In [3]:
```
import pandas as pd
df = pd.read_csv("emotion.csv")
df.head()
```

Out[3]:

	text	label
0	i didnt feel humiliated	0
1	i can go from feeling so hopeless to so damned...	0
2	im grabbing a minute to post i feel greedy wrong	3
3	i am ever feeling nostalgic about the fireplac...	2
4	i am feeling grouchy	3

In [4]:
```
df.shape
```

Out[4]:
```
(16000, 2)
```

In [5]:
```
df.info()
```

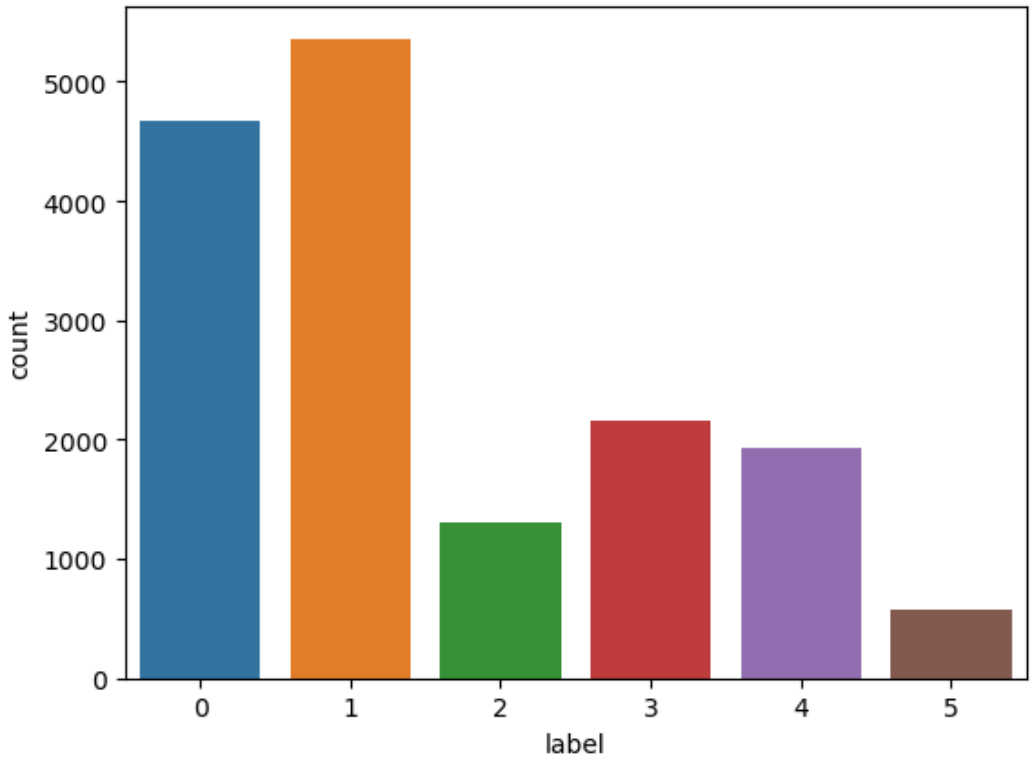
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16000 entries, 0 to 15999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    16000 non-null  object
 1   label   16000 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 250.1+ KB
```

In [6]:
```
df.label.value_counts()
```

Out[6]:
```
1    5362
0    4666
3    2159
4    1937
2    1304
5     572
Name: label, dtype: int64
```

In [7]:
```
import seaborn as sns
sns.countplot(x=df.label)
```

Out[7]:
```
<Axes: xlabel='label', ylabel='count'>
```



In [8]:
```
# checkind for missing values
df.isna().sum()
```

Out[8]:
```
text     0
label    0
dtype: int64
```

Text processing

In [14]:
```
df['text'] = df['text'].apply(lambda x: " ".join(x. lower() for x in x.split()))
```

In [11]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to C:\Users\Pooja
[nltk_data]     Reddy\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[11]:
```
True
```

In [12]:
```
from nltk.corpus import stopwords
stop = stopwords.words('english')
df['text'] = df['text'].apply(lambda x: " ".join(x for x in x.split() if x not in stop))
```

In [15]:
```
!pip install textblob
```

```
Requirement already satisfied: textblob in c:\users\pooja reddy\anaconda3\lib\site-pac
kages (0.19.0)
Requirement already satisfied: nltk>=3.9 in c:\users\pooja reddy\anaconda3\lib\site-pa
ckages (from textblob) (3.9.1)
Requirement already satisfied: click in c:\users\pooja reddy\anaconda3\lib\site-packag
es (from nltk>=3.9->textblob) (8.0.4)
Requirement already satisfied: joblib in c:\users\pooja reddy\anaconda3\lib\site-packa
ges (from nltk>=3.9->textblob) (1.2.0)
Requirement already satisfied: regex>=2021.8.3 in c:\users\pooja reddy\anaconda3\lib\s
ite-packages (from nltk>=3.9->textblob) (2022.7.9)
Requirement already satisfied: tqdm in c:\users\pooja reddy\anaconda3\lib\site-package
s (from nltk>=3.9->textblob) (4.65.0)
Requirement already satisfied: colorama in c:\users\pooja reddy\anaconda3\lib\site-pac
kages (from click->nltk>=3.9->textblob) (0.4.6)
```

In [19]:
```
from nltk.stem import WordNetLemmatizer
from textblob import Word
df['text'] = df['text'].apply(lambda x: " ".join([Word(word).lemmatize() for word in x.
df['text'].head()
```

Out[19]:
```
0                                didnt feel humiliated
1    go feeling hopeless damned hopeful around some...
2            im grabbing minute post feel greedy wrong
3    ever feeling nostalgic fireplace know still pr...
4                                      feeling grouchy
Name: text, dtype: object
```

In [20]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
X = tfidf.fit_transform(df['text'])
X = X.toarray()
y = df.label.values
```

In [21]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2,
                                                    shuffle=True, random_state = 0)
```

Model Building

In [26]:
```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model = model.fit(X_train, y_train)
pred = model.predict(X_test)
```

In [27]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [36]:
```
print(confusion_matrix(y_test, pred))
```

```
[[293 116 117 144 171  64]
 [174 388 194  96 132  69]
 [ 48  60 103  15  32  13]
 [ 84  85  53 146  58  33]
 [ 84  61  38  37 147  30]
 [ 23  16  10   5  20  41]]
```

In [33]:
```
print(accuracy_score(y_test, pred))
```

```
0.349375
```

In [34]:
```
print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

           0       0.42      0.32      0.36       905
           1       0.53      0.37      0.44      1053
           2       0.20      0.38      0.26       271
           3       0.33      0.32      0.32       459
           4       0.26      0.37      0.31       397
           5       0.16      0.36      0.22       115

    accuracy                           0.35      3200
   macro avg       0.32      0.35      0.32      3200
weighted avg       0.40      0.35      0.36      3200
```

In [*]:
```
from sklearn.ensemble import RandomForestClassifier
clf_rf= RandomForestClassifier()
clf_rf.fit(X_train,y_train)
rf_pred=clf_rf.predict(X_test).astype(int)
```

In [*]:
```
print(confusion_matrix(y_test, rf_pred))
```

In [*]:
```
print(accuracy_score(y_test, rf_pred))
```

In [*]:
```
print(classification_report(y_test, rf_pred))
```

In [*]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
lonreg.fir(X_train, y_train)
lr_pred = logreg.predict(X_test)
```

In [*]:
```
print(confusion_matrix(y_test, lr_pred))
```

In [*]:
```
print(classification_report(y_test, lr_pred))
```

In [*]:
```
print(accuracy_score(y_test, lr_pred))
```

conclusion

Random Foresst model has proved better with the accuracy 88% when compared to other two models

In [ ]: