2) On the Handwritten dataset attached, evaluate the bayes classifier with naive bayes assumption. You may rescale the pixel values to 0,1. As described in class, split the dataset randomly to 80:20 to training and test dataset.

In [1]:
```
# Importing libraries and important modules
import numpy as np
import seaborn as sns
from sklearn import *
from sklearn import metrics
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
```

In [3]:
```
# Loading the data from the csv file
dataset = np.loadtxt("A_Z Handwritten Data.csv", delimiter=",")
```

In [4]:
```
# Storing the labels of the datapoints
labels = np.ndarray.flatten(np.hsplit(dataset,[1,785])[0])

# Normalising the data
non_scaled_features = np.hsplit(dataset,[1,785])[1]
features = (non_scaled_features/128).astype(int)
```

In [5]:
```
# Splitting the data into training set and testing set.
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.2, random_
state=1090)

# Declaring the Naive Bayes Multinomial classifier.
model = MultinomialNB()

# Fitting the training data into the model.
model.fit(X_train, y_train)
```

Out[5]:
```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

In [6]:
```
# Making prediction on the test data.
y_pred = model.predict(X_test)
```

In [7]:
```
# Generating the Confusion Matrix and Classification Report for Bayes Classifier.
confMatrix = confusion_matrix(y_test, y_pred)
report = metrics.classification_report(y_test, y_pred)
```

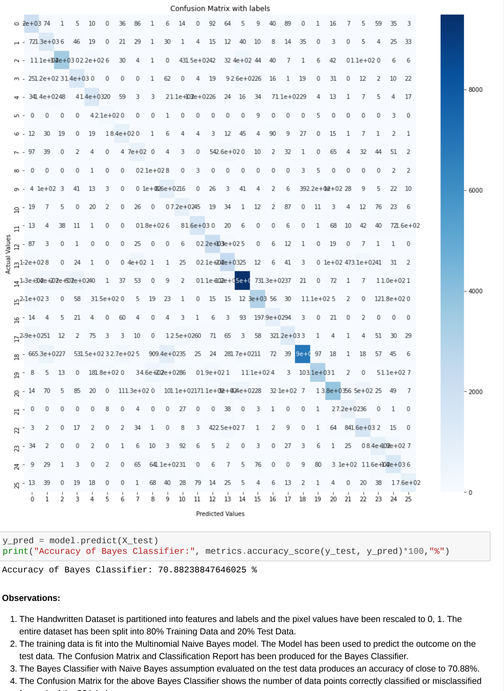
In [8]:
```
print(confMatrix)
```

```
[[2049   74    1    5   10    0   36   86    1    6   14    0   92   64
     5    9   40   89    0    1   16    7    5   59   35    3]
 [  72 1289    6   46   19    0   21   29    1   30    1    4   15   12
    40   10    8   14   35    0    3    0    5    4   25   33]
 [   1  110 3419    0  219    6   30    4    1    0   43  151   42   32
   402   44   40    7    1    6   42    0  106    0    6    6]
 [  25  116    3 1384    0    0    0    0    1   62    0    4   19    9
   264   26   16    1   19    0   31    0   12    2   10   22]
 [  34  138   48    4 1399   20   59    3    3    2  111  126   26   24
    16   34    7  110   29    4   13    1    7    5    4   17]
 [   0    0    0    0    4  211    0    0    0    1    0    0    0    0
     0    9    0    0    0    5    0    0    0    0    3    0]
 [  12   30   19    0   19    1  843    0    1    6    4    4    3   12
    45    4   90    9   27    0   15    1    7    1    2    1]
 [  97   39    0    2    4    0    4  703    0    4    3    0   54  260
     0   10    2   32    1    0   65    4   32   44   51    2]
 [   0    0    0    0    1    0    0    0  206    8    0    3    0    0
     0    0    0    0    3    0    5    0    0    2    2]
 [   4  103    3   41   13    3    0    0  104  857   16    0   26    3
    41    4    2    6   39  219  101   28    9    5   22   10]
 [  19    7    5    0   20    2    0   26    0    0  715   45   19   34
     1   12    2   87    0   11    3    4   12   76   23    6]
 [  13    4   38   11    1    0    0    0  177    6    8 1589    0   20
     6    0    0    6    0    1   68   10   42   40   72  164]
 [  87    3    0    1    0    0    0   25    0    0    6    0 2165  126
     5    0    6   12    1    0   19    0    7    1    1    6]
 [ 120    8    0   24    1    0    0  397    1    1   25    0  212 2376
    25   12    6   41    3    0  104   47  312   41   31    2]
 [ 126  341  267  570   40    1   37   53    0    9    2    0  107  120
   902   73  129   37   21    0   72    1    7    1  105    1]
 [ 213    3    0   58    3  151    0    5   19   23    1    0   15   15
    12 2981   56   30    1  111    5    2    0   12  181    0]
 [  14    4    5   21    4    0   60    4    0    4    3    1    6    3
    93   19  794   94    3    0   21    0    2    0    0    0]
 [ 289   51   12    2   75    3    3   10    0    1  246   60   71   65
     3   58   32 1210    3    1    4    1    4   51   30   29]
 [  66  533   27   53  146    3  273    5   90  944   35   25   24   28
   171   11   72   39 6907   97   18    1   10   57   45    6]
 [   8    5   13    0   18  176    0    3  458  220   86    0  191    1
     1  107    4    3   10 3072    1    2    0    5  106    7]
 [  14   70    5   85   20    0   11  127    0   10  107   17  109  296
   438   28   32  100    7    1 3775   56  502   25   49    7]
 [   0    0    0    0    0    8    0    4    0    0   27    0    0   38
     0    3    1    0    0    2  725   36    0    1    0]
 [   3    2    0   17    2    0    2   34    1    0    8    3   42  249
     7    1    2    9    0    1   64   84 1612    2   15    0]
 [  34    2    0    0    2    0    1    6   10    3   92    6    5    2
     0    3    0   27    3    6    1   25    0  845  186    7]
 [   9   29    1    3    0    2    0   65   64  110   31    0    6    7
     5   76    0    0    9   80    3  100    1  164 1418    6]
 [  13   39    0   19   18    0    0    1   68   40   28   79   14   25
     5    4    6   13    2    1    4    0   20   38    1  765]]
```

In [9]:
```
print(report)
```

```
              precision    recall  f1-score   support

         0.0       0.62      0.76      0.68      2707
         1.0       0.43      0.75      0.55      1722
         2.0       0.88      0.72      0.80      4718
         3.0       0.59      0.68      0.63      2026
         4.0       0.69      0.62      0.65      2244
         5.0       0.36      0.91      0.51       233
         6.0       0.61      0.73      0.66      1156
         7.0       0.44      0.50      0.47      1413
         8.0       0.17      0.90      0.29       230
         9.0       0.37      0.52      0.43      1659
        10.0       0.44      0.63      0.52      1129
        11.0       0.75      0.70      0.72      2276
        12.0       0.66      0.88      0.76      2465
        13.0       0.62      0.63      0.62      3789
        14.0       0.86      0.82      0.84      1612
        15.0       0.84      0.76      0.80      3897
        16.0       0.59      0.69      0.63      1155
        17.0       0.61      0.52      0.56      2314
        18.0       0.97      0.71      0.82      9694
        19.0       0.85      0.68      0.76      4497
        20.0       0.85      0.64      0.73      5891
        21.0       0.66      0.86      0.75       846
        22.0       0.58      0.75      0.66      2160
        23.0       0.57      0.67      0.62      1266
        24.0       0.58      0.65      0.61      2189
        25.0       0.70      0.64      0.67      1203

    accuracy                           0.71     74491
   macro avg       0.63      0.70      0.64     74491
weighted avg       0.75      0.71      0.72     74491
```

In [10]:
```
# Defining the size of the image
fig, ax = plt.subplots(figsize=(15, 15))

# Seaborn Heatmap of the Confusion Matrix
ax = sns.heatmap(confMatrix, annot=True, cmap='Blues')
ax.set_title('Confusion Matrix with labels')
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values')

# To Display the visualization of the Confusion Matrix
plt.show()
```



In [11]:
```
y_pred = model.predict(X_test)
print("Accuracy of Bayes Classifier:", metrics.accuracy_score(y_test, y_pred)*100,"%")
```

```
Accuracy of Bayes Classifier: 70.88238847646025 %
```

Observations:

1. The Handwritten Dataset is partitioned into features and labels and the pixel values have been rescaled to 0, 1. The entire dataset has been split into 80% Training Data and 20% Test Data.
2. The training data is fit into the Multinomial Naive Bayes model. The Model has been used to predict the outcome on the test data. The Confusion Matrix and Classification Report has been produced for the Bayes Classifier.
3. The Bayes Classifier with Naive Bayes assumption evaluated on the test data produces an accuracy of close to 70.88%.
4. The Confusion Matrix for the above Bayes Classifier shows the number of data points correctly classified or misclassified for each of the 26 labels.
5. We observe that the diagonal of the Confusion Matrix is significantly distinct and darker indicating that the majority of the points of a particular label were correctly predicted & classified while a few of the points were misclassified as other labels.
6. We can see that the Multinomial Naive Bayes Classifier works with suitable accuracy for classification on datasets with discrete features.