Name: Ayse Irmak Ercevik

ID: 123456789

Course: BIL570 /BIL470

In [1]:
```
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
```

In [2]:
```
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

In [3]:
```
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(-1,2
    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"], m

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

```
Save complete.
```

Exploratory Data Analysis (EDA)

Veri ön işleme

Logistic Regresyon Modelinin Eğitilmesi

In [23]:
```
lr=LogisticRegression(0.01,15,100);
```

In [ ]:
```
lr.fit(X_train,y_train);
```

Test Değerlerinin Tahmin Edilmesi

In [27]:
```
yhat = lr.predict(X_test)
```

Eğitim Değerlerinin Tahmin Edilmesi

In [28]:
```
xhat = lr.predict(X_train)
```

Tahmin Edilen Test Değerleri ile Beklenen Test Değerlerinin Karşılaştırması

In [33]:
```
print("Beklenen Test Sınıfı Değerleri:")
print(y_test[0:100]);
print("Tahmin Değerleri:")
print(yhat[0:100]);
```

```
Beklenen Test Sınıfı Değerleri:
[7, 2, 1, 0, 4, 1, 4, 9, 5, 9, 0, 6, 9, 0, 1, 5, 9, 7, 3, 4, 9, 6, 6, 5, 4, 0, 7, 4,
0, 1, 3, 1, 3, 4, 7, 2, 7, 1, 2, 1, 1, 7, 4, 2, 3, 5, 1, 2, 4, 4, 6, 3, 5, 5, 6, 0,
4, 1, 9, 5, 7, 8, 9, 3, 7, 4, 6, 4, 3, 0, 7, 0, 2, 9, 1, 7, 3, 2, 9, 7, 7, 6, 2, 7,
8, 4, 7, 3, 6, 1, 3, 6, 9, 3, 1, 4, 1, 7, 6, 9]
Tahmin Değerleri:
[7, 2, 1, 0, 4, 1, 4, 9, 6, 9, 0, 6, 9, 0, 1, 5, 9, 7, 3, 4, 9, 6, 6, 5, 4, 0, 7, 4,
0, 1, 3, 1, 3, 6, 7, 2, 7, 1, 2, 1, 1, 7, 4, 2, 3, 5, 3, 2, 4, 4, 6, 3, 5, 5, 6, 0,
4, 1, 9, 5, 7, 8, 9, 2, 7, 4, 6, 4, 3, 0, 7, 0, 2, 9, 1, 7, 3, 7, 9, 7, 7, 6, 2, 7,
8, 4, 7, 3, 6, 1, 3, 6, 9, 3, 1, 4, 1, 7, 6, 9]
```

Tahmin Edilen Eğitim Değerleri ile Beklenen Eğitim Değerlerinin Karşılaştırması

In [34]:
```
print("Beklenen Eğitim Sınıfı Değerleri:")
print(y_train[0:600]);
print("Tahmin Değerleri:")
print(xhat[0:600]);
```

```
Beklenen Eğitim Sınıfı Değerleri:
[5, 0, 4, 1, 9, 2, 1, 3, 1, 4, 3, 5, 3, 6, 1, 7, 2, 8, 6, 9, 4, 0, 9, 1, 1, 2, 4, 3,
2, 7, 3, 8, 6, 9, 0, 5, 6, 0, 7, 6, 1, 8, 7, 9, 3, 9, 8, 5, 9, 3, 3, 0, 7, 4, 9, 8,
0, 9, 4, 1, 4, 4, 6, 0, 4, 5, 6, 1, 0, 0, 1, 7, 1, 6, 3, 0, 2, 1, 1, 7, 9, 0, 2, 6,
7, 8, 3, 9, 0, 4, 6, 7, 4, 6, 8, 0, 7, 8, 3, 1, 5, 7, 1, 7, 1, 1, 6, 3, 0, 2, 9, 3,
1, 1, 0, 4, 9, 2, 0, 0, 2, 0, 2, 7, 1, 8, 6, 4, 1, 6, 3, 4, 5, 9, 1, 3, 3, 8, 5, 4,
7, 7, 4, 2, 8, 5, 8, 6, 7, 3, 4, 6, 1, 9, 9, 6, 0, 3, 7, 2, 8, 2, 9, 4, 4, 6, 4, 9,
7, 0, 9, 2, 9, 5, 1, 5, 9, 1, 2, 3, 2, 3, 5, 9, 1, 7, 6, 2, 8, 2, 2, 5, 0, 7, 4, 9,
7, 8, 3, 2, 1, 1, 8, 3, 6, 1, 0, 3, 1, 0, 0, 1, 7, 2, 7, 3, 0, 4, 6, 5, 2, 6, 4, 7,
1, 8, 9, 9, 3, 0, 7, 1, 0, 2, 0, 3, 5, 4, 6, 5, 8, 6, 3, 7, 5, 8, 0, 9, 1, 0, 3, 1,
2, 2, 3, 3, 6, 4, 7, 5, 0, 6, 2, 7, 9, 8, 5, 9, 2, 1, 1, 4, 4, 5, 6, 4, 1, 2, 5, 3,
9, 3, 9, 0, 5, 9, 6, 5, 7, 4, 1, 3, 4, 0, 4, 8, 0, 4, 3, 6, 8, 7, 6, 0, 9, 7, 5, 7,
2, 1, 1, 6, 8, 9, 4, 1, 5, 2, 2, 9, 0, 3, 9, 6, 7, 2, 0, 3, 5, 4, 3, 6, 5, 8, 9, 5,
4, 7, 4, 2, 7, 3, 4, 8, 9, 1, 9, 2, 8, 7, 9, 1, 8, 7, 4, 1, 3, 1, 1, 0, 2, 3, 9, 4,
9, 2, 1, 6, 8, 4, 7, 7, 4, 4, 9, 2, 5, 7, 2, 4, 4, 2, 1, 9, 7, 2, 8, 7, 6, 9, 2, 2,
3, 8, 1, 6, 5, 1, 1, 0, 2, 6, 4, 5, 8, 3, 1, 5, 1, 9, 2, 7, 4, 4, 4, 8, 1, 5, 8, 9,
5, 6, 7, 9, 9, 3, 7, 0, 9, 0, 6, 6, 2, 3, 9, 0, 7, 5, 4, 8, 0, 9, 4, 1, 2, 8, 7, 1,
2, 6, 1, 0, 3, 0, 1, 1, 8, 2, 0, 3, 9, 4, 0, 5, 0, 6, 1, 7, 7, 8, 1, 9, 2, 0, 5, 1,
2, 2, 7, 3, 5, 4, 9, 7, 1, 8, 3, 9, 6, 0, 3, 1, 1, 2, 6, 3, 5, 7, 6, 8, 3, 9, 5, 8,
5, 7, 6, 1, 1, 3, 1, 7, 5, 5, 5, 2, 5, 8, 7, 0, 9, 7, 7, 5, 0, 9, 0, 0, 8, 9, 2, 4,
8, 1, 6, 1, 6, 5, 1, 8, 3, 4, 0, 5, 5, 8, 3, 6, 2, 3, 9, 2, 1, 1, 5, 2, 1, 3, 2, 8,
7, 3, 7, 2, 4, 6, 9, 7, 2, 4, 2, 8, 1, 1, 3, 8, 4, 0, 6, 5, 9, 3, 0, 9, 2, 4, 7, 1,
2, 9, 4, 2, 6, 1, 8, 9, 0, 6, 6, 7]
Tahmin Değerleri:
[3, 0, 4, 1, 9, 2, 1, 3, 1, 4, 3, 8, 3, 6, 1, 7, 2, 8, 6, 9, 4, 0, 9, 1, 3, 2, 4, 3,
7, 7, 3, 8, 6, 7, 0, 5, 6, 0, 7, 6, 1, 8, 7, 9, 3, 9, 8, 5, 8, 3, 3, 0, 7, 4, 9, 8,
0, 9, 4, 1, 4, 4, 6, 0, 4, 5, 6, 1, 0, 0, 2, 7, 1, 6, 3, 0, 2, 1, 1, 7, 8, 0, 2, 6,
7, 8, 3, 9, 0, 4, 6, 7, 4, 6, 8, 0, 7, 8, 3, 1, 5, 7, 1, 7, 1, 1, 6, 3, 0, 6, 9, 3,
1, 1, 0, 4, 9, 2, 0, 0, 7, 0, 2, 7, 1, 8, 6, 4, 1, 6, 3, 4, 3, 9, 5, 3, 3, 8, 0, 4,
7, 7, 4, 8, 8, 5, 8, 6, 9, 3, 4, 6, 1, 9, 9, 6, 0, 3, 9, 2, 8, 2, 9, 4, 4, 6, 4, 9,
7, 0, 9, 2, 7, 8, 1, 5, 9, 1, 0, 3, 2, 8, 3, 9, 1, 7, 6, 2, 8, 2, 2, 5, 0, 7, 4, 9,
7, 8, 3, 2, 1, 1, 8, 3, 6, 1, 0, 3, 1, 0, 0, 1, 9, 2, 7, 3, 0, 4, 6, 5, 2, 6, 4, 7,
7, 8, 9, 9, 5, 0, 7, 1, 6, 2, 0, 3, 5, 4, 6, 5, 8, 6, 3, 7, 8, 8, 0, 9, 1, 0, 6, 1,
2, 2, 3, 3, 6, 4, 7, 5, 0, 6, 0, 7, 4, 8, 5, 9, 7, 1, 1, 4, 4, 5, 6, 4, 1, 2, 6, 3,
9, 3, 9, 0, 3, 9, 6, 5, 7, 4, 1, 3, 4, 0, 4, 8, 0, 4, 3, 6, 8, 7, 6, 0, 7, 7, 5, 7,
2, 1, 1, 6, 8, 9, 4, 1, 8, 2, 2, 9, 0, 3, 9, 6, 7, 2, 0, 3, 5, 4, 3, 6, 5, 8, 9, 5,
4, 7, 4, 2, 9, 3, 4, 8, 9, 1, 9, 2, 8, 7, 9, 1, 8, 7, 4, 1, 3, 1, 1, 0, 2, 3, 9, 4,
9, 2, 1, 6, 8, 4, 1, 7, 4, 4, 9, 2, 8, 7, 2, 4, 4, 2, 1, 9, 7, 2, 8, 7, 6, 9, 2, 3,
3, 8, 8, 6, 5, 1, 1, 0, 2, 6, 4, 5, 3, 3, 1, 5, 1, 9, 2, 7, 4, 4, 6, 8, 1, 5, 8, 9,
9, 6, 7, 9, 9, 3, 7, 0, 9, 0, 6, 6, 2, 3, 9, 0, 7, 5, 4, 8, 0, 9, 4, 1, 1, 8, 7, 1,
2, 6, 1, 0, 3, 0, 1, 1, 8, 2, 0, 9, 9, 4, 0, 5, 0, 6, 1, 7, 7, 8, 8, 9, 2, 0, 5, 1,
2, 2, 7, 3, 8, 4, 9, 7, 1, 8, 3, 9, 6, 0, 3, 1, 1, 2, 0, 3, 5, 2, 6, 8, 7, 9, 8, 8,
3, 7, 6, 1, 1, 9, 1, 7, 5, 5, 3, 2, 5, 8, 7, 0, 9, 7, 7, 5, 0, 9, 0, 0, 5, 9, 2, 4,
8, 8, 6, 1, 6, 5, 1, 8, 3, 4, 0, 5, 3, 8, 3, 4, 2, 3, 9, 2, 1, 1, 8, 2, 1, 3, 8, 8,
7, 3, 7, 2, 4, 6, 9, 7, 2, 4, 2, 8, 1, 1, 3, 8, 4, 0, 6, 5, 9, 3, 0, 9, 6, 4, 7, 1,
8, 9, 4, 2, 6, 1, 8, 9, 0, 6, 6, 7]
```

Results

In [35]:
```
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)