In [2]:
```
import numpy as np #For Various Mathematical Funtions
import pandas as pd #For Data Analysis
import seaborn as sns #Statisical Data Visualisation
import matplotlib.pyplot as plt #For Basic Graphs
%matplotlib inline
import warnings, string
warnings.filterwarnings('ignore') #alert the user of some condition
from sklearn.model_selection import train_test_split, GridSearchCV #SKLEARN-Data Analysis,model_
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import nltk #Natural Language Toolkit
from nltk.corpus import stopwords #Dictionary
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer #CV for text to nu

from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier #Random Forest Algo of ML #2#11111
#CNN Approach

from sklearn.svm import SVC #Support Vector ML Algo #5#1111111111111
```

In [3]:
```
df = pd.read_csv('Preprocessed Fake Reviews Detection Dataset.csv')
df.head()
```

Out[3]:

| | Unnamed: 0 | category | rating | label | text_ |
|---|---|---|---|---|---|
| **0** | 0 | Home_and_Kitchen_5 | 5.0 | CG | love well made sturdi comfort i love veri pretti |
| **1** | 1 | Home_and_Kitchen_5 | 5.0 | CG | love great upgrad origin i 've mine coupl year |
| **2** | 2 | Home_and_Kitchen_5 | 5.0 | CG | thi pillow save back i love look feel pillow |
| **3** | 3 | Home_and_Kitchen_5 | 1.0 | CG | miss inform use great product price i |
| **4** | 4 | Home_and_Kitchen_5 | 5.0 | CG | veri nice set good qualiti we set two month |

In [4]:
```
df.drop('Unnamed: 0',axis=1,inplace=True)
```

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

Out[5]:

| | category | rating | label | text_ |
|---|---|---|---|---|
| **0** | Home_and_Kitchen_5 | 5.0 | CG | love well made sturdi comfort i love veri pretti |
| **1** | Home_and_Kitchen_5 | 5.0 | CG | love great upgrad origin i 've mine coupl year |
| **2** | Home_and_Kitchen_5 | 5.0 | CG | thi pillow save back i love look feel pillow |
| **3** | Home_and_Kitchen_5 | 1.0 | CG | miss inform use great product price i |
| **4** | Home_and_Kitchen_5 | 5.0 | CG | veri nice set good qualiti we set two month |

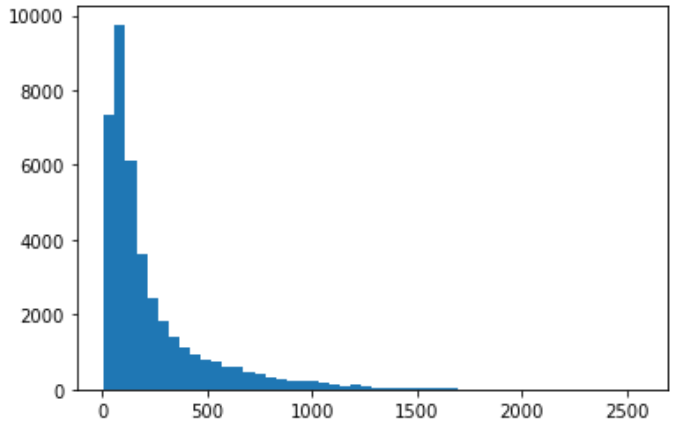
In [6]:
```
df.dropna(inplace=True)
```

In [7]:
```
df['length'] = df['text_'].apply(len)
```

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 40431 entries, 0 to 40431
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   category  40431 non-null  object 
 1   rating    40431 non-null  float64
 2   label     40431 non-null  object 
 3   text_     40431 non-null  object 
 4   length    40431 non-null  int64  
dtypes: float64(1), int64(1), object(3)
memory usage: 1.9+ MB
```

In [9]:
```
plt.hist(df['length'],bins=50)
plt.show()
```

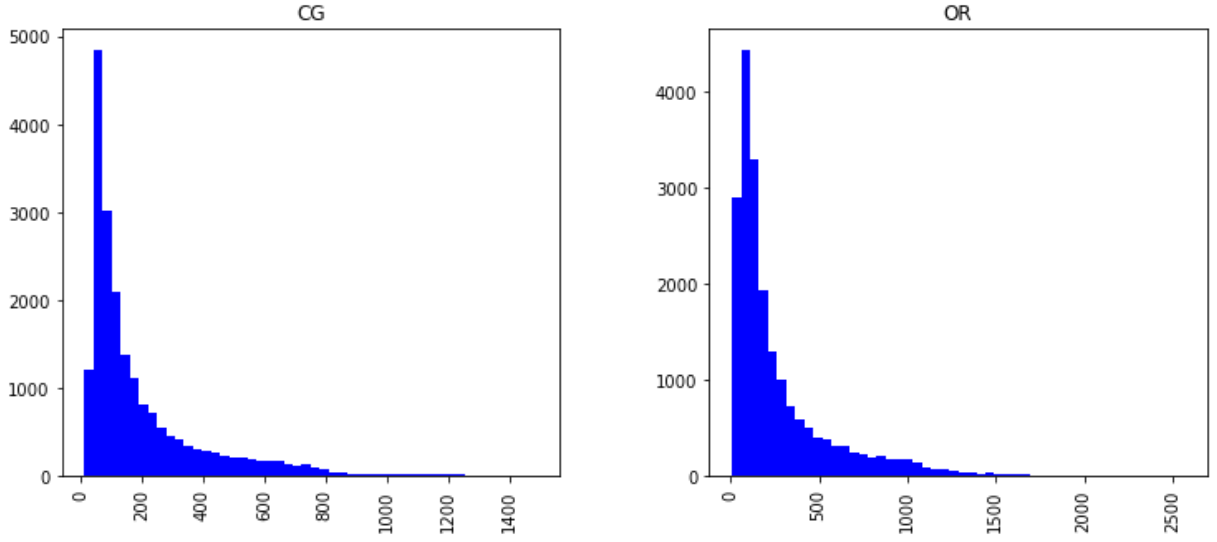



In [10]:
```
df.groupby('label').describe()
```

Out[10]:

| | rating | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | count | mean | std | min | 25% | 50% | 75% | max | count | mean | std | min | 25% | 50% |
| **label** | | | | | | | | | | | | | | |
| **CG** | 20215.0 | 4.259906 | 1.141092 | 1.0 | 4.0 | 5.0 | 5.0 | 5.0 | 20215.0 | 201.664358 | 212.330721 | 13.0 | 64.0 | 115.0 |
| **OR** | 20216.0 | 4.253265 | 1.147652 | 1.0 | 4.0 | 5.0 | 5.0 | 5.0 | 20216.0 | 270.965127 | 294.827351 | 8.0 | 83.0 | 152.0 |

In [11]:
```
df.hist(column='length',by='label',bins=50,color='blue',figsize=(12,5))
plt.show()
```




In [12]:
```
df[df['label']=='OR'][['text_','length']].sort_values(by='length',ascending=False).head().iloc[0
```

Out[12]:
```
'i compar thi rawhid to <a data-hook="product-link-linked" class="a-link-normal" href="/good-buddy-usa-rawhide-braided-sticks-for-dogs-7-to-8-inch-2-count-pack-of-1/dp/b005gwvwja/ref=cm_cr_arp_d_rvw_txt?ie=utf8">good buddi usa rawhid braid stick for dogs, 7 to 8-inch, 2 count (pack of 1)</a> for my 3 year old labradoodle. where it is made: thi is made in china. the <a data-hook="product-link-linked" class="a-link-normal" href="/good-buddy-usa-rawhide-braided-sticks-for-dogs-7-to-8-inch-2-count-pack-of-1/dp/b005gwvwja/ref=cm_cr_arp_d_rvw_txt?ie=utf8">good buddi usa rawhid braid stick for dogs, 7 to 8-inch, 2 count (pack of 1)</a> i from the usa. ingredients: made ingredi is beefhid v rawhide. rawhid is the inner layer of the hide of ani cleft-hoof bovin livestock. beef-hid is premium rawhid made from cow rais for consumption. so, thi is just anoth form of rawhid that is from cow a oppos to just ani anim with a cleft hoof. the healthi hide rawhid ha some extra potato starch, salt, soybean oil, vitamin e (100 iu), and preservatives. the healthi hide stick have almost zero smell. size: these are realli small pencil size rawhid treat (see pictures) and are too small for my 45 pound dog (medium size dog). the <a data-hook="product-link-linked" class="a-link-normal" href="/good-buddy-usa-rawhide-braided-sticks-for-dogs-7-to-8-inch-2-count-pack-of-1/dp/b005gwvwja/ref=cm_cr_arp_d_rvw_txt?ie=utf8">good buddi usa rawhid braid stick for dogs, 7 to 8-inch, 2 count (pack of 1)</a> i a veri good size for almost ani size dog. dog\' opinion: my dog doesn\'t like these rawhid stick but love the <a data-hook="product-link-linked" class="a-link-normal" href="/good-buddy-usa-rawhide-braided-sticks-for-dogs-7-to-8-inch-2-count-pack-of-1/dp/b005gwvwja/ref=cm_cr_arp_d_rvw_txt?ie=utf8">good buddi usa rawhid braid stick for dogs, 7 to 8-inch, 2 count (pack of 1)</a>. the healthi hide stick come with 20 stick but they smell like plain old rawhid and my dog just leav it lie around. the tini amount of vitamin e ad to thi stick isn\'t realli enough in my opinion to give your dog a "healthi and luxuri coat" like it promises. overall: save your money on thi one and buy the <a data-hook="product-link-linked" class="a-link-normal" href="/good-buddy-usa-rawhide-braided-sticks-for-dogs-7-to-8-inch-2-count-pack-of-1/dp/b005gwvwja/ref=cm_cr_arp_d_rvw_txt?ie=utf8">good buddi usa rawhid braid stick for dogs, 7 to 8-inch, 2 count (pack of 1)</a> instead. if you found thi review help plea give it a thumb up below! thanks!!'
```

In [13]:
```
df.length.describe()
```

Out[13]:
```
count    40431.000000
mean       236.315599
std        259.236021
min          8.000000
25%         72.000000
50%        133.000000
75%        291.000000
max       2567.000000
Name: length, dtype: float64
```

In [14]:
```
def text_process(review):
    nopunc = [char for char in review if char not in string.punctuation]
    nopunc = ''.join(nopunc)
    return [word for word in nopunc.split() if word.lower() not in stopwords.words('english')]
```

In [15]:
```
bow_transformer = CountVectorizer(analyzer=text_process)
bow_transformer
```

Out[15]:
```
CountVectorizer(analyzer=<function text_process at 0x000001EB4D502D30>)
```

In [16]:
```
bow_transformer.fit(df['text_'])
print("Total Vocabulary:",len(bow_transformer.vocabulary_))
```

```
Total Vocabulary: 39377
```

In [17]:
```
review4 = df['text_'][3]
review4
```

Out[17]:
```
'miss inform use great product price i'
```

In [18]:
```
bow_msg4 = bow_transformer.transform([review4])
print(bow_msg4)
print(bow_msg4.shape)
```

```
  (0, 15533)    1
  (0, 18102)    1
  (0, 22653)    1
  (0, 27193)    1
  (0, 27345)    1
  (0, 36860)    1
(1, 39377)
```

In [19]:
```
print(bow_transformer.get_feature_names_out()[15841])
print(bow_transformer.get_feature_names_out()[23848])
```

```
guntot
nicew
```

In [20]:
```
bow_reviews = bow_transformer.transform(df['text_'])
```

In [21]:
```
print("Shape of Bag of Words Transformer for the entire reviews corpus:",bow_reviews.shape)
print("Amount of non zero values in the bag of words model:",bow_reviews.nnz)
```

```
Shape of Bag of Words Transformer for the entire reviews corpus: (40431, 39377)
Amount of non zero values in the bag of words model: 1038987
```

In [22]:
```
print("Sparsity:",np.round((bow_reviews.nnz/(bow_reviews.shape[0]*bow_reviews.shape[1]))*100,2))
```

```
Sparsity: 0.07
```

In [23]:
```
tfidf_transformer = TfidfTransformer().fit(bow_reviews)
tfidf_rev4 = tfidf_transformer.transform(bow_msg4)
print(bow_msg4)
```

```
  (0, 15533)    1
  (0, 18102)    1
  (0, 22653)    1
  (0, 27193)    1
  (0, 27345)    1
  (0, 36860)    1
```

In [24]:
```
print(tfidf_transformer.idf_[bow_transformer.vocabulary_['mango']])
print(tfidf_transformer.idf_[bow_transformer.vocabulary_['book']])
```

```
10.91422964906803
2.839736252370183
```

In [25]:
```
tfidf_reviews = tfidf_transformer.transform(bow_reviews)
print("Shape:",tfidf_reviews.shape)
print("No. of Dimensions:",tfidf_reviews.ndim)
```

```
Shape: (40431, 39377)
No. of Dimensions: 2
```

In [26]:
```
review_train, review_test, label_train, label_test = train_test_split(df['text_'],df['label'],te
```

In [27]:
```
#RANDOM FOREST CLASSIFIER
pipeline = Pipeline([
    ('bow',CountVectorizer(analyzer=text_process)),
    ('tfidf',TfidfTransformer()),
    ('classifier',RandomForestClassifier())
])
```

In [28]:
```
pipeline.fit(review_train,label_train)
```

Out[28]:
```
Pipeline(steps=[('bow',
                 CountVectorizer(analyzer=<function text_process at 0x000001EB4D502D30>)),
                ('tfidf', TfidfTransformer()),
                ('classifier', RandomForestClassifier())])
```

In [29]:
```
rfc_pred = pipeline.predict(review_test)
rfc_pred
```

Out[29]:
```
array(['OR', 'OR', 'OR', ..., 'CG', 'CG', 'OR'], dtype=object)
```

In [30]:
```
#EVALUATION OF RANDOM FOREST CLASSIFIER
print('Classification Report:',classification_report(label_test,rfc_pred))
print('Confusion Matrix:',confusion_matrix(label_test,rfc_pred))
print('Accuracy Score:',accuracy_score(label_test,rfc_pred))
```

```
Classification Report:               precision    recall  f1-score   support

          CG       0.81      0.89      0.84      7033
          OR       0.88      0.79      0.83      7118

    accuracy                           0.84     14151
   macro avg       0.84      0.84      0.84     14151
weighted avg       0.84      0.84      0.84     14151

Confusion Matrix: [[6239  794]
 [1507 5611]]
Accuracy Score: 0.8373966504133984
```

In [31]:
```
#RESULT OF RANDOM FOREST CLASSIFIER
print('Model Prediction Accuracy:',str(np.round(accuracy_score(label_test,rfc_pred)*100,2)) + '%
```

```
Model Prediction Accuracy: 83.74%
```

In [32]:
```
#SUPPORT VECTOR
pipeline = Pipeline([
    ('bow',CountVectorizer(analyzer=text_process)),
    ('tfidf',TfidfTransformer()),
    ('classifier',SVC())
])
```

In [33]:
```
pipeline.fit(review_train,label_train)
```

Out[33]:
```
Pipeline(steps=[('bow',
                 CountVectorizer(analyzer=<function text_process at 0x000001EB4D502D30>)),
                ('tfidf', TfidfTransformer()), ('classifier', SVC())])
```

In [34]:
```
svc_pred = pipeline.predict(review_test)
svc_pred
```

Out[34]:
```
array(['OR', 'OR', 'OR', ..., 'CG', 'CG', 'OR'], dtype=object)
```

In [35]:
```
#EVALUTION OF SUPPORT VECTOR
print('Classification Report:',classification_report(label_test,svc_pred))
print('Confusion Matrix:',confusion_matrix(label_test,svc_pred))
print('Accuracy Score:',accuracy_score(label_test,svc_pred))
```

```
Classification Report:               precision    recall  f1-score   support

          CG       0.89      0.86      0.88      7033
          OR       0.87      0.90      0.88      7118

    accuracy                           0.88     14151
   macro avg       0.88      0.88      0.88     14151
weighted avg       0.88      0.88      0.88     14151

Confusion Matrix: [[6074  959]
 [ 736 6382]]
Accuracy Score: 0.8802204791180835
```

In [36]:
```
#RESULTS OF SUPPORT VECTOR
print('Model Prediction Accuracy:',str(np.round(accuracy_score(label_test,svc_pred)*100,2)) + '%
```

```
Model Prediction Accuracy: 88.02%
```

In [37]:
```
#CONCLUSION
print('Performance of various ML models:')
print('\n')
print('Random Forests Classifier Prediction Accuracy:',str(np.round(accuracy_score(label_test,rf
print('Support Vector Machines Prediction Accuracy:',str(np.round(accuracy_score(label_test,svc_
```

```
Performance of various ML models:


Random Forests Classifier Prediction Accuracy: 83.74%
Support Vector Machines Prediction Accuracy: 88.02%
```

In [ ]:

In [ ]: