Topic Analysis of Review Data.

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.

In [ ]:
```
#import libraries
```

In [33]:
```
import numpy as np, pandas as pd
import re, random, os, string

from pprint import pprint #pretty print
import matplotlib.pyplot as plt
%matplotlib inline

from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

import nltk
# nltk.download('punkt')
# nltk.download('averaged_perceptron_tagger')
# nltk.download('tagsets')
# nltk.download('wordnet')
# nltk.download('omw-1.4')
# nltk.download('stopwords')

import warnings
warnings.filterwarnings('ignore')
import os
```

```
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\david\AppData\Roaming\nltk_data...
```

In [2]:
```
for each in os.listdir():
    print(each)
```

```
.ipynb_checkpoints
1580822492_1570782847_proj1
1580822492_1570782847_proj1.zip
Amazon Reviews LDA Topic Modelling Project-20231124T194448Z-001
Amazon Reviews LDA Topic Modelling Project-20231124T194448Z-001.zip
K8 Reviews v0.2.csv
Untitled.ipynb
```

In [6]:
```
df = pd.read_csv("K8 Reviews v0.2.csv")
df.head()
```

Out[6]:

	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...

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

In [7]:
```
df_lower = [sent.lower() for sent in df.review.values]
df_lower[0]
```

Out[7]:
```
'good but need updates and improvements'
```

3.Tokenize the reviews using NLTKs word_tokenize function.

In [12]:
```
df_token = [word_tokenize(sent) for sent in df_lower]
df_token[0]
```

Out[12]:
```
['good', 'but', 'need', 'updates', 'and', 'improvements']
```

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

In [16]:
```
nltk.pos_tag(df_token[0])
```

Out[16]:
```
[('good', 'JJ'),
 ('but', 'CC'),
 ('need', 'VBP'),
 ('updates', 'NNS'),
 ('and', 'CC'),
 ('improvements', 'NNS')]
```

In [18]:
```
df_tagged = [nltk.pos_tag(tokens) for tokens in df_token]
df_tagged[0]
```

Out[18]:
```
[('good', 'JJ'),
 ('but', 'CC'),
 ('need', 'VBP'),
 ('updates', 'NNS'),
 ('and', 'CC'),
 ('improvements', 'NNS')]
```

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

```
5.1 Find out all the POS tags that correspond to nouns.
5.2 Limit the data to only terms with these tags.
```

In [22]:
```
nltk.help.upenn_tagset()
```

```
$: dollar
    $ -$ --$ A$ C$ HK$ M$ NZ$ S$ U.S.$ US$
'': closing quotation mark
    ' ''
(: opening parenthesis
    ( [ {
): closing parenthesis
    ) ] }
,: comma
    ,
--: dash
    --
.: sentence terminator
    . ! ?
:: colon or ellipsis
    : ; ...
CC: conjunction, coordinating
    & 'n and both but either et for less minus neither nor or plus so
    therefore times v. versus vs. whether yet
CD: numeral, cardinal
    mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one forty-
    seven 1987 twenty '79 zero two 78-degrees eighty-four IX '60s .025
    fifteen 271,124 dozen quintillion DM2,000 ...
DT: determiner
    all an another any both del each either every half la many much nary
    neither no some such that the them these this those
EX: existential there
    there
FW: foreign word
    gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous
    lutihaw alai je jour objets salutaris fille quibusdam pas trop Monte
    terram fiche oui corporis ...
IN: preposition or conjunction, subordinating
    astride among uppon whether out inside pro despite on by throughout
    below within for towards near behind atop around if like until below
    next into if beside ...
JJ: adjective or numeral, ordinal
    third ill-mannered pre-war regrettable oiled calamitous first separable
    ectoplasmic battery-powered participatory fourth still-to-be-named
    multilingual multi-disciplinary ...
JJR: adjective, comparative
    bleaker braver breezier briefer brighter brisker broader bumper busier
    calmer cheaper choosier cleaner clearer closer colder commoner costlier
    cozier creamier crunchier cuter ...
```

```
JJS: adjective, superlative
    calmest cheapest choicest classiest cleanest clearest closest commonest
    corniest costliest crassest creepiest crudest cutest darkest deadliest
    dearest deepest densest dinkiest ...
LS: list item marker
    A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002 SP-44005
    SP-44007 Second Third Three Two * a b c d first five four one six three
    two
MD: modal auxiliary
    can cannot could couldn't dare may might must need ought shall should
    shouldn't will would
NN: noun, common, singular or mass
    common-carrier cabbage knuckle-duster Casino afghan shed thermostat
    investment slide humour falloff slick wind hyena override subhumanity
    machinist ...
NNP: noun, proper, singular
    Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos
    Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA
    Shannon A.K.C. Meltex Liverpool ...
NNPS: noun, proper, plural
    Americans Americas Amharas Amityvilles Amusements Anarcho-Syndicalists
    Andalusians Andes Andruses Angels Animals Anthony Antilles Antiques
    Apache Apaches Apocrypha ...
NNS: noun, common, plural
    undergraduates scotches bric-a-brac products bodyguards facets coasts
    divestitures storehouses designs clubs fragrances averages
    subjectivists apprehensions muses factory-jobs ...
PDT: pre-determiner
    all both half many quite such sure this
POS: genitive marker
    ' 's
PRP: pronoun, personal
    hers herself him himself hisself it itself me myself one oneself ours
    ourselves ownself self she thee theirs them themselves they thou thy us
PRP$: pronoun, possessive
    her his mine my our ours their thy your
RB: adverb
    occasionally unabatingly maddeningly adventurously professedly
    stirringly prominently technologically magisterially predominately
    swiftly fiscally pitilessly ...
RBR: adverb, comparative
    further gloomier grander graver greater grimmer harder harsher
    healthier heavier higher however larger later leaner lengthier less-
    perfectly lesser lonelier longer louder lower more ...
```

```
RBS: adverb, superlative
    best biggest bluntest earliest farthest first furthest hardest
    heartiest highest largest least less most nearest second tightest worst
RP: particle
    aboard about across along apart around aside at away back before behind
    by crop down ever fast for forth from go high i.e. in into just later
    low more off on open out over per pie raising start teeth that through
    under unto up up-pp upon whole with you
SYM: symbol
    % & ' '' ''. ) ). * + ,. < = > @ A[fj] U.S U.S.S.R * ** ***
TO: "to" as preposition or infinitive marker
    to
UH: interjection
    Goodbye Goody Gosh Wow Jeepers Jee-sus Hubba Hey Kee-reist Oops amen
    huh howdy uh dammit whammo shucks heck anyways whodunnit honey golly
    man baby diddle hush sonuvabitch ...
VB: verb, base form
    ask assemble assess assign assume atone attention avoid bake balkanize
    bank begin behold believe bend benefit bevel beware bless boil bomb
    boost brace break bring broil brush build ...
VBD: verb, past tense
    dipped pleaded swiped regummed soaked tidied convened halted registered
    cushioned exacted snubbed strode aimed adopted belied figgered
    speculated wore appreciated contemplated ...
VBG: verb, present participle or gerund
    telegraphing stirring focusing angering judging stalling lactating
    hankerin' alleging veering capping approaching traveling besieging
    encrypting interrupting erasing wincing ...
VBN: verb, past participle
    multihulled dilapidated aerosolized chaired languished panelized used
    experimented flourished imitated reunifed factored condensed sheared
    unsettled primed dubbed desired ...
VBP: verb, present tense, not 3rd person singular
    predominate wrap resort sue twist spill cure lengthen brush terminate
    appear tend stray glisten obtain comprise detest tease attract
    emphasize mold postpone sever return wag ...
VBZ: verb, present tense, 3rd person singular
    bases reconstructs marks mixes displeases seals carps weaves snatches
    slumps stretches authorizes smolders pictures emerges stockpiles
    seduces fizzes uses bolsters slaps speaks pleads ...
WDT: WH-determiner
    that what whatever which whichever
WP: WH-pronoun
    that what whatever whatsoever which who whom whosoever
```

```
WP$: WH-pronoun, possessive
    whose
WRB: Wh-adverb
    how however whence whenever where whereby whereever wherein whereof why
``: opening quotation mark
    ` ``
```

In [28]:

```
df_noun=[]
for sent in df_tagged:
    df_noun.append([token for token in sent if re.search("NN.*", token[1])])
df_noun[0]
```

Out[28]:

```
[('updates', 'NNS'), ('improvements', 'NNS')]
```

6.Lemmatize.

```
6.1 Different forms of the terms need to be treated as one.
6.2 No need to provide POS tag to lemmatizer for now.
```

In [34]:

```
lemm = WordNetLemmatizer()
df_lemm=[]
for sent in df_noun:
    df_lemm.append([lemm.lemmatize(word[0]) for word in sent])
```

In [35]:

```
df_lemm[0]
```

Out[35]:

```
['update', 'improvement']
```

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

In [45]:

```
from string import punctuation
from nltk.corpus import stopwords

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

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\david\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

In [47]:

```
stop_updated = stop_nltk + list(punctuation) + ["..."] + [".."]
reviews_sw_removed=[]
```

```
for sent in df_lemm:
    reviews_sw_removed.append([term for term in sent if term not in stop_updated])
```

In [50]:
```
reviews_sw_removed[1]
```

Out[50]:
```
['mobile',
 'battery',
 'hell',
 'backup',
 'hour',
 'us',
 'idle',
 'discharged.this',
 'lie',
 'amazon',
 'lenove',
 'battery',
 'charger',
 'hour']
```

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

8.1 Print out the top terms for each topic. 8.2 What is the coherence of the model with the c_v metric?

In [53]:
```
import gensim
import gensim.corpora as corpora
from gensim.models import CoherenceModel
from gensim.models import ldamodel
```

In [52]:
```
id2word = corpora.Dictionary(reviews_sw_removed)
texts = reviews_sw_removed
corpus = [id2word.doc2bow(text) for text in texts]
```

In [54]:
```
print(corpus[200])
```

```
[(36, 1), (143, 1), (314, 1), (415, 1), (416, 1)]
```

In [55]:
```
lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                           id2word=id2word,
                                           num_topics=12,
                                           random_state=42,
                                           passes=10,
                                           per_word_topics=True)
```

In [56]:

```
pprint(lda_model.print_topics())
```

```
[(0,
  '0.167*"mobile" + 0.049*"screen" + 0.034*"call" + 0.028*"option" + '
  '0.028*"video" + 0.025*"feature" + 0.019*"music" + 0.018*"app" + '
  '0.017*"cast" + 0.016*"sensor"'),
 (1,
  '0.066*"delivery" + 0.050*"superb" + 0.050*"glass" + 0.048*"h" + '
  '0.031*"device" + 0.030*"thanks" + 0.027*"super" + 0.026*"slot" + '
  '0.026*"gorilla" + 0.024*"card"'),
 (2,
  '0.151*"note" + 0.094*"lenovo" + 0.078*"k8" + 0.017*"device" + 0.015*"model" '
  '+ 0.015*"system" + 0.012*"atmos" + 0.011*"version" + 0.010*"power" + '
  '0.010*"k4"'),
 (3,
  '0.230*"problem" + 0.117*"...." + 0.107*"heating" + 0.097*"performance" + '
  '0.088*"battery" + 0.049*"....." + 0.022*"issue" + 0.016*"hang" + '
  '0.013*"awesome" + 0.011*"cell"'),
 (4,
  '0.188*"battery" + 0.077*"phone" + 0.046*"charger" + 0.044*"hour" + '
  '0.036*"backup" + 0.035*"heat" + 0.035*"day" + 0.034*"life" + 0.031*"charge" '
  '+ 0.023*"hai"'),
 (5,
  '0.122*"price" + 0.104*"money" + 0.062*"value" + 0.058*"handset" + '
  '0.045*"range" + 0.043*"feature" + 0.034*"mobile" + 0.028*"please" + '
  '0.021*"pls" + 0.018*"experience"'),
 (6,
  '0.098*"speaker" + 0.074*"sound" + 0.071*"display" + 0.040*"work" + '
  '0.028*"month" + 0.025*"set" + 0.024*"volume" + 0.020*"class" + '
  '0.019*"purchase" + 0.017*"voice"'),
 (7,
  '0.311*"phone" + 0.081*"camera" + 0.033*"price" + 0.026*"performance" + '
  '0.023*"feature" + 0.020*"mode" + 0.017*"processor" + 0.014*"range" + '
  '0.013*"budget" + 0.012*"depth"'),
 (8,
  '0.303*"camera" + 0.197*"quality" + 0.078*"battery" + 0.035*"everything" + '
  '0.025*"mark" + 0.024*"backup" + 0.023*"clarity" + 0.019*"expectation" + '
  '0.019*"smartphone" + 0.015*"photo"'),
 (9,
  '0.136*"issue" + 0.091*"phone" + 0.046*"network" + 0.044*"update" + '
  '0.037*"software" + 0.029*"lot" + 0.023*"time" + 0.020*"battery" + '
  '0.018*"star" + 0.015*"review"'),
 (10,
  '0.102*"phone" + 0.054*"service" + 0.052*"amazon" + 0.031*"day" + '
  '0.030*"problem" + 0.029*"time" + 0.023*"sim" + 0.023*"customer" + '
  '0.021*"call" + 0.021*"replacement"'),
```

```
 (11,
  '0.477*"product" + 0.057*"waste" + 0.049*"money" + 0.022*"worth" + '
  '0.020*"headphone" + 0.020*"excellent" + 0.017*"plz" + 0.015*"amazon" + '
  '0.014*"item" + 0.012*"result"')]
```

In [57]:
```
coherence_model_lda = CoherenceModel(model=lda_model, texts=reviews_sw_removed, dictionary=id2word, coherence='c_v')
coherence_lda = coherence_model_lda.get_coherence()
print('\nCoherence Score: ', coherence_lda)
```

```
Coherence Score:  0.5572093987253456
```

9.Analyze the topics through the business lens.

```
a. Determine which of the topics can be combined.
```

In [ ]:
```
# you can assume that if a pair of topics has very similar top terms, they are very close and can be combined
```

In [ ]:
```
#to get top words count
```

In [61]:
```
from collections import Counter
```

In [64]:
```
term_list = []
for sent in reviews_sw_removed:
    term_list.extend(sent)
```

In [65]:
```
res = Counter(term_list)
res.most_common(100)
```

```
Out[65]: [('phone', 7007),
 ('camera', 3273),
 ('battery', 3143),
 ('product', 2261),
 ('problem', 1565),
 ('mobile', 1517),
 ('issue', 1490),
 ('quality', 1387),
 ('note', 1163),
 ('lenovo', 1003),
 ('time', 1003),
 ('performance', 952),
 ('price', 924),
 ('day', 897),
 ('feature', 841),
 ('backup', 661),
 ('money', 642),
 ('k8', 619),
 ('....', 618),
 ('amazon', 582),
 ('heating', 570),
 ('screen', 549),
 ('network', 515),
 ('hour', 506),
 ('month', 506),
 ('service', 506),
 ('call', 480),
 ('charger', 462),
 ('device', 446),
 ('option', 390),
 ('update', 383),
 ('range', 365),
 ('speaker', 361),
 ('sound', 361),
 ('display', 347),
 ('mode', 342),
 ('life', 339),
 ('use', 337),
 ('experience', 325),
 ('heat', 314),
 ('lot', 312),
 ('processor', 307),
 ('charge', 301),
 ('software', 301),
```

```
('waste', 286),
('thing', 280),
('sim', 267),
('.....', 250),
('value', 247),
('drain', 240),
('video', 230),
('game', 229),
('charging', 227),
('speed', 225),
('music', 225),
('everything', 225),
('ram', 222),
('app', 218),
('usage', 217),
('delivery', 216),
('glass', 216),
('customer', 213),
('handset', 211),
('turbo', 210),
('hai', 203),
('review', 196),
('hr', 194),
('work', 194),
('budget', 188),
('apps', 186),
('please', 184),
('photo', 180),
('data', 177),
('depth', 173),
('look', 173),
('mark', 171),
('system', 170),
('replacement', 168),
('picture', 167),
('return', 167),
('android', 166),
('dolby', 163),
('power', 160),
('star', 159),
('cast', 158),
('purchase', 157),
('stock', 156),
('superb', 153),
```

```
 ('smartphone', 149),
 ('support', 149),
 ('center', 148),
 ('week', 147),
 ('headphone', 146),
 ('box', 144),
 ('card', 143),
 ('earphone', 141),
 ('bit', 141),
 ('hang', 140),
 ('light', 140),
 ('front', 140)]
```

In [60]:

```
pprint(lda_model.print_topics())
```

```
[(0,
  '0.167*"mobile" + 0.049*"screen" + 0.034*"call" + 0.028*"option" + '
  '0.028*"video" + 0.025*"feature" + 0.019*"music" + 0.018*"app" + '
  '0.017*"cast" + 0.016*"sensor"'),
 (1,
  '0.066*"delivery" + 0.050*"superb" + 0.050*"glass" + 0.048*"h" + '
  '0.031*"device" + 0.030*"thanks" + 0.027*"super" + 0.026*"slot" + '
  '0.026*"gorilla" + 0.024*"card"'),
 (2,
  '0.151*"note" + 0.094*"lenovo" + 0.078*"k8" + 0.017*"device" + 0.015*"model" '
  '+ 0.015*"system" + 0.012*"atmos" + 0.011*"version" + 0.010*"power" + '
  '0.010*"k4"'),
 (3,
  '0.230*"problem" + 0.117*"...." + 0.107*"heating" + 0.097*"performance" + '
  '0.088*"battery" + 0.049*"....." + 0.022*"issue" + 0.016*"hang" + '
  '0.013*"awesome" + 0.011*"cell"'),
 (4,
  '0.188*"battery" + 0.077*"phone" + 0.046*"charger" + 0.044*"hour" + '
  '0.036*"backup" + 0.035*"heat" + 0.035*"day" + 0.034*"life" + 0.031*"charge" '
  '+ 0.023*"hai"'),
 (5,
  '0.122*"price" + 0.104*"money" + 0.062*"value" + 0.058*"handset" + '
  '0.045*"range" + 0.043*"feature" + 0.034*"mobile" + 0.028*"please" + '
  '0.021*"pls" + 0.018*"experience"'),
 (6,
  '0.098*"speaker" + 0.074*"sound" + 0.071*"display" + 0.040*"work" + '
  '0.028*"month" + 0.025*"set" + 0.024*"volume" + 0.020*"class" + '
  '0.019*"purchase" + 0.017*"voice"'),
 (7,
  '0.311*"phone" + 0.081*"camera" + 0.033*"price" + 0.026*"performance" + '
  '0.023*"feature" + 0.020*"mode" + 0.017*"processor" + 0.014*"range" + '
  '0.013*"budget" + 0.012*"depth"'),
 (8,
  '0.303*"camera" + 0.197*"quality" + 0.078*"battery" + 0.035*"everything" + '
  '0.025*"mark" + 0.024*"backup" + 0.023*"clarity" + 0.019*"expectation" + '
  '0.019*"smartphone" + 0.015*"photo"'),
 (9,
  '0.136*"issue" + 0.091*"phone" + 0.046*"network" + 0.044*"update" + '
  '0.037*"software" + 0.029*"lot" + 0.023*"time" + 0.020*"battery" + '
  '0.018*"star" + 0.015*"review"'),
 (10,
  '0.102*"phone" + 0.054*"service" + 0.052*"amazon" + 0.031*"day" + '
  '0.030*"problem" + 0.029*"time" + 0.023*"sim" + 0.023*"customer" + '
  '0.021*"call" + 0.021*"replacement"'),
```

```
(11,
  '0.477*"product" + 0.057*"waste" + 0.049*"money" + 0.022*"worth" + '
  '0.020*"headphone" + 0.020*"excellent" + 0.017*"plz" + 0.015*"amazon" + '
  '0.014*"item" + 0.012*"result"')]
```

Topic 7 and 5 possibly talks about 'pricing'

Topic 3, 4 and 9 closely talks about 'battery related issues'

Topic 3 and 7 vaguely talks about 'performance

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

```
- What is the coherence of the model?
```

In [94]:
```
# Build LDA model
lda_model9 = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                           id2word=id2word,
                                           num_topics=9,
                                           random_state=42,
                                           passes=10,
                                           per_word_topics=True)


#Coherence Score
coherence_model_lda = CoherenceModel(model=lda_model9, texts=reviews_sw_removed, dictionary=id2word, coherence='c_v')
coherence_lda = coherence_model_lda.get_coherence()
print('\nCoherence Score: ', coherence_lda)
```

```
Coherence Score:  0.5746914918351291
```

In [ ]:
```
# The coherence is now 0.57 which is a significant increase from 0.55 previously.
```

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

```
- Name each of the identified topics.

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

In [97]:
```
x = lda_model9.show_topics(formatted=False)
topics_words = [(tp[0], [wd[0] for wd in tp[1]]) for tp in x]
```

In [98]:

```
for topic,words in topics_words:
    print(str(topic)+ "::"+ str(words))
print()
```

```
0::['mobile', 'feature', 'screen', 'call', 'option', 'video', 'app', 'music', 'apps', 'cast']
1::['delivery', 'return', 'glass', 'h', 'device', 'amazon', 'policy', 'super', 'gorilla', 'volta']
2::['phone', 'note', 'lenovo', 'k8', 'time', 'issue', 'service', 'day', 'problem', 'network']
3::['problem', 'issue', 'battery', 'phone', 'heating', 'performance', 'camera', 'network', 'update', 'drain']
4::['battery', 'charger', 'hour', 'backup', 'heat', 'charge', 'phone', 'hai', 'charging', 'turbo']
5::['product', 'money', 'waste', 'value', 'handset', 'price', 'amazon', 'experience', 'lenovo', 'plz']
6::['speaker', 'superb', '.....', 'display', 'mobile', 'sound', 'work', '......', 'set', 'item']
7::['phone', 'camera', 'price', 'battery', 'quality', 'performance', 'feature', 'range', 'mode', 'processor']
8::['camera', 'quality', '....', 'battery', 'everything', 'clarity', 'expectation', 'headphone', 'speed', 'mark']
```

In [ ]:

```
#possible topics from terms present

#topic0 = call and video features
#Topic1 = amazon
#Topic2 = service related issues
#Topic3 = battery related issues
#Topic4 = product accessories
#Topic5 = pricing
#Topic6 = sound features
#Topic7 = overall general phone features
#Topic8 = phone performance
```

In [ ]: