LO3 Dummy Classifier

Introduction

In this assignment we are going to classify data. First we test a built-in classifier against the MNIST dataset. After that we implement our own classifier and test it on the MNIST dataset. We then compare the built-in classifier to our own classifier.

Qa Add a Stochastic Gradient Decent [SGD] Classifier

In this exercise we get our dataset. We use a function from an earlier exercise (Datasets). When we get our dataset we split it into a training set and a test set. We then shuffle the data to not have the same instances in a row. From here on we create the classifier and use it to predict a classification of a specific sample in the dataset.

Qa Implementation

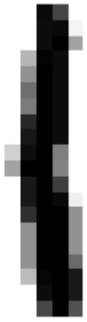
In [11]:

```
from sklearn.linear_model import SGDClassifier
import numpy as np
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
from libitmal import dataloaders as dl

#Fetch MNIST Dataset
X, y = dl.MNIST_GetDataSet()
print(X.shape)
print(y.shape)

#Split Data set into test and train sets
X_train, X_test, y_train, y_test = X[:60000], X[60000:], y[:60000], y[60000:]

#Shuffle the data
shuffle_index = np.random.permutation(60000)
X_train, y_train = X_train[shuffle_index], y_train[shuffle_index]

#Target vectors for classification
y_train_5 = (y_train == '5')
y_test_5 = (y_test == '5')

print("ok")
```

```
(70000, 784)
(70000,)
ok
```

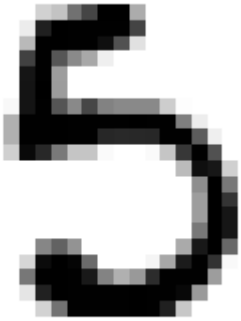
In [14]:

```
#Function for predicting and printing number using SGDClassifier
def Predict_Print_number(some_digit):
    some_digit_pred = X[some_digit]
    some_digit_pred_image = some_digit_pred.reshape(28, 28)

    #Show the image
    plt.imshow(some_digit_pred_image, cmap = matplotlib.cm.binary, interpolation="neare
st")
    plt.axis("off")
    plt.show()

    #Use SGDClassifier to classify if the number is the 5 or not
    sgd_clf = SGDClassifier(random_state=42)
    sgd_clf.fit(X_train, y_train_5)
    pred = sgd_clf.predict([some_digit_pred])
    print(pred)

#False
Predict_Print_number(35000)
#False
Predict_Print_number(41025)
#False
Predict_Print_number(30800)
#True
Predict_Print_number(41000)
```



```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
```

```
[False]
```



```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
```

```
[False]
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)

[False]
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)

[ True]
```

Qa Result

As seen on the printout above the SGDClassifier classfies correctly in some cases. But in one of the cases the five is classifed as false. This makes the classifier less reliable.

Qb Implement a dummy binary classifier

In this exercise we create our own classifier. We compare the SGD classifier with our own classifier by using a validation score, and later a confusion matrix. Through the confusion matrix we see how the classfiers, classified the data.

Qb Implementation

In [15]:

```
from sklearn.base import BaseEstimator
from sklearn.base import ClassifierMixin
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from libitmal import utils as itmalutils
import numpy as np

#Class for the dummy classifier
class DummyClassifier(BaseEstimator):
    def fit(self, X, y=None):
        pass
    def predict(self, X):
        return np.zeros((len(X),1), dtype=bool)

#Dummy classifier object
clf_Obj = DummyClassifier()
clf_Obj.fit(X_train, y_train_5)
clf_pred = clf_Obj.predict([35005])
print(clf_pred)

#SGDClassifier
sgd_clf_two = SGDClassifier(random_state=42)
sgd_clf_two.fit(X_train, y_train_5)

#Cross_val_predict on dummy classifier
y_train_predict=cross_val_predict(clf_Obj, X_train, y_train_5, cv=3)
#Cross_val_predict on SGDClassifier
y_train_predict_sgd=cross_val_predict(sgd_clf_two, X_train, y_train_5, cv=3)

#Cross_val_score on dummy classifier
c=cross_val_score(clf_Obj, X_train, y_train_5, cv=3, scoring="accuracy")
#Cross_val_score on SGDClassifier
c_two=cross_val_score(sgd_clf_two, X_train, y_train_5, cv=3, scoring="accuracy")
print("c_Dummy=",c)
print("c_SGD=",c_two)

#Confusion matrix for dummy classifier
M=confusion_matrix(y_train_5, y_train_predict)
itmalutils.PrintMatrix(M,"M_Dummy=")

#Confusion matrix for SGDClassifer
M_two=confusion_matrix(y_train_5, y_train_predict_sgd)
itmalutils.PrintMatrix(M_two,"M_SGD=")

y_train_5.shape

#Print the number of true and false 5's in the training set
print("True=", sum(y_train_5==True))
print("False=",sum(y_train_5==False))


# Inspiration for running cross evaluation on a model, and printing the confusion matri
x:
# c=cross_val_score(model, X_test, y_test_5, cv=3, scoring="accuracy")
# print("c=",c)
# M=confusion_matrix(y_test_true, y_test_pred)
# itmalutils.PrintMatrix(M,"M=")
```

```
[[False]]

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic
_gradient.py:166: FutureWarning: max_iter and tol parameters have been add
ed in SGDClassifier in 0.19. If both are left unset, they default to max_i
ter=5 and tol=None. If tol is not None, max_iter defaults to max_iter=100
0. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)

c_Dummy= [0.9071  0.91195 0.9099 ]
c_SGD= [0.95795 0.9165  0.96535]
M_Dummy=[[54579     0]
         [ 5421     0]]
M_SGD=[[52765  1814]
       [ 1390  4031]]
True= 5421
False= 54579
```

Qb Result

As seen on the printout above the Dummy classifier has a validation score of 90%. This is because only 10% of the dataset is a 5. Also we see the SDG classifier having a validationscore of 95%. From the confusion matrix we see that our dummy classifier only has True negatives and False negatives. This is because it only sees non-fives. From the SGD's confusion matrix we see a better result since it actually predicts some True positives, but it also predicts some False positives. We count how many true and false there are in the training set and see that the dummy classifiers has the same numbers in its confusion matrix. This makes sense, since it classifies everything as non-five.