New Topics                                  Current LDA model Topics                     Key Words for new topic classification

Sale and Customer support                           3,8                 Amazon, service, support, replacement , refund, purchase, expectation, gorilla , glass, button, power, range, software, game

Daily usage experience                              2,9                 Camera, quality, day, time, use, usage, time, network, call, signal, volta, music, speaker, processor, app, charging

Phone features and performance                              4,7,11,12                    Feature, performance, speed, ram, price, sim, sound, experience, display, screen, video, stock, android, user, inte
rface, apps, response, contact, gallery, photo, flash, mp, sensor, clarity

Problems/issues and Pricing                                 1,5,6,10                              Issue, problem, waste, update, bug, function, battery, backup, hour, hr, life, charger, charge, heat, heati
ng, money, value, worth, cost, budget
```

Task 10: Create topic model using LDA with what you think is the optimal number of topics:



What is the coherence of the model?:

```
Coherence Score:  0.510218984012
----------------------------------------
```

Task 11: The business should be able to interpret the topics.:

```
0 ['problem', 'issue', 'time', 'money', 'heating', 'update', 'heat', 'software', 'charge', 'waste']
1 ['camera', 'battery', 'quality', 'day', 'price', 'performance', 'network', 'backup', 'device', 'hour']
2 ['feature', 'amazon', 'month', 'call', 'service', 'charger', 'sound', 'option', 'delivery', 'bit']
3 ['screen', 'life', 'turbo', 'charging', 'ram', 'work', 'budget', 'glass', 'card', 'sensor']
    Topic 1      Topic 2   Topic 3   Topic 4
0   problem       camera   feature    screen
1     issue      battery    amazon      life
2      time      quality     month     turbo
3     money          day      call  charging
4   heating        price   service       ram
5    update  performance   charger      work
6      heat      network     sound    budget
7  software       backup    option     glass
8    charge       device  delivery      card
9     waste         hour       bit    sensor
```

Create a table with the topic name and the top 10 terms in each to present to the business.:

Create a table with the topic name and the top 10 terms in each to present to the business.

	Problems and Issues	Key features for user	Sales and customer service	Hardware specs and value features
0	problem	camera	feature	screen
1	issue	battery	amazon	life
2	time	quality	month	turbo
3	money	day	call	charging
4	heating	price	service	ram
5	update	performance	charger	work
6	heat	network	sound	budget
7	software	backup	option	glass
8	charge	device	delivery	card
9	waste	hour	bit	sensor