code

January 28, 2017

0.1 1. Shuffle Data

In [1]:
```
import numpy as np
import scipy
import scipy.io
from random import shuffle

mnistTrain = []
mnistTrainLabels = []
mnistValid = []
mnistValidLabels = []

spamValid = []
spamValidLabels = []
spamTrain = []
spamTrainLabels = []

cifarValid = []
cifarValidLabels = []
cifarTrain = []
cifarTrainLabels = []

mnist = scipy.io.loadmat("./hw01_data/mnist/train")
np.random.shuffle(mnist["trainX"])
#print(mnist["testX"][0])
#print(mnist["trainX"][0])

numMnistTrain = 5000 # <= 50000

numMnistValid = 10000
for i in range(numMnistValid):
    mnistValid.append(mnist["trainX"][i][0:-1])
    mnistValidLabels.append(mnist["trainX"][i][-1])
for i in range(numMnistValid, numMnistValid + numMnistTrain):
    mnistTrain.append(mnist["trainX"][i][0:-1])
    mnistTrainLabels.append(mnist["trainX"][i][-1])

#----------
```

```
        spam = scipy.io.loadmat("./hw01_data/spam/spam_data")
        #print(spam)

        indices = [i for i in range(len(spam["training_data"]))]
        shuffle(indices)

        spamValidLen = int(0.2*len(spam["training_data"]))
        #np.random.shuffle(spam["training_data"])
        for i in range(spamValidLen):
            spamValid.append(spam["training_data"][indices[i]])
            spamValidLabels.append(spam["training_labels"][0][indices[i]])
        for i in range(len(spam["training_data"])):
            spamTrain.append(spam["training_data"][indices[i]])
            spamTrainLabels.append(spam["training_labels"][0][indices[i]])
        #for i in range(spamValidLen):
            #del spamValid[i][-1]

        #print(spamValid)
        #print(spamTrainLabels)

        #----------

        cifar = scipy.io.loadmat("./hw01_data/cifar/train")
        np.random.shuffle(cifar["trainX"])
        print(cifar)

        numCifarValid = 5000
        numCifarTrain = 5000
        for i in range(numCifarValid):
            cifarValid.append(cifar["trainX"][i])
            cifarValidLabels.append(cifar["trainX"][i][-1])
        for i in range(numCifarValid, numCifarValid + numCifarTrain):
            cifarTrain.append(cifar["trainX"][i])
            cifarTrainLabels.append(cifar["trainX"][i][-1])
        #for i in range(numCifar):
            #del cifarValid[i][-1]

        #print(cifarValid)
        #print(cifarTrainLabels)
```

```
{'trainX': array([[188, 204, 201, ..., 216, 217,   8],
       [140, 138, 138, ..., 124, 123,   0],
       [136, 135, 132, ...,  75,  73,   5],
       ...,
       [136, 137, 127, ..., 139, 129,   4],
       [178, 179, 181, ..., 104, 103,   9],
       [ 20,  21,  27, ...,  28,  26,   0]], dtype=int64), '__globals__': [], '__he
```

In [2]:
```
from sklearn import svm
from sklearn import model_selection

import math

import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
import warnings
warnings.filterwarnings('ignore')

trainLens = [100, 200, 500, 1000, 2000, 5000, 10000]

for trainLen in trainLens:
    clf = svm.SVC(kernel='linear')
    clf.fit(mnistTrain[0:trainLen], mnistTrainLabels[0:trainLen])
    success = 0
    tries = 0
    for i in range(len(mnistValid)):
        if ((clf.predict(mnistValid[i]))[0] == mnistValidLabels[i]):
            success += 1
        tries += 1
    print("Training set size: " + str(trainLen) + ", accuracy: " + str(succ
```

```
Training set size: 100, accuracy: 0.748
Training set size: 200, accuracy: 0.8188
Training set size: 500, accuracy: 0.8531
Training set size: 1000, accuracy: 0.8751
Training set size: 2000, accuracy: 0.8926
Training set size: 5000, accuracy: 0.9059
Training set size: 10000, accuracy: 0.9099
```

In [29]:
```
#clf = svm.SVC(kernel='linear')
#clf.fit(spamTrain, spamTrainLabels)

trainLens = [100, 200, 500, 1000, 2000, 4000]

for trainLen in trainLens:
    clf = svm.SVC(kernel='linear')
    #print(spamTrainLabels)
    clf.fit(spamTrain[0:trainLen], spamTrainLabels[0:trainLen])
    success = 0
    tries = 0
    for i in range(len(spamValid)):
        if ((clf.predict(spamValid[i]))[0] == spamValidLabels[i]):
            success += 1
```

```
                tries += 1
            print("Training set size: " + str(trainLen) + ", accuracy: " + str(suc
```

```
Training set size: 100, accuracy: 0.7446808510638298
Training set size: 200, accuracy: 0.7794970986460348
Training set size: 500, accuracy: 0.7920696324951644
Training set size: 1000, accuracy: 0.7978723404255319
Training set size: 2000, accuracy: 0.8056092843326886
Training set size: 4000, accuracy: 0.7959381044487428
```

In [ ]:
```
trainLen = 40000
clf = svm.SVC(kernel='linear')
clf.fit(mnistTrain[0:trainLen], mnistTrainLabels[0:trainLen])

mnistTestSet = scipy.io.loadmat("./hw01_data/mnist/test")["testX"]
results = []
#print(mnistTestSet)
#print("Id,Category")
for i in range(len(mnistTestSet)):
    #print( clf.predict(mnistTestSet[i]) )
    #print(str(i) + "," + str(clf.predict(mnistTestSet[i])[0]))
    results.append([i, clf.predict(mnistTestSet[i])[0]])
temp = np.asarray(results)
#temp.tofile("./submission.py")
np.savetxt("submission.csv", temp, fmt="%i,%i", delimiter=",", header="Id,C
```

In [3]:
```
trainLen = len(spamTrain)
clf = svm.SVC(kernel='linear')
clf.fit(spamTrain[0:], spamTrainLabels[0:trainLen])

#spamTestSet = scipy.io.loadmat("./hw01_data/spam/test")["testX"]
spamTestSet = spam["test_data"]
results = []
#print(mnistTestSet)
#print("Id,Category")
for i in range(len(spamTestSet)):
    #print( clf.predict(mnistTestSet[i]) )
    #print(str(i) + "," + str(clf.predict(mnistTestSet[i])[0]))
    results.append([i, clf.predict(spamTestSet[i])[0]])
temp = np.asarray(results)
#temp.tofile("./submission.py")
np.savetxt("submission_spam.csv", temp, fmt="%i,%i", delimiter=",", header=
```

In [5]:
```
#clf = svm.SVC(kernel='linear')
#clf.fit(cifarTrain, cifarTrainLabels)

trainLens = [100, 200, 500, 1000, 2000, 4000]
```

```
        for trainLen in trainLens:
            clf = svm.SVC(kernel='linear')
            clf.fit(cifarTrain[0:trainLen], cifarTrainLabels[0:trainLen])
            success = 0
            tries = 0
            for i in range(len(cifarValid)):
                if ((clf.predict(cifarValid[i]))[0] == cifarValidLabels[i]):
                    success += 1
                tries += 1
            print("Training set size: " + str(trainLen) + ", accuracy: " + str(succ
```

```
Training set size: 100, accuracy: 0.2134
Training set size: 200, accuracy: 0.2526
Training set size: 500, accuracy: 0.283
Training set size: 1000, accuracy: 0.2818
Training set size: 2000, accuracy: 0.2954
Training set size: 4000, accuracy: 0.2982
```

In [30]:
```
import math

hyperParamTrainLen = 1000

for cValue in range(1,15):
    clf = svm.SVC(C=float(1000.0/math.pow(10.0, cValue)), kernel='linear')
    clf.fit(mnistTrain[0:hyperParamTrainLen], mnistTrainLabels[0:hyperPara
    success = 0
    tries = 0
    for i in range(len(mnistValid)):
        if ((clf.predict(mnistValid[i]))[0] == mnistValidLabels[i]):
            success += 1
        tries += 1
    print("Training set size: " + str(hyperParamTrainLen) + ", C: " + str
```

```
Training set size: 1000, C: 100.0, accuracy: 0.8832
Training set size: 1000, C: 10.0, accuracy: 0.8832
Training set size: 1000, C: 1.0, accuracy: 0.8832
Training set size: 1000, C: 0.1, accuracy: 0.8832
Training set size: 1000, C: 0.01, accuracy: 0.8832
Training set size: 1000, C: 0.001, accuracy: 0.8832
Training set size: 1000, C: 0.0001, accuracy: 0.8832
Training set size: 1000, C: 1e-05, accuracy: 0.8832
Training set size: 1000, C: 1e-06, accuracy: 0.8874
Training set size: 1000, C: 1e-07, accuracy: 0.8683
Training set size: 1000, C: 1e-08, accuracy: 0.629
Training set size: 1000, C: 1e-09, accuracy: 0.1121
Training set size: 1000, C: 1e-10, accuracy: 0.1121
Training set size: 1000, C: 1e-11, accuracy: 0.1121
```

In [10]:
```
import math

        for cValueIter in range(1,10):
            cValue = float(10.0/math.pow(10.0, cValueIter))
            #clf = svm.SVC(C=cValue, kernel='linear')
            k = 5
            kf = model_selection.KFold(n_splits=k, shuffle=True)
            avgAccuracy = 0.0
            for train_index, test_index in kf.split(spamTrain):
                #print("TRAIN:", train_index, "TEST:", test_index)
                X_train = spam["training_data"][train_index]
                X_test = spam["training_labels"][0][train_index]
                y_train = spam["training_data"][test_index],
                y_test = spam["training_labels"][0][test_index]
                #clf = svm.SVC(C=float(10.0/math.pow(10.0, cValue)), kernel='linea
                clf = svm.SVC(C=cValue, kernel='linear')
                clf.fit(X_train, X_test)
                success = 0
                tries = 0
                for i in range(len(y_train[0])):
                    if ((clf.predict(y_train[0][i]))[0] == y_test[i]):
                        success += 1
                    tries += 1
                #print("Accuracy: " + str(success/tries) + " " + str(success) + "
                avgAccuracy += success/tries;
            avgAccuracy /= k
            print("Average accuracy: " + str(avgAccuracy) + ", cValue: " + str(cVa
```

```
Average accuracy: 0.8012311832478345, cValue: 1.0
Average accuracy: 0.796017155832142, cValue: 0.1
Average accuracy: 0.7780338070809856, cValue: 0.01
Average accuracy: 0.7519296573505638, cValue: 0.001
Average accuracy: 0.7177147983068428, cValue: 0.0001
Average accuracy: 0.709977854399686, cValue: 1e-05
Average accuracy: 0.709980470757529, cValue: 1e-06
Average accuracy: 0.7099838346461843, cValue: 1e-07
Average accuracy: 0.7099772937515768, cValue: 1e-08
```

In [ ]:

In [ ]:

In [ ]: