Name: Muhammed Emin Kilicaslan

ID: 191401007

Course: BIL570 /BIL470

```
%load_ext autoreload
%autoreload 2

from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from sklearn.preprocessing import label_binarize

from itertools import cycle

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from urllib import request
import gzip
import pickle
import shutil

from logreg import LogisticRegression
```

Veri Setinin Yüklenmesi:

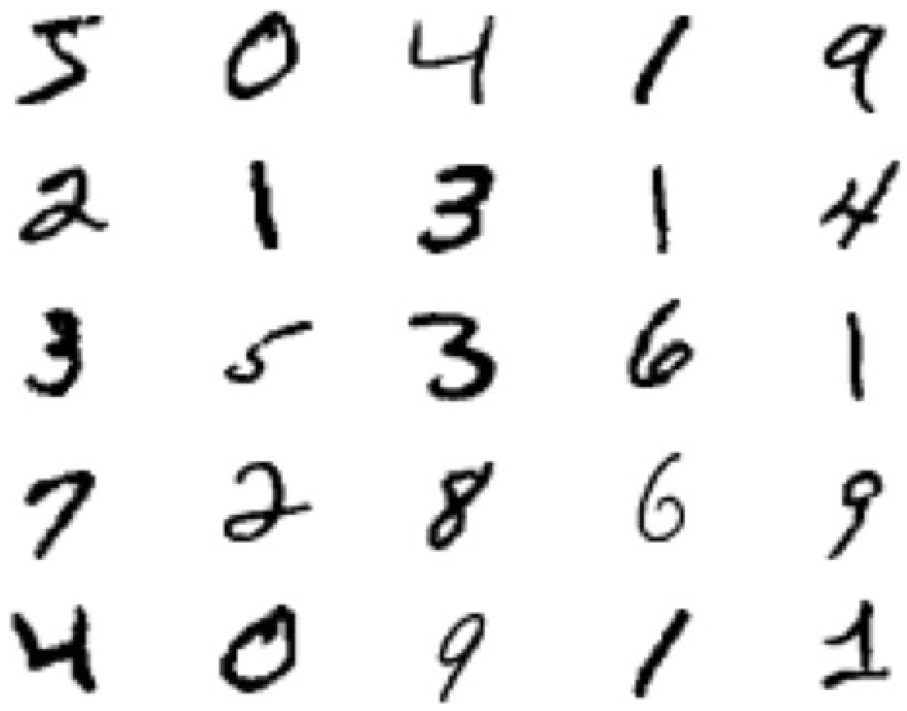
Aşağıdaki adreslerden .gz uzantılı veri seti yüklenmiştir. https://ossci-datasets.s3.amazonaws.com/mnist/train-images-idx3-ubyte.gz https://ossci-datasets.s3.amazonaws.com/mnist/train-labels-idx1-ubyte.gz https://ossci-datasets.s3.amazonaws.com/mnist/t10k-images-idx3-ubyte.gz https://ossci-datasets.s3.amazonaws.com/mnist/t10k-labels-idx1-ubyte.gz

```
filename = [
["training_images","train-images-idx3-ubyte.gz"],
["test_images","t10k-images-idx3-ubyte.gz"],
["training_labels","train-labels-idx1-ubyte.gz"],
["test_labels","t10k-labels-idx1-ubyte.gz"]
]

def save_mnist():
    with gzip.open("train-images-idx3-ubyte.gz",'rb') as f_in:
        with open("train-images.idx3-ubyte",'wb') as f_out:
            shutil.copyfileobj(f_in,f_out)
    with gzip.open("t10k-images-idx3-ubyte.gz",'rb') as f_in:
        with open("t10k-images.idx3-ubyte",'wb') as f_out:
            shutil.copyfileobj(f_in,f_out)
    with gzip.open("train-labels-idx1-ubyte.gz",'rb') as f_in:
        with open("train-labels.idx1-ubyte",'wb') as f_out:
            shutil.copyfileobj(f_in,f_out)
    with gzip.open("t10k-labels-idx1-ubyte.gz",'rb') as f_in:
        with open("t10k-labels.idx1-ubyte",'wb') as f_out:
            shutil.copyfileobj(f_in,f_out)

    mnist = {}
    for name in filename[:2]:
        with gzip.open(name[1], 'rb') as f:
            mnist[name[0]] = np.frombuffer(f.read(), np.uint8,
offset=16).reshape(-1,28*28)
    for name in filename[-2:]:
        with gzip.open(name[1], 'rb') as f:
            mnist[name[0]] = np.frombuffer(f.read(), np.uint8,
offset=8)
    with open("mnist.pkl", 'wb') as f:
        pickle.dump(mnist,f)
    print("Save complete.")

def init():
    save_mnist()

def load():
    with open("mnist.pkl",'rb') as f:
        mnist = pickle.load(f)
    return mnist["training_images"], mnist["training_labels"],
mnist["test_images"], mnist["test_labels"];

def load_list():
    with open("mnist.pkl",'rb') as f:
        mnist = pickle.load(f)
    return mnist["training_images"].tolist(),
mnist["training_labels"].tolist(), mnist["test_images"].tolist(),
mnist["test_labels"].tolist()
init();
```

```
Save complete.
```

Exploratory Data Analysis (EDA)

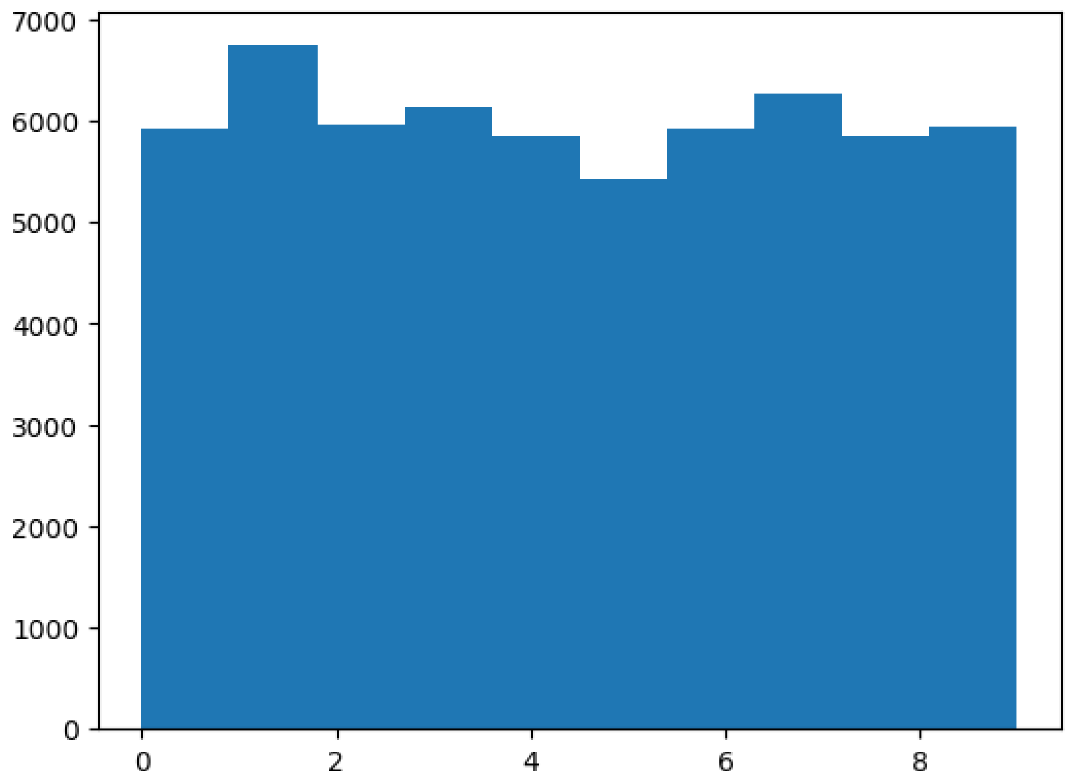
```
data = load()
data_list = load_list()

X_train, y_train, X_test, y_test = (i for i in data)

for i in range(25):
    plt.subplot(5, 5, i + 1)
    plt.imshow(X_train[i].reshape(28, -1), cmap = "gray_r")
    plt.axis("off")
plt.show()
```



```
X_train_pd = pd.DataFrame(X_train)
y_train_pd = pd.Series(y_train)
X_train_pd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 60000 entries, 0 to 59999
Columns: 784 entries, 0 to 783
dtypes: uint8(784)
memory usage: 44.9 MB
```

```
counts, bins, hist = plt.hist(y_train)
```



Veri ön işleme

Logistic Regresyon Modelinin Eğitilmesi

$$\frac{dL}{dW_i} = (\hat{y}_i - y_i) * \mathrm{x}$$

$$\frac{dL}{db_i} = \hat{y}_i - y_i$$

```
lr=LogisticRegression(0.01,15,100);

lr.fit(X_train,y_train);

-----------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
Cell In[209], line 1
----> 1 lr.fit(X_train,y_train);

File ~\OneDrive\Desktop\Python\Yap470HW2\logreg.py:29, in
```

```
LogisticRegression.fit(self, X, y)
     27 for i, label in enumerate(batch_y):
     28     loss[i][label] = 1 - softmax[i][label]
---> 29 grad_w, grad_b = self._compute_gradients(batch_X, loss)
     30 self.weights = [[w - self.learning_rate * gw for w, gw in
zip(ws, grad_ws)] for ws, grad_ws in zip(self.weights, grad_w)]
     31 self.b = [b - self.learning_rate * gb for b, gb in zip(self.b,
grad_b)]

File ~\OneDrive\Desktop\Python\Yap470HW2\logreg.py:82, in
LogisticRegression._compute_gradients(self, X, loss)
     80     for j in range(num_classes):
     81         for k in range(num_features):
---> 82             grad_w[j][k] += X[i][k] * loss[i][j]
     83         grad_b[j] += loss[i][j]
     85 grad_w = [[gw / num_samples for gw in grad_ws] for grad_ws in
grad_w]

KeyboardInterrupt:
```

Test Değerlerinin Tahmin Edilmesi

```
yhat = lr.predict(X_test)
```

Eğitim Değerlerinin Tahmin Edilmesi

```
xhat = lr.predict(X_train)
```

Tahmin Edilen Test Değerleri ile Beklenen Test Değerlerinin Karşılaştırması

```
print("Beklenen Test Sınıfı Değerleri:")
print(y_test[0:100]);
print("Tahmin Değerleri:")
print(yhat[0:100]);
```

Tahmin Edilen Eğitim Değerleri ile Beklenen Eğitim Değerlerinin Karşılaştırması

```
print("Beklenen Eğitim Sınıfı Değerleri:")
print(y_train[0:600]);
print("Tahmin Değerleri:")
print(xhat[0:600]);
```

Results

```
y_pred2 = pd.Series(yhat);
y_test2 = pd.Series(y_test);
```

Test Verilerinin Confusion Matriksi

Eğitim Verilerinin Confusion Matriksi

Eğitim/Test F1-Score

Eğitim/Test Accuracy

Eğitim/Test Precision

Eğitim/Test Recal

ROC Eğrileri (Test/Eğitim) ve Eğrilerin AUC Değerleri (Test/Eğitim)