Neural Network on MNIST dataset

Ödevde ilk olarak sckit-learn kullanılarak MNIST dataseti implemente edildi. Daha sonra bağımlı ve bağımsız değişkenler ayrıldı.

```
from sklearn.datasets import fetch_openml
X, y = fetch_openml('mnist_784', version=1, return_X_y=True)
X = X / 255.
```

Scikit-learn kütüphanesine ait MLPClassifier kullanılarak farklı aktivasyon fonksiyonları, regularization parametreleri, single hidden layer ve birden fazla hidden layer kullanıldığında bu farklılıkların ayrı ayrı nasıl etkilediği incelendi.

Single hidden layer uygulamasında solver ‘adam’ iken ;

```
Dataset 1: Neural net classifier, 1 layer, 1 units
Train score = 0.403029, Test score = 0.395029
, loss = 1.505383
```

Eğer unit 1 ise sonuç bu şekildedir. Loss değeri yüksek train ve test skorlarıda oldukça düşüktür. Bu yüzde burada bir underfit durumu bulunmaktadır.

```
Dataset 1: Neural net classifier, 1 layer, 10 units
Train score = 0.962933, Test score = 0.932457
, loss = 0.131018
```

Eğer unit 10 ise sonuç bu şekildedir. Burada loss değeri oldukça düşük ve train, test skorları oldukça yüksektir. Bu durum ise istenilen bir durumdur.

```
Dataset 1: Neural net classifier, 1 layer, 100 units
Train score = 1.000000, Test score = 0.976286
, loss = 0.001004
```

Eğer unit 100 ise sonuç bu şekildedir. Burada los değeri çok fazla düşük ve test ile train skorlarıda fazla yüksektir. Bu da yine istenilen bir durumdur.

Single hidden layer uygulamasında solver ‘lbfgs’ iken ;

```
Dataset 1: Neural net classifier, 1 layer, 1 units
Train score = 0.112495, Test score = 0.112629, loss = 2.301123
```

Eğer unit 1 ise sonuç bu şekildedir. Loss değeri ‘adam’ ile karşılaştırıldığında daha yüksek çıkmıştır. Aynı şekilde test ve train skorlarıda daha düşüktür. Burada da underfit durumu gözlemlenmektedir.

```
Dataset 1: Neural net classifier, 1 layer, 10 units
Train score = 0.918324, Test score = 0.903200, loss = 0.289801
```

Eğer unit 10 ise sonuç bu şekildedir. Unit 1 de de olduğu gibi ‘adam’ ile karşılaştırıldığında değerler daha kötü gelmiştir. Fakat elde edilen sonuç istenilmeyen bir durum değildir.

```
Dataset 1: Neural net classifier, 1 layer, 100 units
Train score = 1.000000, Test score = 0.974457, loss = 0.000419
```

Eğer unit 100 ise sonuç bu şekildedir. Burada öncekiler gibi ‘adam’ ile karşılaştırıldığında değerler daha kötü çıkmamıştır. Sonuçlar unit 100’de aynı şekilde gelmektedir.

Solver = ‘adam’ olduğunda;

```
Dataset 1: Neural net classifier, 2 layers, 10/10 units
Train score = 0.969276, Test score = 0.933314, loss = 0.110510
```

Solver = ’lbfgs’ olduğunda;

```
Dataset 1: Neural net classifier, 3 layers, 30/30/30 units
Train score = 0.987048, Test score = 0.960457, loss = 0.043491
```

Solver = ‘lbfgs’ olduğunda;

```
Dataset 1: Neural net classifier, 2 layers, 10/10 units
Train score = 0.922686, Test score = 0.909886, loss = 0.266652
```

Solver = ‘adam’ olduğunda;

```
Dataset 1: Neural net classifier, 3 layers, 30/30/30 units
Train score = 0.999448, Test score = 0.962400, loss = 0.004381
```

Yukarıda görülen dört çıktıda hangi durumlarda ne olduğu yazmaktadır. Buradan çıkarılan sonuç şu şekildedir;
Solver = ‘adam’ kullanıldığında elde edilen sonuçlar daha iyidir. Loss daha az ve train ile test skorları daha iyi gelmektedir. Ayrıca hidden layer 3 tane olduğu zaman sonuçlar daha iyi gelmektedir.

Solver = ‘adam’ ve activation = ‘relu’ olduğu durumda;

```
Dataset 2: NN classifier, alpha = 0.010
Train score = 0.967257, Test score = 0.935086, loss = 0.133110

Dataset 2: NN classifier, alpha = 0.100
Train score = 0.955676, Test score = 0.935714, loss = 0.196007

Dataset 2: NN classifier, alpha = 1.000
Train score = 0.935924, Test score = 0.928514, loss = 0.394480

Dataset 2: NN classifier, alpha = 5.000
Train score = 0.894762, Test score = 0.889029, loss = 0.833031
```

Yukarıdaki çıktılarda görüldüğü gibi alpha değeri arttıkça train ve test skorları azalmaktadır. Bu durumda alpha değerinin yüksek olması durumunda underfit durumu gerçekleşebilir. Fakat çıktılardaki sonuçlar kötü değerler olmadığından underfit durumunun olması gözlemlenmemektedir.

Solver = ‘lbfgs’ ve activation = ‘tanh’ olduğu durumda;

```
Dataset 2: NN classifier, alpha = 0.010
Train score = 0.962895, Test score = 0.929657, loss = 0.136123

Dataset 2: NN classifier, alpha = 0.100
Train score = 0.963257, Test score = 0.929257, loss = 0.135026

Dataset 2: NN classifier, alpha = 1.000
Train score = 0.963143, Test score = 0.930914, loss = 0.140907

Dataset 2: NN classifier, alpha = 5.000
Train score = 0.962952, Test score = 0.932514, loss = 0.153408
```

Yukarıdaki çıktıda ise bir öncekiyle karşılaştırıldığında daha iyi sonuçların geldiği görülmektedir. Ayrıca burada alpha değerleri arttıkça loss değeri ve train, test skorlarının nerdeyse yok denecek kadar az miktarda arttığı görülmektedir. Yani bu parametreler kullanıldığında alpha değerinin artmasına rağmen skorlarda bir değişim gözlenmemektedir.

```
Dataset 2: NN classifier, 2 layers 10/10, identity activation function
Train score = 0.939848, Test score = 0.916457, loss = 0.219484

Dataset 2: NN classifier, 2 layers 10/10, logistic activation function
Train score = 0.956133, Test score = 0.930514, loss = 0.162744

Dataset 2: NN classifier, 2 layers 10/10, tanh activation function
Train score = 0.963257, Test score = 0.929257, loss = 0.135026

Dataset 2: NN classifier, 2 layers 10/10, relu activation function
Train score = 0.923771, Test score = 0.910229, loss = 0.264048
```

Yukarıda bulunan şekil farklı aktivasyon fonksiyonlarının etkisini incelemektedir. Burada en kötü sonucu veren aktivasyon fonksiyonu ‘relu’ dur. En iyi train skorunu verip en düşük loss değerini veren fonskiyon tanh’tır.Fakat bu fonksiyon kullanıldığında overfit durumu olma olasılığı diğerlerine göre daha yüksektir. Logistic aktivasyon fonksiyonu ise en iyi test skorunu verdiği gözlenmektedir.

GridSearchCV fonksiyonunu parametreleri otomatik olarak optimize etmek için kullanıldı.

```
parameters = {'solver': ['lbfgs','adam'], 'alpha':[0.001,0.01,0.1,1],
              'hidden_layer_sizes':np.arange(5, 12),}
clf_grid = GridSearchCV(neural_network.MLPClassifier(), parameters,n_jobs=-1)
clf_grid.fit(X, y)
```

GridSearchCV fonksiyonuna yukarıda görülen parametreler verildiğinde ;

```
clf_grid.best_score_
```

```
0.9402857142857143
```

```
clf_grid.best_params_
```

```
{'alpha': 0.1, 'hidden_layer_sizes': 11, 'solver': 'adam'}
```

Bu sonuçlar elde edilmektedir.