Naive Bayes on Amazon Fine Food Review

Problem defintion

Problem description:

Our objective is to apply the Naive Bayes algorithm on the reviews and to create a model that can predict a review is a positive or negative review on unseen data.

About Input Data:

1. Amazon Fine food reviews dataset
2. Removing neutral reviews that is the Score field = 3
3. Score of 1 and 2 is considered as negative while positive reviews are of score 4 and 5
4. 0 represents negative review and 1 represents positive review

Overview:

1. We will have a train:cv:test split up as 60:20:20
2. We will apply a simple CV on CV data against the train data
3. We will apply different text processing techniques like BoW and TF-IDF. W2V is not used since NB does not work with negative values

Assumptions:

1. We will weight both the classes are important.
2. The distribution of test and the train data are not very different
3. The model with the lowest False Positive Rate is chosen as the best model since the positive class is dominating.

Running instance:

8 Core - Processor with 52 GB RAM on Google Cloud.

Dataset Pre-Processing

Downloading Dataset

In [3]:

```
#from google.colab import files
#files.upload()
!pip install -q kaggle
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!kaggle datasets download -d snap/amazon-fine-food-reviews
!unzip amazon-fine-food-reviews.zip
```

```
Warning: Your Kaggle API key is readable by otherusers on this system! To fix this, you can run'ch
mod 600 /content/.kaggle/kaggle.json'
Downloading amazon-fine-food-reviews.zip to /content/datalab
100%|████████████████████████████████████████| 251M/251M [00:04<00:00, 51.7MB/s]

Archive:  amazon-fine-food-reviews.zip
  inflating: Reviews.csv
  inflating: database.sqlite
  inflating: hashes.txt
```

Importing Dataset

In [4]:

```
import pandas as pd
input_data = pd.read_csv('Reviews.csv')
```

```
input_data = pd.read_csv('Reviews.csv')
print("Shape of data is ", input_data.shape)
```

```
Shape of data is  (568454, 10)
```

Cleaning Dataset

In [5]:

```
# Removing all the neutral reviews
input_data = input_data[input_data.Score != 3]
sorted_data=input_data.sort_values('ProductId', axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last')
input_data = sorted_data.drop_duplicates(subset={"UserId","ProfileName","Time","Text"}, keep='first', inplace=False)
input_data = input_data[input_data.HelpfulnessNumerator<=input_data.HelpfulnessDenominator]
```

Pre-Processing

In [6]:

```
sorted_data = input_data.iloc[input_data.Time.argsort()]
sorted_data.to_csv('am.csv',sep = '\t')
ii = pd.read_csv('am.csv',sep = '\t')
```

In [7]:

```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
import string

sno = SnowballStemmer('english')
stop = set(stopwords.words('english'))

def cleanhtml(sentence):
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, ' ', sentence)
    return cleantext

def cleanpunc(sentence):
    cleaned = re.sub(r'[?|!|\'|"|#]',r'',sentence)
    cleaned = re.sub(r'[.|,|)|(|\|/]',r' ',cleaned)
    return  cleaned
```

```
[nltk_data] Downloading package stopwords to /content/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

In [8]:

```
#Stemming
import re
i=0
str1=' '
final_string=[]
s=''
for sent in ii['Text'].values:
    filtered_sentence=[]
    sent=cleanhtml(sent)
    for w in sent.split():
        for cleaned_words in cleanpunc(w).split():
            if((cleaned_words.isalpha()) & (len(cleaned_words)>2)):
                if(cleaned_words.lower() not in stop):
                    s=(sno.stem(cleaned_words.lower())).encode('utf8')
                    filtered_sentence.append(s)
                else:
                    continue
            else:
                continue
```

```
                continue
    str1 = b" ".join(filtered_sentence)
    final_string.append(str1)
    i+=1


ii['Cleaned_stemmed']=final_string
ii['Cleaned_stemmed']=ii['Cleaned_stemmed'].str.decode("utf-8")
```

In [9]:

```
#Cleaning words
i=0
str1=' '
final_string=[]
s=''
for sent in ii['Text'].values:
    filtered_sentence=[]
    sent=cleanhtml(sent)
    for w in sent.split():
        for cleaned_words in cleanpunc(w).split():
            if((cleaned_words.isalpha()) & (len(cleaned_words)>2)):
                if(cleaned_words.lower() not in stop):
                    filtered_sentence.append(cleaned_words.lower())
                else:
                    continue
            else:
                continue
    str1 = ' '.join(filtered_sentence)
    final_string.append(str1)
    i+=1


ii['Cleaned']=final_string
```

In [10]:

```
#0 for negative reviews and 1 for positive reviews
ii.Score = ii.Score.map(lambda x : 1 if (x > 3) else 0)

pick = ii[['Cleaned','Score']] # data
pick = pick[pick.Cleaned.notnull()]
x_vect = pick.Cleaned

y = pick.Score # label

from sklearn.model_selection import train_test_split
#Splitting into train and test
X_train, X_t, Y_train, Y_t = train_test_split(x_vect, y, test_size=0.40, random_state=42,shuffle=Fa
lse)
#Splitting test into CV and data
X_CV,X_test,Y_CV,Y_test =  train_test_split(X_t, Y_t, test_size=0.50, random_state=42,shuffle=False
)
```

Train Class Distribution

In [11]:

```
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
def plot_bar_val(label,ct,title):
    index = np.arange(len(label))
    plt.figure(figsize=(10,5))
    plt.bar(index, ct)
    plt.xlabel('Review Type', fontsize=15)
    plt.ylabel('Total Reviews', fontsize=15)
    plt.xticks(index, label, fontsize=15)
    plt.title(title)
    plt.show()

vc = Y_train.value_counts().to_frame()
```

```
label = ['Positive','Negative']
ct = vc.Score.values
print(vc)
plot_bar_val(label,ct,'Train Classes')
```

```
    Score
1  186939
0   31563
```

Out[11]:

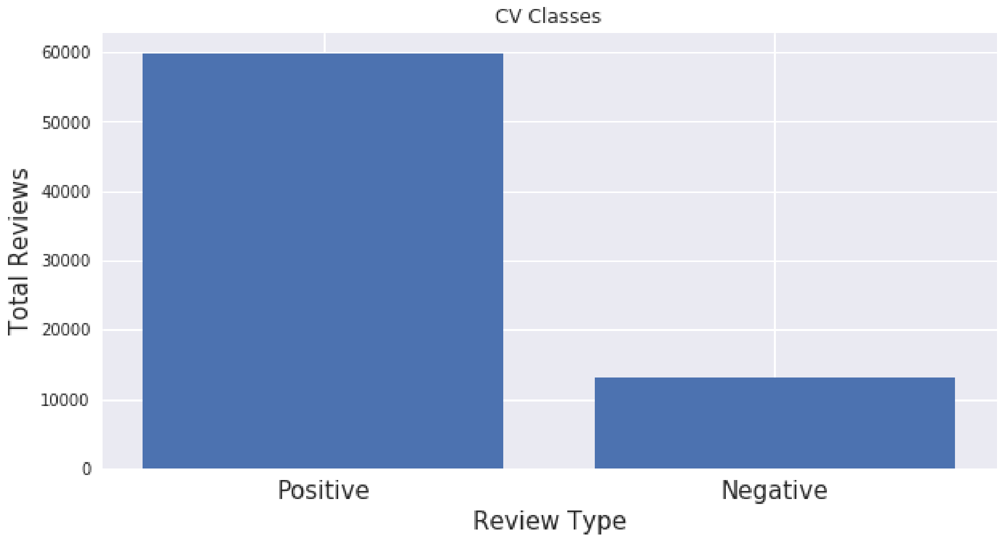
CV Class Distribution

In [12]:

```
cvv = Y_CV.value_counts().to_frame()
cv = cvv.Score.values
print(cvv)
plot_bar_val(label,cv,'CV Classes')
```

```
   Score
1  59818
0  13016
```

Out[12]:



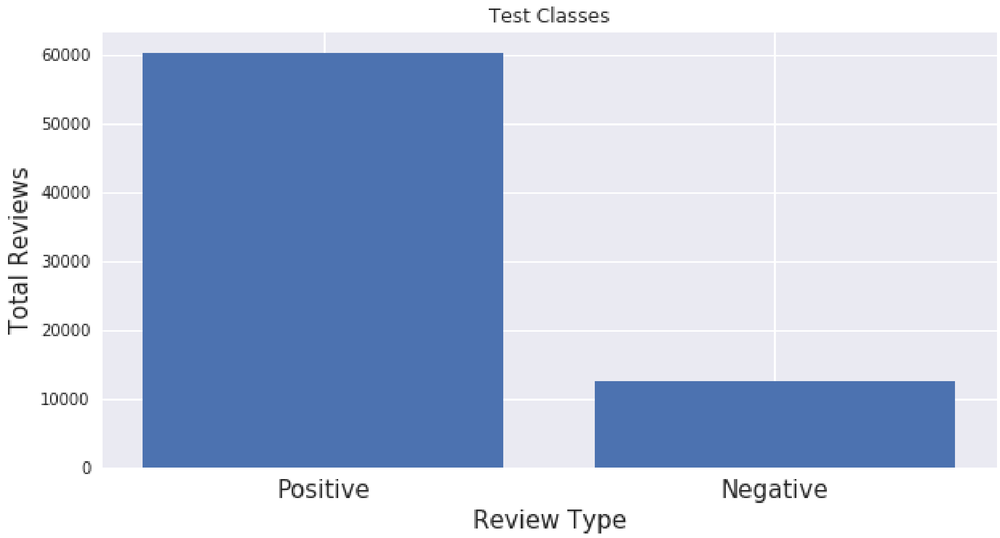
Test Class Distribution

In [13]:

```
tstt = Y_test.value_counts().to_frame()
tst = tstt.Score.values
print(tstt)
plot_bar_val(label,tst,'Test Classes')
```

```
   Score
1  60304
0  12531
```

Out[13]:



Importing Essential Packages

In [14]:

```
#%matplotlib inline
import pandas as pd
from sklearn.naive_bayes import MultinomialNB # sklearn.naive_bayes.MultinomialNB(alpha=1.0,
fit_prior=True, class_prior=None)
from sklearn.metrics import confusion_matrix,classification_report
import seaborn as sns
```

Method Declrations

In [64]:

```
#Start of method for implementing NB
def important_features(vectorizer_dict,classifier,n=20):
  class_labels = classifier.classes_
  topn_class1 = sorted(zip(classifier.feature_count_[0], vectorizer_dict.get_feature_names()),rever
se=True)[:n]
  topn_class2 = sorted(zip(classifier.feature_count_[1], vectorizer_dict.get_feature_names()),rever
se=True)[:n]
  print('\tNegative\t\t\t\tPositive')
  print("-------------------------------------------------------------------")
  print('\tCoeff\t\tWord\t\t\tCoeff\t\tWord')
  print("-------------------------------------------------------------------")
  for (coef_1, fn_1), (coef_2, fn_2) in zip(topn_class1,topn_class2):
        print ("\t%.4f\t%-15s\t\t\t%.4f\t%-15s" % (coef_1, fn_1, coef_2, fn_2))
  print("-------------------------------------------------------------------")

def NB(alpha,train_data,train_label,CV_data,CV_label,vectorizer_dict,typ):
  nb = MultinomialNB(alpha).fit(train_data,train_label)
  #using feature log prabability
  if typ != 'Test':
    train_pred = nb.predict(train_data)
    important_features(vectorizer_dict,nb,20)
    confusion_matrix_display(confusion_matrix(train_label,train_pred),alpha,train_label,'Train',FP_
```

```
train_dict)
  preds = nb.predict(CV_data)
  print(classification_report(CV_label,preds))
  confusion_matrix_display(confusion_matrix(CV_label,preds),alpha,CV_label,typ,FP_test_dict)

FP_test_dict = dict()
FP_train_dict = dict()

# Confusion Matrix
def confusion_matrix_display(conf_mtrx,alpha,tst_labels,Title,diction):
  class_names = [0,1]
  alpha = float_to_str(alpha)
  df_cm = pd.DataFrame(conf_mtrx, index=class_names, columns=class_names)
  ts = tst_labels.value_counts().to_frame()
  positive_count = int(ts.iloc[0])
  negative_count = int(ts.iloc[1])
  diction[alpha] = (int(df_cm.iloc[0,1])/positive_count)
  if Title != 'Train':
    print('\nThe TPR is : ', (int(df_cm.iloc[1,1])/positive_count))
    print('\nThe FPR is : ', (int(df_cm.iloc[0,1])/positive_count))
    print('\nThe TNR is : ', (int(df_cm.iloc[0,0])/negative_count))
    print('\nThe FNR is : ', (int(df_cm.iloc[1,0])/negative_count))
    heatmap = sns.heatmap(df_cm, annot=True, fmt="d")
    heatmap.yaxis.set_ticklabels(heatmap.yaxis.get_ticklabels(), rotation=0, ha='right')
    heatmap.xaxis.set_ticklabels(heatmap.xaxis.get_ticklabels(), rotation=45, ha='right')
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.title(Title + ' Confusion Matrix')
    plt.show()


import decimal
ctx = decimal.Context()
ctx.prec = 20
def float_to_str(f):
    d1 = ctx.create_decimal(repr(f))
    return format(d1, 'f')
```

NB With Bag Of Words

In [16]:

```
from sklearn.feature_extraction.text import CountVectorizer
BoW_dict_bigram = CountVectorizer(ngram_range = (1,2)).fit(X_train) #bi-gram
BoW_train = BoW_dict_bigram.transform(X_train)
BoW_CV = BoW_dict_bigram.transform(X_CV)
BoW_test = BoW_dict_bigram.transform(X_test)
```

Alpha = 0.00001

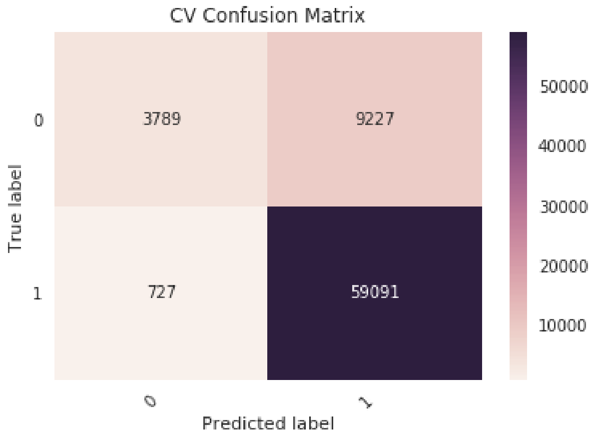
In [65]:

```
NB(0.00001,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative    Positive
-----------------------------------------------------------------
 Coeff  Word   Coeff  Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
```

```
 5640.0000 amazon              30654.0000 amazon
 5493.0000 buy                 30277.0000 use
 5400.0000 much                30227.0000 food
 5201.0000 really              30124.0000 best
 4512.0000 tried               29085.0000 also
 4499.0000 box                 28472.0000 much
 4200.0000 time                28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.84      0.29      0.43     13016
          1       0.86      0.99      0.92     59818

avg / total       0.86      0.86      0.83     72834


The TPR is :  0.9878464676184426

The FPR is :  0.15425122872713898

The TNR is :  0.29110325752919486

The FNR is :  0.055854333128457286
```

Out[65]:



Alpha = 0.0001

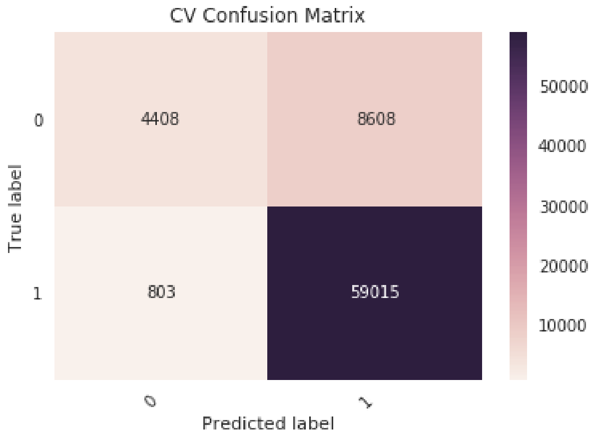
In [45]:

```
NB(0.0001,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff   Word
-----------------------------------------------------------------
 16505.0000 like               73883.0000 like
 12465.0000 taste              66173.0000 good
 12421.0000 product            62479.0000 great
 10485.0000 one                53075.0000 one
 9730.0000 would               50046.0000 taste
 8273.0000 good                48679.0000 tea
 7885.0000 flavor              44538.0000 flavor
 7721.0000 coffee              43869.0000 love
 6475.0000 dont                43634.0000 product
 5994.0000 get                 43537.0000 coffee
 5993.0000 even                33713.0000 get
 5944.0000 tea                 32158.0000 would
 5693.0000 food                31347.0000 really
 5640.0000 amazon              30654.0000 amazon
 5493.0000 buy                 30277.0000 use
 5400.0000 much                30227.0000 food
 5201.0000 really              30124.0000 best
 4512.0000 tried               29085.0000 also
 4499.0000 box                 28472.0000 much
 4200.0000 time                28071.0000 find
-----------------------------------------------------------------
```

```
              precision    recall  f1-score   support

           0       0.85      0.34      0.48     13016
           1       0.87      0.99      0.93     59818

avg / total       0.87      0.87      0.85     72834


The TPR is :  0.9865759470393527

The FPR is :  0.14390317295797253

The TNR is :  0.338660110633067

The FNR is :  0.06169330055316533
```

Out[45]:



Alpha = 0.001

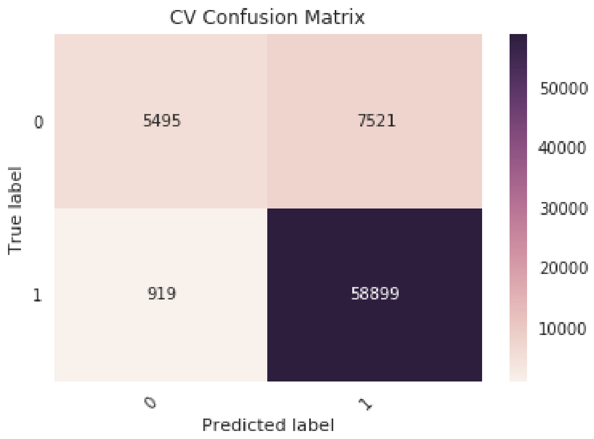
In [46]:

```
NB(0.001,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word   Coeff   Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.86      0.42      0.57     13016
           1       0.89      0.98      0.93     59818

avg / total       0.88      0.88      0.87     72834
```

```
The TPR is :  0.9846367314186365

The FPR is :  0.12573138520177873

The TNR is :  0.4221727105101414

The FNR is :  0.07060540872771973
```

Out[46]:



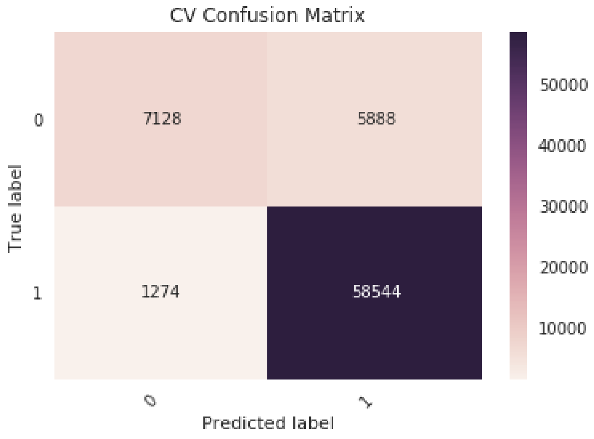
Aplha = 0.01

In [48]:

```
NB(0.01,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative    Positive
-----------------------------------------------------------------
 Coeff  Word   Coeff  Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.85      0.55      0.67     13016
          1       0.91      0.98      0.94     59818

avg / total       0.90      0.90      0.89     72834


The TPR is :  0.9787020629242034

The FPR is :  0.0984319101273864

The TNR is :  0.5476336816226183

The FNR is :  0.09787953288260602
```

Out[48]:



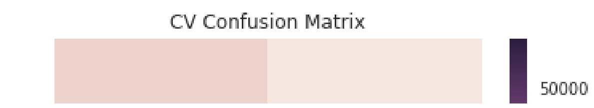
Alpha = 0.1

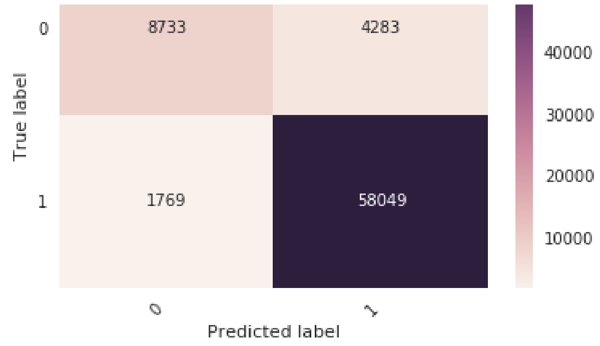
In [49]:

```
NB(0.1,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff   Word
-----------------------------------------------------------------
 16505.0000 like             73883.0000 like
 12465.0000 taste            66173.0000 good
 12421.0000 product          62479.0000 great
 10485.0000 one              53075.0000 one
 9730.0000 would            50046.0000 taste
 8273.0000 good             48679.0000 tea
 7885.0000 flavor           44538.0000 flavor
 7721.0000 coffee           43869.0000 love
 6475.0000 dont             43634.0000 product
 5994.0000 get              43537.0000 coffee
 5993.0000 even             33713.0000 get
 5944.0000 tea              32158.0000 would
 5693.0000 food             31347.0000 really
 5640.0000 amazon           30654.0000 amazon
 5493.0000 buy              30277.0000 use
 5400.0000 much             30227.0000 food
 5201.0000 really           30124.0000 best
 4512.0000 tried            29085.0000 also
 4499.0000 box              28472.0000 much
 4200.0000 time             28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.83      0.67      0.74     13016
          1       0.93      0.97      0.95     59818

avg / total       0.91      0.92      0.91     72834


The TPR is :  0.9704269617840784

The FPR is :  0.07160052158213247

The TNR is :  0.6709434542102028

The FNR is :  0.1359096496619545
```

Out[49]:





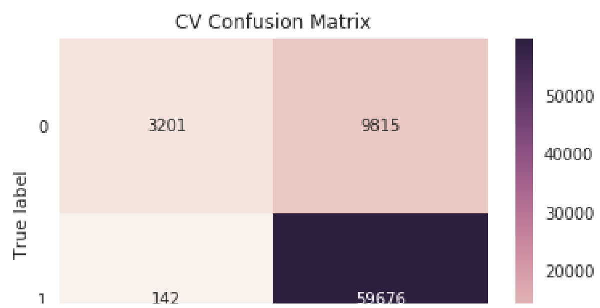
Alpha = 1

In [50]:

```
NB(1,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff   Word
-----------------------------------------------------------------
 16505.0000 like           73883.0000 like
 12465.0000 taste          66173.0000 good
 12421.0000 product        62479.0000 great
 10485.0000 one            53075.0000 one
 9730.0000 would          50046.0000 taste
 8273.0000 good           48679.0000 tea
 7885.0000 flavor         44538.0000 flavor
 7721.0000 coffee         43869.0000 love
 6475.0000 dont           43634.0000 product
 5994.0000 get            43537.0000 coffee
 5993.0000 even           33713.0000 get
 5944.0000 tea            32158.0000 would
 5693.0000 food           31347.0000 really
 5640.0000 amazon         30654.0000 amazon
 5493.0000 buy            30277.0000 use
 5400.0000 much           30227.0000 food
 5201.0000 really         30124.0000 best
 4512.0000 tried          29085.0000 also
 4499.0000 box            28472.0000 much
 4200.0000 time           28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.96      0.25      0.39     13016
          1       0.86      1.00      0.92     59818

avg / total       0.88      0.86      0.83     72834


The TPR is :  0.9976261326022268

The FPR is :  0.1640810458390451

The TNR is :  0.2459280885064536

The FNR is :  0.010909649661954518
```

Out[50]:



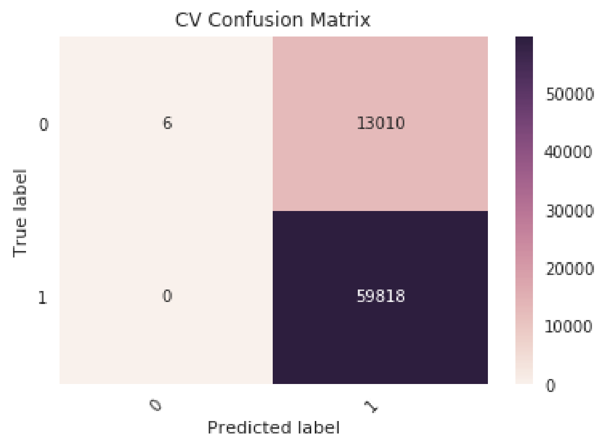
Alpha = 10

In [51]:

```
NB(10,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
------------------------------------------------------------------
 Coeff  Word   Coeff  Word
------------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
------------------------------------------------------------------
             precision    recall  f1-score   support

          0       1.00      0.00      0.00     13016
          1       0.82      1.00      0.90     59818

avg / total       0.85      0.82      0.74     72834


The TPR is :  1.0

The FPR is :  0.21749306228894313

The TNR is :  0.00046097111247695143

The FNR is :  0.0
```

Out[51]:



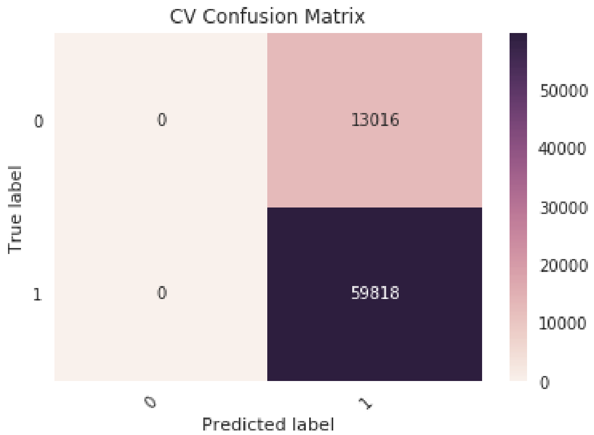
Alpha = 100

In [52]:

```
NB(100,BoW_train,Y_train,BoW_CV,Y_CV,BoW_dict_bigram,'CV')
```

```
 Negative     Positive
------------------------------------------------------------------
 Coeff  Word    Coeff  Word
------------------------------------------------------------------
 16505.0000 like              73883.0000 like
 12465.0000 taste             66173.0000 good
 12421.0000 product           62479.0000 great
 10485.0000 one               53075.0000 one
 9730.0000 would             50046.0000 taste
 8273.0000 good              48679.0000 tea
 7885.0000 flavor            44538.0000 flavor
 7721.0000 coffee            43869.0000 love
 6475.0000 dont              43634.0000 product
 5994.0000 get               43537.0000 coffee
 5993.0000 even              33713.0000 get
 5944.0000 tea               32158.0000 would
 5693.0000 food              31347.0000 really
 5640.0000 amazon            30654.0000 amazon
 5493.0000 buy               30277.0000 use
 5400.0000 much              30227.0000 food
 5201.0000 really            30124.0000 best
 4512.0000 tried             29085.0000 also
 4499.0000 box               28472.0000 much
 4200.0000 time              28071.0000 find
------------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.00      0.00      0.00     13016
          1       0.82      1.00      0.90     59818

avg / total       0.67      0.82      0.74     72834


The TPR is :  1.0

The FPR is :  0.21759336654518707

The TNR is :  0.0

The FNR is :  0.0
```

Out[52]:



Optimal Alpha

In [0]:

```
def best(FP,FP1,title):
  x = list(FP.keys())
```

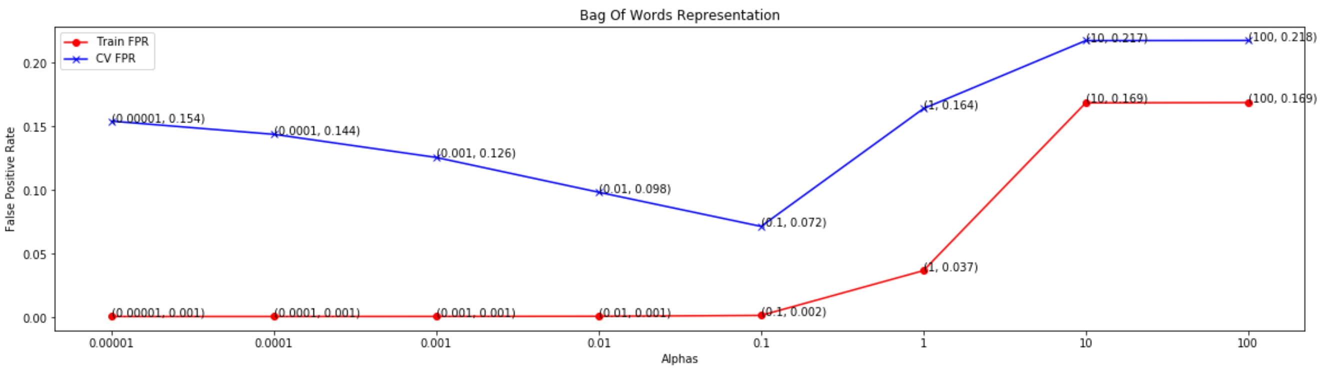
```
  y = list(FP.values())
  x1 = list(FP1.keys())
  y1 = list(FP1.values())
  best_alpha_value(FP1)
  plt.figure(figsize=(20, 5))
  plt.plot(x,y, 'or-',label='Train FPR')
  plt.plot(x1, y1, 'xb-',label='CV FPR')
  plt.legend()
  plt.xlabel('Alphas')
  plt.ylabel('False Positive Rate')
  plt.title('Bag Of Words Representation')
  for xy in zip(x, np.round(y,3)):
    plt.annotate('(%s, %s)' % xy, xy=xy, textcoords='data')
  for xy in zip(x1, np.round(y1,3)):
    plt.annotate('(%s, %s)' % xy, xy=xy, textcoords='data')
  plt.show()
def best_alpha_value(error):
  lowest = min(error.values())
  keys = [k for k, v in error.items() if v == lowest]
  print('The optimal Alpha value is : ',sorted(keys)[len(keys) - 1], 'with FPR \
        : ',lowest)
```

In [1]:

```
best(FP_train_dict,FP_test_dict,'Bag Of Words')
```

```
The optimal Alpha value is :  0.1 with FPR          :  0.07160052158213247
```

Out[1]:

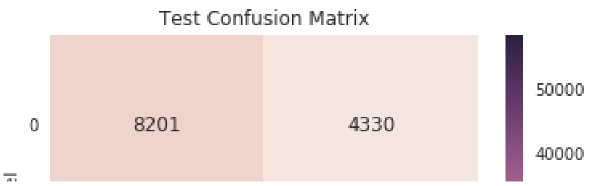


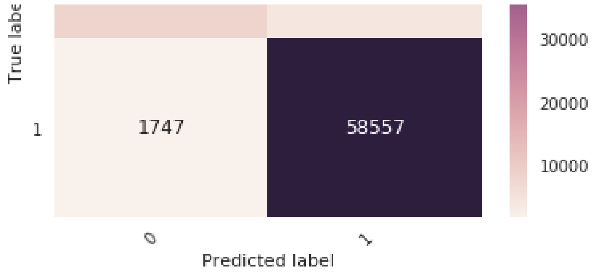
In [0]:

```
NB(0.1,BoW_train,Y_train,BoW_test,Y_test,BoW_dict_bigram,'Test')
```

```
             precision    recall  f1-score   support

          0       0.82      0.65      0.73     12531
          1       0.93      0.97      0.95     60304

avg / total       0.91      0.92      0.91     72835


The TPR is :  0.9710301140886177

The FPR is :  0.07180286548156009

The TNR is :  0.6544569467720054

The FNR is :  0.13941425265341953
```

Out[0]:





NB with TFIDF

In [53]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
TFIDF_dict_bigram = CountVectorizer(ngram_range = (1,2)).fit(X_train) #bi-gram
TFIDF_train = TFIDF_dict_bigram.transform(X_train)
TFIDF_CV = TFIDF_dict_bigram.transform(X_CV)
TFIDF_test = TFIDF_dict_bigram.transform(X_test)
FP_test_dict = dict()
FP_train_dict = dict()
```

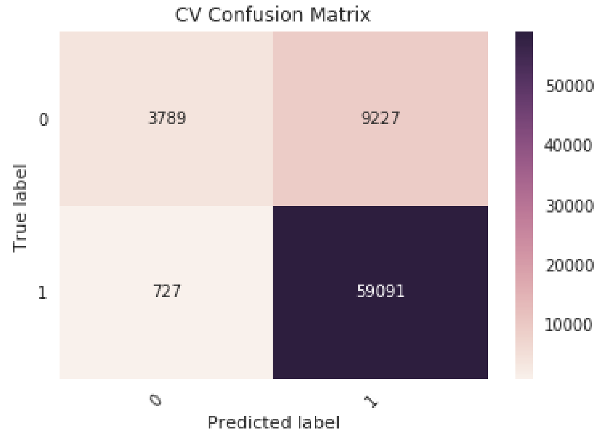
Alpha = 0.00001

In [54]:

```
NB(0.00001,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative    Positive
-----------------------------------------------------------------
 Coeff   Word   Coeff   Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.84      0.29      0.43     13016
          1       0.86      0.99      0.92     59818

avg / total       0.86      0.86      0.83     72834


The TPR is :  0.9878464676184426

The FPR is :  0.15425122872713898

The TNR is :  0.29110325752919486

The FNR is :  0.055854333128457286
```

Out[54]:



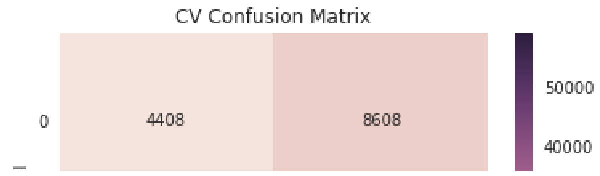
Alpha = 0.0001

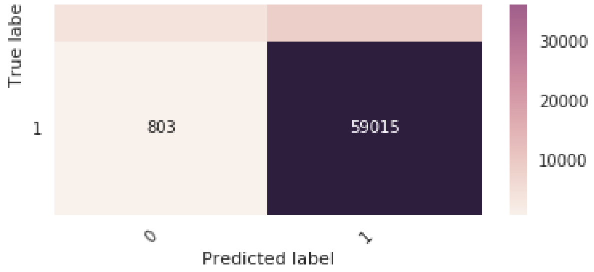
In [55]:

```
NB(0.0001,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff  Word
-----------------------------------------------------------------
 16505.0000 like           73883.0000 like
 12465.0000 taste          66173.0000 good
 12421.0000 product        62479.0000 great
 10485.0000 one            53075.0000 one
 9730.0000 would          50046.0000 taste
 8273.0000 good           48679.0000 tea
 7885.0000 flavor         44538.0000 flavor
 7721.0000 coffee         43869.0000 love
 6475.0000 dont           43634.0000 product
 5994.0000 get            43537.0000 coffee
 5993.0000 even           33713.0000 get
 5944.0000 tea            32158.0000 would
 5693.0000 food           31347.0000 really
 5640.0000 amazon         30654.0000 amazon
 5493.0000 buy            30277.0000 use
 5400.0000 much           30227.0000 food
 5201.0000 really         30124.0000 best
 4512.0000 tried          29085.0000 also
 4499.0000 box            28472.0000 much
 4200.0000 time           28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.85      0.34      0.48     13016
          1       0.87      0.99      0.93     59818

avg / total       0.87      0.87      0.85     72834


The TPR is :  0.9865759470393527

The FPR is :  0.14390317295797253

The TNR is :  0.338660110633067

The FNR is :  0.06169330055316533
```

Out[55]:





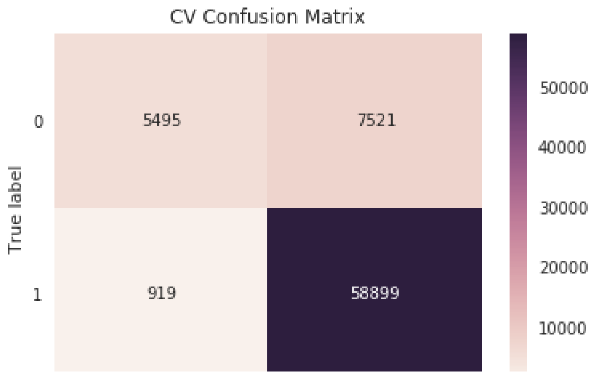
Alpha = 0.001

In [56]:

```
NB(0.001,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative    Positive
-----------------------------------------------------------------
 Coeff  Word   Coeff  Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.86      0.42      0.57     13016
          1       0.89      0.98      0.93     59818

avg / total       0.88      0.88      0.87     72834


The TPR is :  0.9846367314186365

The FPR is :  0.12573138520177873

The TNR is :  0.4221727105101414

The FNR is :  0.07060540872771973
```

Out[56]:



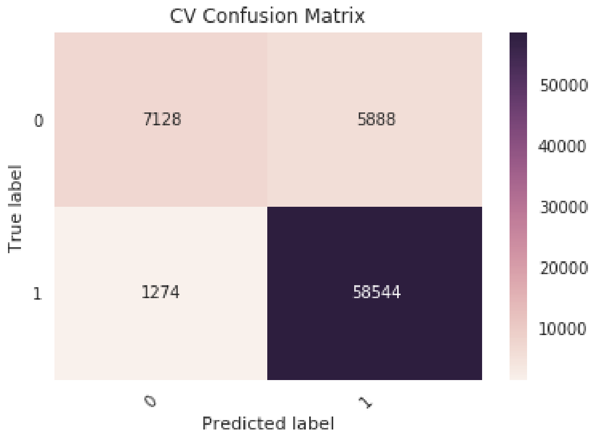
Alpha = 0.01

In [57]:

```
NB(0.01,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
------------------------------------------------------------------
 Coeff   Word    Coeff   Word
------------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
------------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.85      0.55      0.67     13016
          1       0.91      0.98      0.94     59818

avg / total       0.90      0.90      0.89     72834


The TPR is :  0.9787020629242034

The FPR is :  0.0984319101273864

The TNR is :  0.5476336816226183

The FNR is :  0.09787953288260602
```

Out[57]:



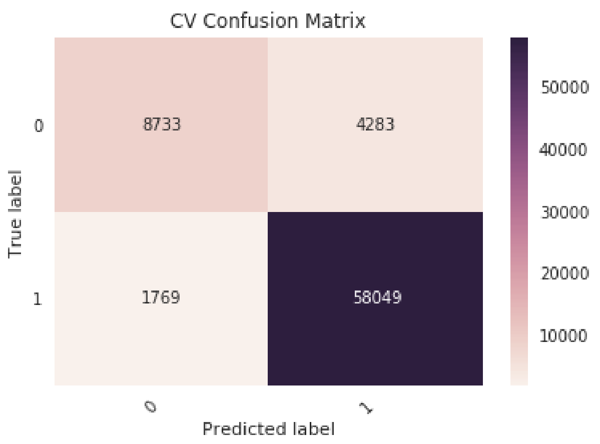
Alpha = 0.1

In [58]:

```
NB(0.1,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff   Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.83      0.67      0.74     13016
          1       0.93      0.97      0.95     59818

avg / total       0.91      0.92      0.91     72834


The TPR is :  0.9704269617840784

The FPR is :  0.07160052158213247

The TNR is :  0.6709434542102028

The FNR is :  0.1359096496619545
```

Out[58]:



Alpha = 1

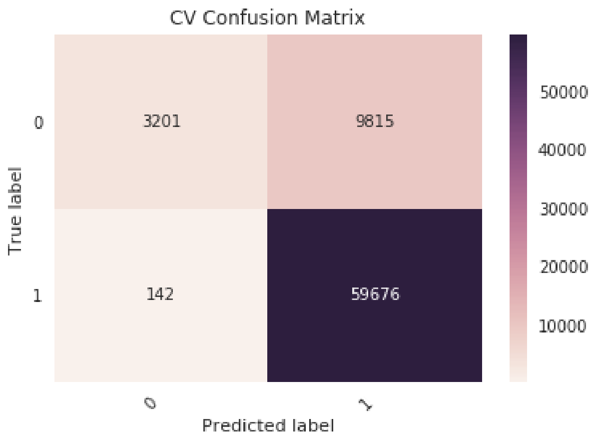
In [59]:

```
NB(1,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
```

```
 Coeff  Word   Coeff  Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
 7885.0000 flavor          44538.0000 flavor
 7721.0000 coffee          43869.0000 love
 6475.0000 dont            43634.0000 product
 5994.0000 get             43537.0000 coffee
 5993.0000 even            33713.0000 get
 5944.0000 tea             32158.0000 would
 5693.0000 food            31347.0000 really
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.96      0.25      0.39     13016
          1       0.86      1.00      0.92     59818

avg / total       0.88      0.86      0.83     72834


The TPR is :  0.9976261326022268

The FPR is :  0.1640810458390451

The TNR is :  0.2459280885064536

The FNR is :  0.010909649661954518
```

Out[59]:



Alpha = 10

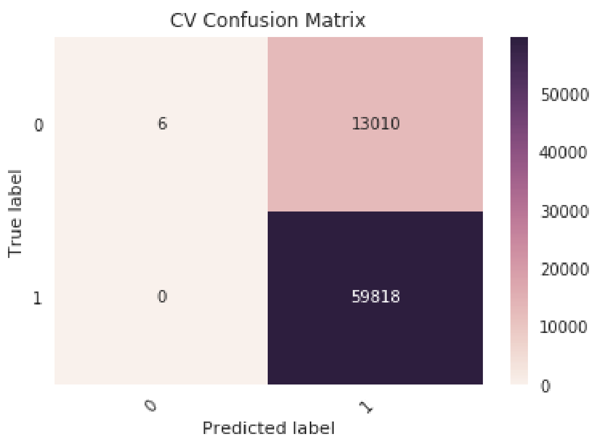
In [60]:

```
NB(10,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff  Word   Coeff  Word
-----------------------------------------------------------------
 16505.0000 like            73883.0000 like
 12465.0000 taste           66173.0000 good
 12421.0000 product         62479.0000 great
 10485.0000 one             53075.0000 one
 9730.0000 would           50046.0000 taste
 8273.0000 good            48679.0000 tea
```

```
 8273.0000 good             48679.0000 tea
 7885.0000 flavor           44538.0000 flavor
 7721.0000 coffee           43869.0000 love
 6475.0000 dont             43634.0000 product
 5994.0000 get              43537.0000 coffee
 5993.0000 even             33713.0000 get
 5944.0000 tea              32158.0000 would
 5693.0000 food             31347.0000 really
 5640.0000 amazon           30654.0000 amazon
 5493.0000 buy              30277.0000 use
 5400.0000 much             30227.0000 food
 5201.0000 really           30124.0000 best
 4512.0000 tried            29085.0000 also
 4499.0000 box              28472.0000 much
 4200.0000 time             28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       1.00      0.00      0.00     13016
          1       0.82      1.00      0.90     59818

avg / total       0.85      0.82      0.74     72834


The TPR is :  1.0

The FPR is :  0.21749306228894313

The TNR is :  0.00046097111247695143

The FNR is :  0.0
```

Out[60]:



Alpha = 100

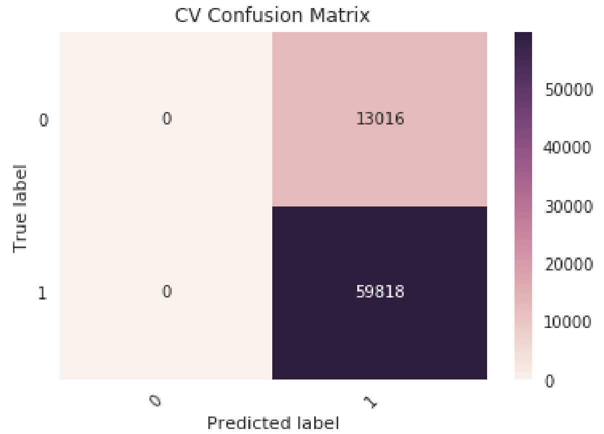
In [61]:

```
NB(100,TFIDF_train,Y_train,TFIDF_CV,Y_CV,TFIDF_dict_bigram,'CV')
```

```
 Negative     Positive
-----------------------------------------------------------------
 Coeff   Word    Coeff   Word
-----------------------------------------------------------------
 16505.0000 like             73883.0000 like
 12465.0000 taste            66173.0000 good
 12421.0000 product          62479.0000 great
 10485.0000 one              53075.0000 one
 9730.0000 would            50046.0000 taste
 8273.0000 good             48679.0000 tea
 7885.0000 flavor           44538.0000 flavor
 7721.0000 coffee           43869.0000 love
 6475.0000 dont             43634.0000 product
 5994.0000 get              43537.0000 coffee
 5993.0000 even             33713.0000 get
 5944.0000 tea              32158.0000 would
 5693.0000 food             31347.0000 really
```

```
 5640.0000 amazon          30654.0000 amazon
 5493.0000 buy             30277.0000 use
 5400.0000 much            30227.0000 food
 5201.0000 really          30124.0000 best
 4512.0000 tried           29085.0000 also
 4499.0000 box             28472.0000 much
 4200.0000 time            28071.0000 find
-----------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.00      0.00      0.00     13016
          1       0.82      1.00      0.90     59818

avg / total       0.67      0.82      0.74     72834


The TPR is :  1.0

The FPR is :  0.21759336654518707

The TNR is :  0.0

The FNR is :  0.0
```

Out[61]:



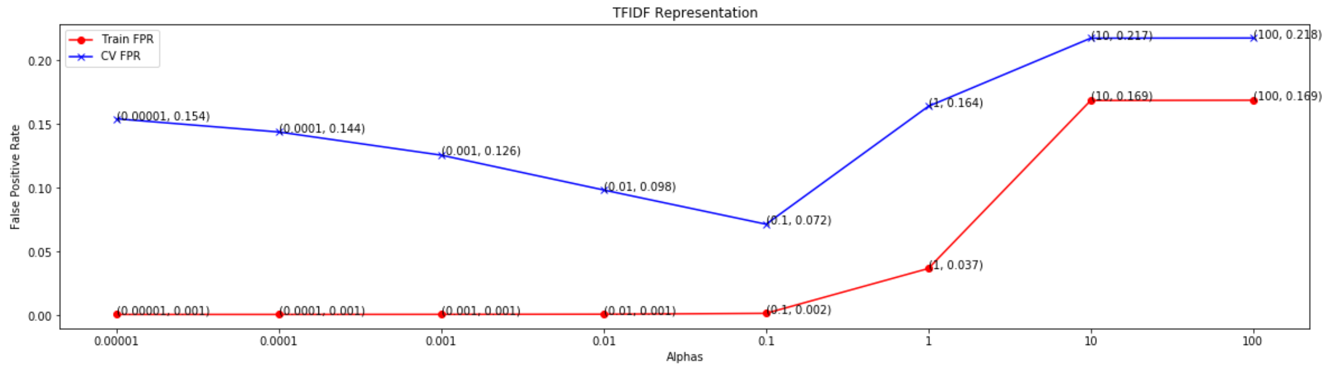
Optimal

In [2]:

```
best(FP_train_dict,FP_test_dict,'TFIDF')
```

```
The optimal Alpha value is :  0.1 with FPR         :  0.07160052158213247
```

Out[2]:

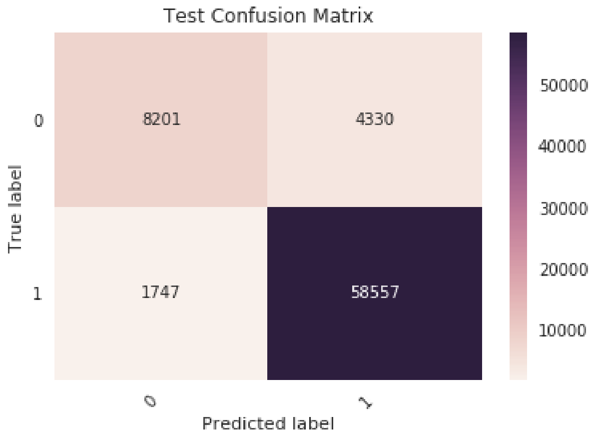


In [63]:

```
NB(0.1,TFIDF_train,Y_train,TFIDF_test,Y_test,TFIDF_dict_bigram,'Test')
```

```
             precision    recall  f1-score   support

          0       0.82      0.65      0.73     12531
          1       0.93      0.97      0.95     60304

avg / total       0.91      0.92      0.91     72835


The TPR is :  0.9710301140886177

The FPR is :  0.07180286548156009

The TNR is :  0.6544569467720054

The FNR is :  0.13941425265341953
```

Out[63]:



Conclusion

Model	Alpha	CV FPR	Test FPR
BoW - Bi Gram	0.1	7.2%	7.2%
TFIDF - Bi Gram	0.1	7.2%	7.2%

Bag of words and TFIDF both produce the same results.