Project

Text Classification

In [171…

```
from sklearn.datasets import fetch_20newsgroups
import string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from sklearn.model_selection import train_test_split
import numpy as np
from sklearn.naive_bayes import MultinomialNB
```

In [172…

```
#fetching the dataset
#instead of using the data from the website i used the inbuilt dataset which is same as
newsgroups=fetch_20newsgroups()
#importing stopwords and punctuations
stops=set(stopwords.words('english'))
punctuations=list(string.punctuation)
stops.update(punctuations)
```

In [173…

```
len(newsgroups.data)
```

Out[173…

```
11314
```

In [174…

```
all_documents=newsgroups.data
all_categories=newsgroups.target
#dividing into words as we have to work with words not with sentences
all_documents_modified=[word_tokenize(doc) for doc in all_documents]
```

In [175…

```
#splitting into training and testing
x_train, x_test, y_train, y_test=train_test_split(all_documents_modified, all_categorie
```

In [176…

```
all_words=[]
#this list is going to contain all the words from all our tokenized documents which we w
#and unnecessary stopswords and punctuations are removed as they dont make sense
for doc in x_train:
    for word in doc:
        # removing unncecessary words
        if (word.lower() not in stops) and len(word)!=1 and len(word)!=2 and word[0]!="
            #appending necessary words or features
            all_words.append(word)
```

In [177…

```
# this functions returns the frequency of all the words from all_words which we will use
def freq_dict(all_words):
    dic=dict()
    #it iterates through all the elements in the list and increases the frequency by one
    for word in all_words:
        if word in dic.keys():
            dic[word]+=1
        else:
            dic[word]=1
    return dic

dic=freq_dict(all_words)
```

In [178…

```
#diving the freq and words into two lists and sorting them into deccreasing order of fre
#to get the maximum frequency words.
import numpy as np
freq=np.array([i for i in dic.values()])
words=np.array([i for i in dic.keys()])

words=words[np.argsort(freq)][::-1]
```

In [179…

```
for i in range(50):
    print(words[i])
```

```
Subject
Lines
Organization
would
writes
one
article
people
like
University
know
get
think
use
time
also
MAX
could
good
may
way
even
much
make
see
two
say
Distribution
God
many
right
new
want
Nntp-Posting-Host
said
first
used
NNTP-Posting-Host
system
work
need
something
well
world
anyone
problem
going
```

```
still
really
believe
```

In [180…

```
# taking only the releavent words as features since the most common are useless so that
# upto 10,000 top words
features=words[20:10000]
```

In [181…

```
# It takes the patameters x_train or x_test and the list of all features and converts i
# in that particular document, where rows are the documents and columns are the features
def data_modifier(x_data, features):
    modified_data=np.zeros((len(x_data), len(features)))
    #creating the empty 2d array

    for i in range(len(x_data)):
        #looping over each and every row in the x_data
        current_doc=x_data[i]
        #current_doc contains the current document on which we are iterating.(As the nam
        d=dict()
        #this dictionary contains the frequency of all the elements in our current_doc.
        for word in current_doc:
            if word in d.keys():
                d[word]+=1
            else:
                d[word]=1
        #dictionary created
        for j in range(len(features)):
            #now for each feature in features we will insert the value of the dictionary
            #the frequency of each feature in that current document.
            if features[j] in d.keys():
                modified_data[i][j]=d[features[j]]
            else:
                continue
    #finally I have returned the modified array.
    return modified_data
```

In [182…

```
len(x_train)
```

Out[182…

```
8485
```

In [183…

```
#converting the training data into 2d form
x_train_modified = data_modifier(x_train, features)
```

In [184…

```
#converting the testing data into 2d form
x_test_modified= data_modifier(x_test, features)
```

Inbuilt Naive Bayes

In [186…

```
#first trying out the inbuilt Multinomial naive bayes classifier.
clf=MultinomialNB()
clf.fit(x_train_modified, y_train)
print(clf.score(x_test_modified, y_test)*100, "%")
```

```
86.46164722516791 %
```

Building our own Naive Bayes classifier from scratch

In [187…

```
#this function takes our xtrain and ytrain and combine them into a dictionary with featu
#in them and then returns a dictionary
def fit(x_train , y_train):
    d = {}
    #defining a dictionary
    for i in range(20):
        docs = x_train[y_train == i]
        #taking the classes one by one from x_train
        d[i] = {}
        #making a dictionary on the ith class to save the features and their total value
        d[i]['total'] = 0
        #this holds the value of the total words present in the class to be used in the
        for j in range(len(features)):
            d[i][features[j]] = docs[:, j].sum()
            #how many times jth feature is coming corresponding to class i
            d[i]['total']+=d[i][features[j]]
            #stores the sum of all the values of ith key
    return d
```

In [188…

```
# finding probabilty of each word in document for the current class
def probability(dictionary , x , current_class):

    prob_word = []
    #it will save all the probabs
    for i in range(len(x)):

        if x[i]!=0:
            #we dont want to consider words which are not present
            num = dictionary[current_class][features[i]]
            #finding numerator
            denom = dictionary[current_class]['total']
            #finding denominator
            prob = np.log((num + 1)/(denom + len(x)))
            #finding probability with laplace correction
            prob_word.append(prob)
            # appending in the list

    return sum(prob_word)
```

In [189…

```
#finding the best class using the above function by comparing all the probabilities
def predictSinglePoint(dictionary, x):

    classes = dictionary.keys()
    # finding all classes
    bestp = -20
    #taking best probability negative
    bestc = -20
    #taking the best class negative
    firstrun = True
    #firstrun is created to update with the first probability no matter the case so nega

    for clas in classes:
        #iterating through each class
        prob_class = probability(dictionary, x, clas)
        #finding the probab of current class using the probabilty function as given abov
        if(firstrun == True or bestp < prob_class):
            #updating the values in our variables to get the maximum probab class
            bestp = prob_class
            bestc = clas
        firstrun = False
        #making firstrun as false as we dont want to use it anymore

    return bestc
```

In [190…

```
#this function return the predicted classes by using the above functions
def predict(x_test, dictionary):
    y_pred = []
    #creating the empty list for predicted values
    for doc in x_test:
        #iterating through every doc and predicting values and appending to the predict
        y_pred.append(predictSinglePoint(dictionary ,doc))

    return y_pred
```

In [191…

```
#dictionary created through fit function contains classes and their features list
dictionary=fit(x_train_modified, y_train)
```

In [192…

```
#example of class 2 in dictionary which contains 20 classes
dictionary[2]
```

Out[192…

```
{'total': 19433.0,
 'way': 66.0,
 'even': 51.0,
 'much': 71.0,
 'make': 80.0,
 'see': 49.0,
 'two': 42.0,
 'say': 34.0,
 'Distribution': 101.0,
 'God': 0.0,
 'many': 47.0,
 'right': 53.0,
 'new': 79.0,
 'want': 69.0,
 'Nntp-Posting-Host': 105.0,
 'said': 25.0,
 'first': 48.0,
 'used': 74.0,
 'NNTP-Posting-Host': 77.0,
 'system': 87.0,
 'work': 96.0,
 'need': 88.0,
 'something': 76.0,
 'well': 79.0,
 'world': 41.0,
 'anyone': 105.0,
 'problem': 163.0,
 'going': 36.0,
 'still': 47.0,
 'really': 51.0,
 'believe': 25.0,
 'back': 36.0,
 'years': 12.0,
 'must': 22.0,
 'find': 64.0,
 'year': 15.0,
 'using': 168.0,
'point': 24.0,
'take': 28.0,
'better': 40.0,
'things': 33.0,
'Reply-To': 79.0,
'information': 56.0,
'might': 27.0,
'file': 232.0,
'program': 127.0,
'last': 24.0,
'question': 33.0,
'got': 53.0,
'government': 0.0,
'never': 36.0,
'help': 87.0,
'made': 17.0,
'available': 71.0,
'sure': 49.0,
'since': 43.0,
'number': 36.0,
'Thanks': 120.0,
'without': 43.0,
'New': 16.0,
'thing': 36.0,
'someone': 39.0,
'another': 30.0,
'read': 43.0,
'David': 29.0,
'Computer': 44.0,
'little': 24.0,
'come': 20.0,
'etc': 48.0,
'version': 125.0,
'give': 29.0,
'part': 28.0,
'John': 21.0,
'around': 36.0,
'case': 17.0,
'fact': 12.0,
'drive': 44.0,
'different': 29.0,
'anything': 19.0,
'long': 12.0,
'course': 9.0,
'1993': 20.0,
'least': 26.0,
'set': 99.0,
'says': 21.0,
'data': 26.0,
'look': 26.0,
'power': 2.0,
'best': 30.0,
'lot': 27.0,
'probably': 30.0,
'tell': 41.0,
'day': 7.0,
'possible': 38.0,
'enough': 14.0,
'seems': 42.0,
'car': 1.0,
'every': 24.0,
'put': 18.0,
'true': 24.0,
'name': 31.0,
'key': 19.0,
'run': 111.0,
'Jesus': 0.0,
'far': 22.0,
'please': 44.0,
'law': 0.0,
'list': 32.0,
'try': 56.0,
'Q,3': 821.0,
'card': 170.0,
'either': 24.0,
'line': 30.0,
'files': 212.0,
'Windows': 501.0,
'else': 32.0,
'though': 37.0,
'hard': 34.0,
'team': 0.0,
'let': 24.0,
'game': 6.0,
'called': 45.0,
'problems': 73.0,
'great': 23.0,
'Well': 22.0,
'support': 45.0,
'mean': 20.0,
'life': 6.0,
'bit': 22.0,
'example': 15.0,
'wrong': 23.0,
'rather': 27.0,
'reason': 13.0,
'found': 38.0,
'done': 12.0,
'person': 9.0,
'keep': 15.0,
'send': 23.0,
'Inc.': 39.0,
'Please': 42.0,
'old': 19.0,
'Center': 23.0,
'USA': 34.0,
'thought': 20.0,
'nothing': 15.0,
'software': 65.0,
'post': 30.0,
'end': 22.0,
'able': 31.0,
'One': 9.0,
'message': 37.0,
'Keywords': 50.0,
'real': 21.0,
'order': 20.0,
'next': 22.0,
'always': 26.0,
'looking': 41.0,
'public': 12.0,
'means': 21.0,
'bad': 17.0,
'place': 21.0,
'less': 22.0,
'seen': 47.0,
'others': 5.0,
'state': 4.0,
'group': 30.0,
'State': 27.0,
'trying': 18.0,
'actually': 21.0,
'Science': 18.0,
'Mark': 8.0,
'following': 17.0,
'Israel': 0.0,
'away': 12.0,
'quite': 26.0,
'free': 33.0,
'Research': 28.0,
'wrote': 30.0,
'high': 24.0,
'window': 55.0,
'several': 30.0,
'ever': 11.0,
'heard': 22.0,
'left': 12.0,
'second': 15.0,
'already': 16.0,
'Also': 30.0,
'play': 15.0,
'start': 25.0,
'call': 19.0,
'opinions': 27.0,
'evidence': 0.0,
'However': 21.0,
'getting': 18.0,
'idea': 15.0,
'control': 33.0,
'Jews': 0.0,
'kind': 14.0,
'man': 3.0,
'seem': 22.0,
'netcom.com': 8.0,
'makes': 13.0,
'info': 30.0,
'three': 10.0,
'money': 8.0,
'space': 38.0,
'chip': 13.0,
'current': 23.0,
'human': 1.0,
'based': 21.0,
'Steve': 21.0,
'American': 1.0,
'Christian': 0.0,
'given': 4.0,
'Apr': 17.0,
'ago': 23.0,
'games': 9.0,
'times': 13.0,
'today': 7.0,
'whether': 34.0,
'change': 52.0,
'small': 41.0,
'Michael': 21.0,
'yet': 21.0,
'came': 17.0,
'code': 22.0,
'encryption': 0.0,
'local': 14.0,
'book': 8.0,
'email': 33.0,
'source': 14.0,
'April': 9.0,
'Internet': 46.0,
'answer': 15.0,
'interested': 16.0,
'usa': 48.0,
'Institute': 16.0,
'running': 71.0,
'told': 7.0,
'ask': 8.0,
'saying': 3.0,
'standard': 21.0,
'home': 14.0,
'B8F': 543.0,
'gun': 0.0,
'large': 26.0,
'Technology': 33.0,
'whole': 9.0,
'mail': 38.0,
'Bill': 5.0,
'National': 19.0,
'questions': 24.0,
'issue': 8.0,
'children': 3.0,
'buy': 28.0,
'Paul': 1.0,
'important': 9.0,
'disk': 71.0,
'Department': 31.0,
'works': 35.0,
'matter': 4.0,
'posting': 7.0,
'speed': 22.0,
'Canada': 19.0,
'address': 24.0,
'Robert': 25.0,
'President': 3.0,
'show': 8.0,
'days': 9.0,
'live': 3.0,
'machine': 55.0,
'Article-I.D': 26.0,
'agree': 2.0,
'pretty': 17.0,
'stuff': 24.0,
'server': 16.0,
'Systems': 48.0,
'feel': 11.0,
'big': 13.0,
'word': 10.0,
'access': 61.0,
'went': 9.0,
'Mike': 26.0,
'comes': 28.0,
'memory': 69.0,
'side': 4.0,
'claim': 5.0,
'including': 12.0,
'California': 12.0,
'computer': 53.0,
'Mac': 14.0,
'general': 11.0,
'DOS': 157.0,
'package': 30.0,
'rights': 0.0,
'started': 12.0,
'working': 14.0,
'provide': 12.0,
'price': 17.0,
'Bible': 0.0,
'include': 11.0,
'programs': 54.0,
'understand': 4.0,
'simply': 8.0,
'often': 7.0,
'output': 15.0,
'X-Newsreader': 27.0,
'Jim': 2.0,
'Yes': 17.0,
'everything': 21.0,
'remember': 29.0,
'1.1': 29.0,
'care': 2.0,
'Armenian': 0.0,
'SCSI': 2.0,
'systems': 15.0,
'Turkish': 0.0,
'Space': 0.0,
'original': 18.0,
'making': 1.0,
'hope': 15.0,
'tried': 45.0,
'maybe': 19.0,
'Christians': 0.0,
'similar': 14.0,
'cost': 8.0,
'phone': 14.0,
'San': 11.0,
'Israeli': 0.0,
'couple': 20.0,
'country': 0.0,
'full': 22.0,
'hand': 6.0,
'e-mail': 26.0,
'took': 10.0,
'known': 3.0,
'image': 14.0,
'area': 6.0,
'mind': 2.0,
'argument': 1.0,
'consider': 5.0,
'difference': 18.0,
'everyone': 17.0,
'certain': 11.0,
'likely': 10.0,
'news': 18.0,
'Armenians': 0.0,
'almost': 11.0,
'later': 11.0,
'within': 15.0,
'however': 12.0,
'Christ': 0.0,
'season': 0.0,
'size': 58.0,
'York': 3.0,
'guess': 9.0,
'men': 0.0,
'write': 19.0,
'sort': 8.0,
'type': 16.0,
'per': 6.0,
'Dept': 22.0,
'A86': 413.0,
'words': 1.0,
'numbers': 10.0,
'talking': 1.0,
'cause': 5.0,
'145': 388.0,
'faith': 0.0,
'pay': 6.0,
'love': 14.0,
'religion': 0.0,
'truth': 0.0,
'color': 21.0,
'Mr.': 2.0,
'usually': 17.0,
'FAQ': 8.0,
'IBM': 20.0,
'deal': 7.0,
'College': 24.0,
'DoD': 0.0,
'single': 8.0,
'GMT': 14.0,
'Pittsburgh': 0.0,
'instead': 20.0,
'open': 6.0,
'sense': 5.0,
'wanted': 7.0,
'display': 25.0,
'experience': 15.0,
'Jewish': 0.0,
'saw': 4.0,
'particular': 14.0,
'subject': 7.0,
'keys': 1.0,
'opinion': 3.0,
'period': 3.0,
'Clinton': 3.0,
'certainly': 8.0,
'Washington': 10.0,
'death': 0.0,
'100': 11.0,
'correct': 18.0,
'Engineering': 28.0,
'win': 18.0,
'Sun': 7.0,
'via': 17.0,
'goes': 16.0,
'uses': 30.0,
'groups': 19.0,
'body': 2.0,
'video': 49.0,
'points': 1.0,
'reading': 23.0,
'entry': 12.0,
'advance': 39.0,
'U.S.': 2.0,
'taken': 3.0,
'killed': 0.0,
'bike': 1.0,
'nice': 19.0,
'driver': 146.0,
'Dave': 18.0,
'screen': 58.0,
'clear': 5.0,
'Software': 20.0,
'anybody': 41.0,
'top': 21.0,
'hardware': 28.0,
'become': 1.0,
'written': 15.0,
'windows': 193.0,
'players': 0.0,
'First': 11.0,
'position': 6.0,
'James': 12.0,
'Since': 13.0,
'exist': 6.0,
'fast': 18.0,
'private': 1.0,
'hit': 3.0,
'unless': 8.0,
'common': 6.0,
'Apple': 0.0,
'cars': 0.0,
'copy': 33.0,
'discussion': 5.0,
'women': 0.0,
'war': 3.0,
'May': 6.0,
'asked': 8.0,
'posted': 10.0,
'Toronto': 7.0,
'turn': 13.0,
'lines': 17.0,
'build': 17.0,
'fine': 32.0,
'exactly': 11.0,
'light': 6.0,
'rest': 4.0,
'especially': 9.0,
'anonymous': 3.0,
'return': 8.0,
'laws': 1.0,
'NASA': 0.0,
'police': 0.0,
'history': 0.0,
'needed': 13.0,
'check': 24.0,
'considered': 6.0,
'simple': 20.0,
'except': 16.0,
'value': 16.0,
'net': 15.0,
'Clipper': 1.0,
'happened': 5.0,
'sale': 2.0,
'statement': 9.0,
'Corporation': 17.0,
'Let': 2.0,
'past': 13.0,
'format': 19.0,
'ground': 0.0,
'Summary': 19.0,
'kill': 0.0,
'longer': 8.0,
'application': 36.0,
'Texas': 10.0,
'needs': 15.0,
'service': 8.0,
'strong': 1.0,
'Chicago': 10.0,
'United': 0.0,
'form': 5.0,
'easy': 17.0,
'drivers': 131.0,
'worth': 16.0,
'low': 3.0,
'effect': 2.0,
'ones': 7.0,
'among': 1.0,
'situation': 5.0,
'user': 28.0,
'graphics': 18.0,
'job': 13.0,
'security': 9.0,
'Western': 12.0,
'NHL': 0.0,
'fire': 2.0,
'level': 9.0,
'System': 14.0,
'talk': 9.0,
'Brian': 17.0,
'head': 6.0,
'States': 0.0,
'mine': 15.0,
'view': 6.0,
'behind': 3.0,
'although': 14.0,
'business': 4.0,
'gets': 6.0,
'monitor': 17.0,
'sell': 5.0,
'drives': 16.0,
'major': 3.0,
'Smith': 5.0,
'week': 6.0,
'1D9': 327.0,
'interesting': 3.0,
'dead': 3.0,
'due': 11.0,
'board': 17.0,
'weapons': 0.0,
'accept': 8.0,
'looks': 19.0,
'allow': 27.0,
'previous': 16.0,
'text': 28.0,
'contact': 8.0,
'company': 7.0,
'whatever': 7.0,
'hockey': 0.0,
'TIN': 17.0,
'mode': 52.0,
'Bob': 17.0,
'sound': 19.0,
'stop': 3.0,
'section': 15.0,
'Peter': 19.0,
'happen': 5.0,
'hear': 4.0,
'together': 12.0,
'runs': 10.0,
'friend': 20.0,
'front': 0.0,
'move': 18.0,
'assume': 6.0,
'night': 1.0,
'months': 5.0,
'specific': 11.0,
'short': 12.0,
'coming': 4.0,
'perhaps': 0.0,
'moral': 1.0,
'early': 4.0,
'test': 5.0,
'parts': 5.0,
'future': 4.0,
'rules': 3.0,
'mentioned': 8.0,
'File': 17.0,
'expect': 18.0,
'upon': 2.0,
'crime': 0.0,
'rate': 17.0,
'Division': 16.0,
'books': 11.0,
'cases': 10.0,
'PL+': 309.0,
'technology': 2.0,
'study': 2.0,
'directory': 48.0,
'Good': 9.0,
'military': 0.0,
'taking': 5.0,
'add': 11.0,
'offer': 9.0,
'necessary': 6.0,
'Services': 23.0,
'Even': 6.0,
'note': 4.0,
'Maybe': 8.0,
'members': 0.0,
'recently': 22.0,
'personal': 12.0,
'religious': 0.0,
'process': 7.0,
'purpose': 7.0,
'1992': 3.0,
'users': 20.0,
'force': 3.0,
'guy': 2.0,
'Note': 12.0,
'sent': 9.0,
'soon': 15.0,
'special': 10.0,
'includes': 9.0,
'result': 7.0,
'thinking': 5.0,
'present': 6.0,
'explain': 5.0,
'Andrew': 7.0,
'various': 12.0,
'Richard': 24.0,
'anyway': 11.0,
'science': 0.0,
'America': 4.0,
'close': 6.0,
'legal': 1.0,
'four': 5.0,
'images': 1.0,
'guns': 0.0,
'World': 9.0,
'along': 9.0,
'weeks': 4.0,
'Germany': 20.0,
'deleted': 25.0,
'Chris': 23.0,
'million': 9.0,
'included': 19.0,
'Many': 8.0,
'Tom': 22.0,
'water': 1.0,
'Motif': 5.0,
'doubt': 2.0,
'face': 1.0,
'leave': 9.0,
'cards': 36.0,
'involved': 0.0,
'story': 5.0,
'shall': 0.0,
'response': 4.0,
'yes': 19.0,
'Perhaps': 8.0,
'research': 2.0,
'third': 3.0,
'near': 5.0,
'55.0': 0.0,
'knows': 19.0,
'wants': 11.0,
'speak': 4.0,
'STEPHANOPOULOS': 0.0,
'MR.': 0.0,
'device': 13.0,
'Keith': 3.0,
'performance': 32.0,
'controller': 7.0,
'takes': 9.0,
'hold': 6.0,
'bus': 21.0,
'higher': 8.0,
'details': 5.0,
'model': 10.0,
'appreciated': 20.0,
'population': 0.0,
'Thomas': 17.0,
'ideas': 13.0,
'knowledge': 3.0,
'completely': 9.0,
'Scott': 15.0,
'Computing': 21.0,
'ways': 4.0,
'Eric': 10.0,
'cover': 2.0,
'Access': 43.0,
'***': 26.0,
'lost': 2.0,
'exists': 1.0,
'teams': 0.0,
'action': 1.0,
'site': 31.0,
'market': 8.0,
'hell': 1.0,
'cut': 12.0,
'designed': 7.0,
'series': 4.0,
'quality': 15.0,
'results': 27.0,
'Group': 15.0,
'willing': 6.0,
'algorithm': 0.0,
'food': 2.0,
'built': 6.0,
'outside': 2.0,
'belief': 0.0,
'box': 15.0,
'required': 7.0,
'newsgroup': 10.0,
'sometimes': 12.0,
'die': 2.0,
'building': 2.0,
'living': 0.0,
'mouse': 108.0,
'chance': 3.0,
'Russian': 0.0,
'existence': 0.0,
'create': 29.0,
'Turkey': 0.0,
'political': 0.0,
'wish': 3.0,
'average': 1.0,
'St.': 5.0,
'protect': 0.0,
'useful': 5.0,
'carry': 2.0,
'created': 9.0,
'figure': 6.0,
'Could': 11.0,
'appears': 9.0,
'George': 9.0,
'uunet': 17.0,
'player': 2.0,
'save': 13.0,
'engine': 13.0,
'objective': 0.0,
'act': 3.0,
'devices': 1.0,
'error': 30.0,
'Frank': 2.0,
'currently': 9.0,
'serious': 9.0,
'Univ': 8.0,
'young': 0.0,
'BBS': 28.0,
'states': 3.0,
'looked': 14.0,
'interest': 4.0,
'FBI': 0.0,
'individual': 2.0,
'sources': 3.0,
'reference': 12.0,
   'Public': 13.0,
   'bought': 20.0,
   'church': 0.0,
   'network': 24.0,
   'RAM': 27.0,
   'interface': 12.0,
   'Gordon': 0.0,
   'peace': 0.0,
   'record': 4.0,
   'suggest': 9.0,
   'family': 5.0,
   'Second': 1.0,
   'health': 0.0,
   'IDE': 5.0,
   'land': 0.0,
   'normal': 9.0,
   'Anyone': 22.0,
   'report': 9.0,
   'half': 4.0,
   'widget': 0.0,
   'Christianity': 0.0,
   'product': 32.0,
   'continue': 1.0,
   'claims': 9.0,
   'black': 6.0,
   'Boston': 4.0,
   '0T-': 246.0,
   'Los': 2.0,
   'clearly': 10.0,
   'main': 5.0,
   'manager': 23.0,
   'serial': 25.0,
   'FTP': 15.0,
   'Law': 0.0,
   'machines': 21.0,
   'gas': 1.0,
   'Information': 11.0,
   'atheists': 0.0,
   'choice': 3.0,
   'difficult': 0.0,
   'require': 5.0,
   'amount': 21.0,
   'Would': 2.0,
   'attack': 0.0,
   'reply': 9.0,
   'nature': 2.0,
   'condition': 1.0,
   'privacy': 0.0,
   'North': 5.0,
   'ftp': 38.0,
   'reasons': 2.0,
   'arms': 0.0,
   'shot': 0.0,
   'secret': 0.0,
   'necessarily': 17.0,
   'mention': 3.0,
   'press': 4.0,
   'white': 7.0,
   'Laboratory': 14.0,
   '3.1': 159.0,
   'goal': 1.0,
   'Earth': 1.0,
   'basic': 6.0,
   'chips': 9.0,
'follow': 5.0,
'blood': 0.0,
'learn': 5.0,
'modem': 35.0,
'applications': 38.0,
'radio': 5.0,
'Illinois': 13.0,
'City': 5.0,
'bits': 1.0,
'Sorry': 14.0,
'worse': 3.0,
'launch': 1.0,
'port': 23.0,
'Jon': 5.0,
'supposed': 17.0,
'society': 0.0,
'insurance': 5.0,
'issues': 2.0,
'city': 1.0,
'enforcement': 0.0,
'easily': 10.0,
'decided': 4.0,
'White': 8.0,
'Doug': 4.0,
'installed': 32.0,
'General': 8.0,
'allowed': 1.0,
'basis': 3.0,
'keyboard': 2.0,
'Georgia': 14.0,
'request': 1.0,
'avoid': 7.0,
'otherwise': 4.0,
'court': 0.0,
'entire': 8.0,
'baseball': 3.0,
'input': 2.0,
'cheap': 5.0,
'child': 6.0,
'andrew.cmu.edu': 0.0,
'School': 16.0,
'terms': 1.0,
'commercial': 5.0,
'complete': 11.0,
'directly': 8.0,
'Actually': 3.0,
'theory': 4.0,
'places': 2.0,
'giving': 7.0,
'Greek': 0.0,
'citizens': 0.0,
'inside': 4.0,
'oil': 0.0,
'Graphics': 14.0,
'printer': 81.0,
'JPEG': 0.0,
'comments': 5.0,
'media': 3.0,
'function': 13.0,
'design': 8.0,
'MSG': 0.0,
'South': 11.0,
'morality': 0.0,
'Turks': 0.0,
'therefore': 3.0,
'plan': 2.0,
'generally': 4.0,
'gives': 6.0,
'Email': 9.0,
'Gary': 3.0,
'house': 1.0,
'thus': 1.0,
'gave': 4.0,
'stupid': 4.0,
'related': 3.0,
'appropriate': 5.0,
'tax': 4.0,
'House': 0.0,
'secure': 0.0,
'Colorado': 7.0,
'solution': 15.0,
'command': 28.0,
'community': 2.0,
'thanks': 16.0,
'obvious': 5.0,
'GIZ': 220.0,
'Angeles': 2.0,
'title': 6.0,
'school': 4.0,
'project': 1.0,
'stated': 2.0,
'vs.': 6.0,
'Fax': 23.0,
'Unix': 10.0,
'apply': 4.0,
'final': 1.0,
'Dan': 4.0,
'knew': 6.0,
'asking': 3.0,
'limited': 7.0,
'tape': 4.0,
'reasonable': 0.0,
'Phone': 15.0,
'addition': 7.0,
'class': 15.0,
'recent': 5.0,
'approach': 3.0,
'advice': 4.0,
'faster': 18.0,
'league': 0.0,
'Originator': 15.0,
'break': 3.0,
'values': 0.0,
'field': 3.0,
'___': 5.0,
'unit': 40.0,
'changed': 12.0,
'posts': 1.0,
'pick': 5.0,
'FAX': 22.0,
'Armenia': 0.0,
'alone': 4.0,
'safety': 0.0,
'actions': 0.0,
'died': 0.0,
'Tim': 4.0,
'appreciate': 16.0,
'sites': 11.0,
'Koresh': 0.0,
'meaning': 0.0,
'paper': 7.0,
'Vancouver': 7.0,
'Thank': 13.0,
'received': 9.0,
'Communications': 16.0,
'Americans': 0.0,
'Box': 9.0,
'wait': 9.0,
'language': 9.0,
'People': 0.0,
'defense': 0.0,
'Office': 14.0,
'brought': 4.0,
'earth': 0.0,
'lower': 6.0,
'method': 1.0,
'bring': 1.0,
'1st': 2.0,
'lots': 5.0,
'former': 2.0,
'define': 4.0,
'month': 10.0,
'possibly': 3.0,
'trouble': 12.0,
'Congress': 0.0,
'contains': 12.0,
'happens': 8.0,
'wondering': 15.0,
'folks': 6.0,
'god': 2.0,
'Dr.': 5.0,
'Data': 11.0,
'event': 0.0,
'Roger': 0.0,
'base': 5.0,
'7EY': 206.0,
'Joe': 1.0,
'attempt': 1.0,
'worked': 8.0,
'drugs': 0.0,
'across': 6.0,
'Banks': 0.0,
'jobs': 2.0,
'fall': 2.0,
'views': 8.0,
'somebody': 13.0,
'supply': 0.0,
'shows': 5.0,
'happy': 10.0,
'Red': 5.0,
'cable': 5.0,
'realize': 4.0,
'comment': 2.0,
'companies': 5.0,
'hours': 2.0,
'evil': 2.0,
'voice': 10.0,
'sin': 0.0,
'published': 2.0,
'author': 11.0,
'i.e': 13.0,
'extra': 3.0,
 'provided': 4.0,
 'supports': 7.0,
 'office': 7.0,
 ...}
```

In [193…

```
#predicted values from the predict function
y_predicted=predict(x_test_modified, dictionary)
```

In [194…

```
#comparing our predcited values with the y_test
from sklearn.metrics import classification_report
print(classification_report(y_true=y_test, y_pred=y_predicted))
```

```
              precision    recall  f1-score   support

           0       0.87      0.89      0.88       114
           1       0.64      0.82      0.72       152
           2       0.95      0.67      0.78       139
           3       0.61      0.82      0.70       152
           4       0.76      0.88      0.82       138
           5       0.85      0.83      0.84       153
           6       0.83      0.69      0.75       147
           7       0.81      0.91      0.85       137
           8       0.95      0.90      0.93       131
           9       0.91      0.94      0.92       135
          10       0.98      0.95      0.97       136
          11       0.93      0.97      0.95       145
          12       0.91      0.70      0.79       157
          13       0.98      0.94      0.96       151
          14       0.96      0.88      0.92       155
          15       0.84      0.94      0.88       159
          16       0.88      0.91      0.90       140
          17       0.96      0.91      0.94       149
          18       0.87      0.85      0.86       138
          19       0.80      0.60      0.69       101

    accuracy                           0.85      2829
   macro avg       0.87      0.85      0.85      2829
weighted avg       0.87      0.85      0.85      2829
```

The result of our Mutlinomial Naive Bayes is almost same as that of our inbuilt naive bayes

In [195…

```
np.savetxt('predicted_data.csv',y_predicted, delimiter=',')
```

In [ ]: