Library used for this program are

- panda
- numpy
- sklearn.model_selection #To split model into train/test data
- sklearn.naive_bayes #To use GaussianNB and BernoulliNB
- sklearn.ensemble #To use RandomForestClassifier
- PIL #For Image modification
- scipy.misc #For image resize

Importing MNIST Data

In [5]:
```
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import numpy as np
print('Importing data ...')
from sklearn.datasets import fetch_mldata
mnist = fetch_mldata('MNIST original')
x = mnist.data
y = mnist.target
```

```
Importing data ...
```

Data Preparation

In [48]:
```
print('Data prep ...')
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(x, y,test_size=0.20,random_state=42)
```

```
Data prep ...
```

Running Gaussian Model - untouched images

In [49]:
```
from sklearn.naive_bayes import GaussianNB
modelG = GaussianNB()
modelG.fit(train_x,train_y)
```

Out[49]:
```
GaussianNB(priors=None)
```

Running Bernoulli Model - untouched images

In [50]:
```
from sklearn.naive_bayes import BernoulliNB
modelB = BernoulliNB()
modelB.fit(train_x,train_y)
```

Out[50]:
```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
```

Running Decision Forest - untouched images

In [51]:
```
#Random Forest Model depth 4, tree 10
from sklearn.ensemble import RandomForestClassifier
rf1 = RandomForestClassifier(max_depth=4,n_estimators=10)
rf1.fit(train_x, train_y)

#Random Forest Model depth 8, tree 10
from sklearn.ensemble import RandomForestClassifier
rf2 = RandomForestClassifier(max_depth=8,n_estimators=10)
rf2.fit(train_x, train_y)

#Random Forest Model depth 16, tree 10
from sklearn.ensemble import RandomForestClassifier
rf3 = RandomForestClassifier(max_depth=16,n_estimators=10)
rf3.fit(train_x, train_y)

#Random Forest Model depth 4, tree 20
from sklearn.ensemble import RandomForestClassifier
rf4 = RandomForestClassifier(max_depth=4,n_estimators=20)
rf4.fit(train_x, train_y)

#Random Forest Model depth 8, tree 20
from sklearn.ensemble import RandomForestClassifier
```

```
rf5 = RandomForestClassifier(max_depth=8,n_estimators=20)
rf5.fit(train_x, train_y)

#Random Forest Model depth 16, tree 20
from sklearn.ensemble import RandomForestClassifier
rf6 = RandomForestClassifier(max_depth=16,n_estimators=20)
rf6.fit(train_x, train_y)

#Random Forest Model depth 4, tree 30
from sklearn.ensemble import RandomForestClassifier
rf7 = RandomForestClassifier(max_depth=4,n_estimators=30)
rf7.fit(train_x, train_y)

#Random Forest Model depth 8, tree 30
from sklearn.ensemble import RandomForestClassifier
rf8 = RandomForestClassifier(max_depth=8,n_estimators=30)
rf8.fit(train_x, train_y)

#Random Forest Model dpeth 16, tree 30
rf9 = RandomForestClassifier(max_depth=16,n_estimators=30)
rf9.fit(train_x, train_y)
```

Out[51]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=16, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=30, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [52]:
```
#Image crop, boundbox and resize
from PIL import Image
from scipy.misc import imresize

def rescale_strech_image(image):
    image_data = np.asarray(image)
    image_data_bw = np.reshape(image_data, (28, 28))
    non_empty_columns = np.where(image_data_bw.max(axis=0) > 128)[0]
    non_empty_rows = np.where(image_data_bw.max(axis=1) > 128)[0]
    cropBox = (min(non_empty_rows), max(non_empty_rows), min(non_empty_columns), max(non_empty_columns))
    #print(cropBox)
    image_data_new = image_data_bw[cropBox[0]:cropBox[1] + 1, cropBox[2]:cropBox[3] + 1]
    #image_data_new = np.resize(image_data_new, (20, 20))
    image_data_new = imresize(image_data_new, (20, 20))
    #image_data_new.show()
    return (np.array(image_data_new).astype(np.uint8))

train_modified = np.apply_along_axis(rescale_strech_image, axis=1, arr=train_x)
test_modified = np.apply_along_axis(rescale_strech_image, axis=1, arr=test_x)

train_final = np.reshape(train_modified, (train_modified.shape[0], 400))
test_final = np.reshape(test_modified, (test_modified.shape[0], 400))
```

Running Gaussian Model - stretched bounding box

In [53]:
```
from sklearn.naive_bayes import GaussianNB
modelG1 = GaussianNB()
modelG1.fit(train_final,train_y)
```

Out[53]: GaussianNB(priors=None)

Running Bernoulli Model - stretched bounding box

In [54]:
```
from sklearn.naive_bayes import BernoulliNB
modelB1 = BernoulliNB()
modelB1.fit(train_final,train_y)
```

Out[54]:
```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
```

Running Decision Forest - stretched bounding box

In [55]:
```
#Random Forest Model depth 4, tree 10
from sklearn.ensemble import RandomForestClassifier
rf11 = RandomForestClassifier(max_depth=4,n_estimators=10)
rf11.fit(train_final, train_y)

#Random Forest Model depth 8, tree 10
from sklearn.ensemble import RandomForestClassifier
rf12 = RandomForestClassifier(max_depth=8,n_estimators=10)
rf12.fit(train_final, train_y)

#Random Forest Model depth 16, tree 10
from sklearn.ensemble import RandomForestClassifier
rf13 = RandomForestClassifier(max_depth=16,n_estimators=10)
rf13.fit(train_final, train_y)

#Random Forest Model depth 4, tree 20
from sklearn.ensemble import RandomForestClassifier
rf14 = RandomForestClassifier(max_depth=4,n_estimators=20)
rf14.fit(train_final, train_y)
#print(rf14.score(test_final, test_y))
#print(rf1.score(train_x, train_y))

#Random Forest Model depth 8, tree 20
from sklearn.ensemble import RandomForestClassifier
rf15 = RandomForestClassifier(max_depth=8,n_estimators=20)
rf15.fit(train_final, train_y)

#Random Forest Model depth 16, tree 20
from sklearn.ensemble import RandomForestClassifier
rf16 = RandomForestClassifier(max_depth=16,n_estimators=20)
rf16.fit(train_final, train_y)
```

```
#Random Forest Model depth 4, tree 30
from sklearn.ensemble import RandomForestClassifier
rf17 = RandomForestClassifier(max_depth=4,n_estimators=30)
rf17.fit(train_final, train_y)

#Random Forest Model depth 8, tree 30
from sklearn.ensemble import RandomForestClassifier
rf18 = RandomForestClassifier(max_depth=8,n_estimators=30)
rf18.fit(train_final, train_y)

#Random Forest Model dpeth 16, tree 30
rf19 = RandomForestClassifier(max_depth=16,n_estimators=30)
rf19.fit(train_final, train_y)
```

Out[55]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=16, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=30, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

Part A Final Result

In [56]:
```
print('Accuracy             ' + ' GaussianNB      ' + '  Bernoulli')
print('untouched images       ' + str(modelG.score(test_x, test_y)) + '    ' + str(modelB.score(test_x, test_y
)))
print('stretched bounding box ' + str(modelG1.score(test_final, test_y)) + '    ' + str(modelB1.score(test_fin
al, test_y)))
print(' ')
```

```
Accuracy             GaussianNB        Bernoulli
untouched images       0.557857142857   0.832428571429
stretched bounding box 0.842857142857   0.791428571429
```

1. For untouched pixels Bernoulli model is better

2. For stretched bounding box Gaussian model is better

Part B Final Result

In [59]:
```
print('Untouched raw' + '    depth=4         ' + ' depth=8          ' + 'depth=16      ')
print('trees = 10       ' + str(rf1.score(test_x, test_y)) + '   ' + str(rf2.score(test_x, test_y)) + '   ' +
 str(rf3.score(test_x, test_y)))
print('trees = 20       ' + str(rf4.score(test_x, test_y)) + '   ' + str(rf5.score(test_x, test_y)) + '   ' +
 str(rf6.score(test_x, test_y)))
print('trees = 30       ' + str(rf7.score(test_x, test_y)) + '   ' + str(rf8.score(test_x, test_y)) + '
      ' + str(rf9.score(test_x, test_y)))
print(' ')

print('Stretched BBox       ' + ' depth=4      ' + ' depth=8      ' + 'depth=16      ')
print('trees = 10       ' + str(rf11.score(test_final, test_y)) + '   ' + str(rf12.score(test_final, test_y))
 + '   ' + str(rf13.score(test_final, test_y)))
print('trees = 20       ' + str(rf14.score(test_final, test_y)) + '   ' + str(rf15.score(test_final, test_y))
 + '   ' + str(rf16.score(test_final, test_y)))
print('trees = 30       ' + str(rf17.score(test_final, test_y)) + '   ' + str(rf18.score(test_final, test_y))
 + '           ' + str(rf19.score(test_final, test_y)))
```

```
Untouched raw    depth=4          depth=8          depth=16
trees = 10       0.740214285714   0.902857142857   0.944
trees = 20       0.784214285714   0.915785714286   0.956357142857
trees = 30       0.792571428571   0.919            0.963285714286

Stretched BBox        depth=4       depth=8      depth=16
trees = 10       0.770285714286   0.918214285714   0.954214285714
trees = 20       0.780428571429   0.923928571429   0.964785714286
trees = 30       0.779285714286   0.9275           0.965428571429
```

For Decision Forest, it seems that higher the depth/trees, it gets better. Results are also significantly better than Bernoulli or Gaussian models.