In [1]:

```
import pandas as pd
import numpy as np
from nltk.tokenize import word_tokenize
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.preprocessing import LabelEncoder
from collections import defaultdict
from nltk.corpus import wordnet as wn
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import model_selection, naive_bayes, svm
from sklearn.metrics import accuracy_score
import nltk
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\wordnet.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Out[1]:

```
True
```

In [3]:

```
np.random.seed(500)
Corpus = pd.read_csv("C:/Users/Admin/Downloads/corpus.csv")
```

In [4]:

```
Corpus['text'].dropna(inplace=True)
Corpus['text'] = [entry.lower() for entry in Corpus['text']]
Corpus['text']= [word_tokenize(entry) for entry in Corpus['text']]
```

In [5]:

```
tag_map = defaultdict(lambda : wn.NOUN)
tag_map['J'] = wn.ADJ
tag_map['V'] = wn.VERB
tag_map['R'] = wn.ADV
```

In [6]:

```
for index,entry in enumerate(Corpus['text']):
  Final_words = []
  word_Lemmatized = WordNetLemmatizer()
  for word, tag in pos_tag(entry):
    if word not in stopwords.words('english') and word.isalpha():
      word_Final = word_Lemmatized.lemmatize(word,tag_map[tag[0]])
      Final_words.append(word_Final)
  Corpus.loc[index,'text_final'] = str(Final_words)
```

In [7]:

```
Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_split(Corpus['text_final'],Corpus['labe
```

In [8]:

```
Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)
```

In [9]:

```
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(Corpus['text_final'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
```

In [10]:
```
print(Tfidf_vect.vocabulary_)
```

```
{'stun': 322, 'even': 96, 'sound': 308, 'track': 346, 'beautiful': 21, 'paint': 237, 'senery': 295,
'mind': 208, 'well': 369, 'would': 383, 'recomend': 272, 'people': 241, 'hate': 152, 'video': 364,
'game': 131, 'music': 217, 'play': 248, 'chrono': 42, 'cross': 62, 'ever': 98, 'best': 26, 'back': 1
9, 'away': 18, 'crude': 63, 'keyboarding': 181, 'take': 329, 'fresh': 125, 'step': 319, 'grate': 14
4, 'guitar': 149, 'soulful': 307, 'orchestra': 230, 'impress': 167, 'anyone': 11, 'care': 39, 'liste
n': 193, 'soundtrack': 309, 'anything': 12, 'read': 267, 'lot': 198, 'review': 281, 'say': 288, 'fig
ure': 114, 'write': 384, 'disagree': 72, 'bit': 29, 'opinino': 228, 'yasunori': 387, 'mitsuda': 211,
'ultimate': 355, 'masterpiece': 204, 'timeless': 340, 'year': 388, 'beauty': 23, 'simply': 300, 'ref
use': 274, 'price': 255, 'tag': 328, 'pretty': 254, 'stagger': 315, 'must': 218, 'go': 139, 'buy': 3
5, 'cd': 40, 'much': 216, 'money': 213, 'one': 227, 'feel': 111, 'worth': 382, 'every': 99, 'penny':
240, 'amazing': 6, 'favorite': 110, 'time': 339, 'hand': 151, 'intense': 174, 'sadness': 286, 'priso
ner': 257, 'fate': 109, 'mean': 206, 'hope': 160, 'distant': 76, 'promise': 260, 'girl': 135, 'stea
l': 318, 'star': 316, 'important': 166, 'inspiration': 172, 'personally': 244, 'throughout': 338, 't
een': 330, 'high': 157, 'energy': 89, 'like': 192, 'dreamwatch': 83, 'chronomantique': 43, 'indefina
bly': 169, 'remeniscent': 278, 'trigger': 349, 'absolutely': 1, 'superb': 323, 'probably': 258, 'com
poser': 50, 'work': 380, 'hear': 154, 'xenogears': 386, 'ca': 37, 'sure': 324, 'never': 220, 'twic
e': 352, 'wish': 376, 'could': 54, 'give': 137, 'excellent': 102, 'truly': 351, 'enjoy': 91, 'relaxi
ng': 275, 'disk': 75, 'scar': 289, 'life': 190, 'death': 66, 'forest': 121, 'illusion': 164, 'fortre
ss': 123, 'ancient': 8, 'dragon': 81, 'lose': 197, 'fragment': 124, 'drown': 84, 'two': 353, 'draggo
ns': 80, 'galdorb': 129, 'home': 159, 'gale': 130, 'girlfriend': 136, 'zelbessdisk': 390, 'three': 3
37, 'garden': 132, 'god': 140, 'chronopolis': 44, 'jellyfish': 177, 'sea': 291, 'burn': 34, 'orphang
e': 231, 'prayer': 251, 'tower': 345, 'radical': 265, 'dreamer': 82, 'unstealable': 360, 'bring': 3
3, 'remember': 277, 'pull': 262, 'jaw': 175, 'floor': 118, 'know': 185, 'divine': 77, 'single': 302,
'song': 305, 'tell': 331, 'story': 321, 'good': 142, 'great': 145, 'without': 377, 'doubt': 79, 'mag
ical': 202, 'wind': 375, 'unstolen': 361, 'jewel': 178, 'translation': 347, 'varies': 363, 'perfec
t': 242, 'ask': 14, 'pour': 250, 'heart': 156, 'paper': 238, 'absolute': 0, 'quite': 264, 'actuall
y': 3, 'least': 187, 'heard': 155, 'whether': 370, 'aware': 17, 'contribute': 52, 'greatly': 146, 'm
ood': 214, 'minute': 209, 'whole': 372, 'exact': 101, 'count': 55, 'impressively': 168, 'remarkabl
e': 276, 'assure': 15, 'forget': 122, 'everything': 100, 'listener': 194, 'energetic': 87, 'dance':
64, 'tokage': 342, 'termina': 334, 'slow': 303, 'haunting': 153, 'purely': 263, 'beautifully': 22,
'compose': 49, 'fantastic': 107, 'vocal': 366, 'videogame': 365, 'soundtracks': 310, 'surely': 325,
'buyer': 36, 'beware': 27, 'book': 30, 'want': 367, 'paragraph': 239, 'haddon': 150, 'family': 106,
'friend': 126, 'perhaps': 243, 'imagine': 165, 'thing': 335, 'spend': 314, 'evening': 97, 'hysteri
c': 163, 'piece': 246, 'another': 10, 'definitely': 68, 'bad': 20, 'enough': 92, 'enter': 93, 'kin
d': 184, 'contest': 51, 'believe': 25, 'amazon': 7, 'sell': 293, 'maybe': 205, 'offer': 226, 'grad
e': 143, 'term': 333, 'kill': 183, 'mockingbird': 212, 'anyway': 13, 'unless': 359, 'send': 294, 'so
meone': 304, 'joke': 179, 'far': 108, 'glorious': 138, 'love': 200, 'whisper': 371, 'wicked': 373,
'saint': 287, 'pleasantly': 249, 'surprise': 327, 'change': 41, 'normaly': 221, 'romance': 283, 'nov
el': 222, 'world': 381, 'rave': 266, 'brilliant': 32, 'true': 350, 'wonderful': 379, 'typical': 354,
'crime': 60, 'becuase': 24, 'miss': 210, 'warm': 368, 'five': 117, 'finish': 116, 'fell': 112, 'cara
cters': 38, 'expect': 104, 'average': 16, 'instead': 173, 'find': 115, 'think': 336, 'predict': 253,
'outcome': 232, 'shock': 299, 'writting': 385, 'descriptive': 70, 'break': 31, 'julia': 180, 'felt':
113, 'reader': 268, 'lover': 201, 'let': 189, 'cover': 58, 'fool': 119, 'spectacular': 313, 'disappo
inted': 73, 'romanian': 285, 'opinion': 229, 'bias': 28, 'angle': 9, 'europe': 95, 'clean': 46, 'pro
per': 261, 'fail': 105, 'shed': 298, 'light': 191, 'rest': 279, 'understand': 357, 'tourist': 344,
'guide': 148, 'tour': 343, 'country': 56, 'however': 161, 'use': 362, 'reference': 273, 'see': 292,
'travel': 348, 'outline': 233, 'exclude': 103, 'romania': 284, 'precision': 252, 'detail': 71, 'dk':
78, 'series': 297, 'leave': 188, 'yet': 389, 'disappointment': 74, 'lonely': 195, 'planet': 247, 'fu
ll': 128, 'picture': 245, 'info': 171, 'need': 219, 'also': 5, 'end': 86, 'indiv': 170, 'city': 45,
'scratch': 290, 'surface': 326, 'receive': 271, 'defective': 67, 'move': 215, 'germany': 133, 'get':
134, 'overview': 235, 'unfortunately': 358, 'page': 236, 'greece': 147, 'english': 90, 'look': 196,
'spanish': 312, 'sort': 306, 'printing': 256, 'problem': 259, 'highlight': 158, 'label': 186, 'memph
is': 207, 'tn': 341, 'come': 47, 'reatards': 270, 'front': 127, 'course': 57, 'wild': 374, 'kid': 18
2, 'jay': 176, 'reatard': 269, 'start': 317, 'eric': 94, 'oblivians': 223, 'goner': 141, 'tender': 3
32, 'age': 4, 'still': 320, 'rock': 282, 'offend': 225, 'sensibility': 296, 'community': 48, 'overdr
iven': 234, 'loud': 199, 'crackle': 59, 'underlay': 356, 'southern': 311, 'croon': 61, 'howl': 162,
'energize': 88, 'either': 85, 'depend': 69, 'foot': 120, 'od': 224, 'copper': 53, 'wo': 378, 'accep
t': 2, 'return': 280, 'since': 301, 'make': 203, 'deal': 65}
```

In [11]:
```
print(Train_X_Tfidf)
```

```
  (0, 385)	0.15181783245925792
  (0, 373)	0.13182822297400662
  (0, 371)	0.13182822297400662
  (0, 339)	0.08347290620977138
  (0, 336)	0.15181783245925792
  (0, 316)	0.08347290620977138
  (0, 313)	0.15181783245925792
  (0, 299)	0.15181783245925792
  (0, 287)	0.13182822297400662
  (0, 283)	0.26365644594801324
  (0, 268)	0.15181783245925792
  (0, 267)	0.29296727954779
  (0, 253)	0.15181783245925792
  (0, 232)	0.15181783245925792
  (0, 227)	0.08347290620977138
  (0, 222)	0.13182822297400662
  (0, 218)	0.11764536933451467
  (0, 201)	0.15181783245925792
  (0, 200)	0.13182822297400662
  (0, 189)	0.15181783245925792
  (0, 180)	0.15181783245925792
  (0, 173)	0.30363566491851585
  (0, 156)	0.11764536933451467
  (0, 119)	0.15181783245925792
  (0, 117)	0.15181783245925792
  :	:
  (9, 279)	0.1256205421490419
  (9, 273)	0.1446688576328972
  (9, 261)	0.1446688576328972
  (9, 252)	0.1446688576328972
  (9, 233)	0.1446688576328972
  (9, 229)	0.1446688576328972
  (9, 220)	0.11210554723202179
  (9, 191)	0.1446688576328972
  (9, 188)	0.1446688576328972
  (9, 161)	0.1446688576328972
  (9, 148)	0.2512410842980838
  (9, 122)	0.1256205421490419
  (9, 105)	0.1446688576328972
  (9, 103)	0.1446688576328972
  (9, 95)	0.2512410842980838
  (9, 78)	0.1446688576328972
  (9, 74)	0.1446688576328972
  (9, 73)	0.2893377152657944
  (9, 71)	0.1446688576328972
  (9, 56)	0.1256205421490419
  (9, 46)	0.1446688576328972
  (9, 30)	0.1716307966027709
  (9, 28)	0.1446688576328972
  (9, 11)	0.11210554723202179
  (9, 9)	0.1446688576328972
```

In [12]:
```
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
print("SVM Accuracy Score -> ",accuracy_score(predictions_SVM, Test_Y)*100)
```

```
SVM Accuracy Score ->  60.0
```