CS156 (Introduction to AI), Spring 2022

Homework 1 submission

Roster Name: Your Name

Preferred Name (if different): Chosen Name

Student ID: xxxxxxx

Email address: me@me.com

Any special notes or anything you would like to communicate to me about this homework submission goes in here.

References and sources

List all your references and sources here. This includes all sites/discussion boards/blogs/posts/etc. where you grabbed some code examples.

Solution

Load libraries and set random number generator seed

```
# Citation :
# 1. From Assignment 6
# 2. https://www.geeksforgeeks.org/violinplot-using-seaborn-in-python/ (Flatten, X)
import numpy as np
import pandas as pd
from sklearn import datasets
import matplotlib.pyplot as plt
from PIL import Image
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import plot_confusion_matrix
from sklearn.exceptions import ConvergenceWarning
from sklearn.datasets import load_digits
```

```
from sklearn.model_selection import cross_val_score
import seaborn



np.random.seed(42)
```

Code the solution

```
# Data Loading
data = load_digits()
n_samples = len(data.images)
# Data Normalized and Flattened
X = data.images.reshape((n_samples, -1))
X = X.astype ("float32") / 255
Y = pd.DataFrame(data.target)


# One-Hot encode
Y_ohe = pd.get_dummies(Y, columns = Y.columns, prefix = Y.columns)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y_ohe, test_size=0.2, random_state=0,
X_train.shape, Y_train.shape, X_test.shape, Y_test.shape
```

```
((1437, 64), (1437, 10), (360, 64), (360, 10))
```

```
model1 = MLPClassifier(random_state=1, max_iter=2500).fit(X_train, Y_train)
res1 = cross_val_score(model1, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 1 : ')
print (res1)
print('Mean Cross-Validation Accuracies 1 : ')
print (res1.mean())
model1.score(X_test, Y_test)
model2 = MLPClassifier(hidden_layer_sizes=(400, 150, 50), activation = 'relu', max_iter=2500)
res2 = cross_val_score(model2, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 2 : ')
print (res2)
print('Mean Cross-Validation Accuracies 2 : ')
print (res2.mean())
model2.score(X_test, Y_test)
model3 = MLPClassifier(hidden_layer_sizes=(400, 150, 50), activation = 'logistic', max_iter=2
res3 = cross_val_score(model3, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 3 : ')
print (res3)
print('Mean Cross-Validation Accuracies 3 : ')
print (res3.mean())
model3.score(X_test, Y_test)
model4 = MLPClassifier(hidden_layer_sizes=(64, 32, 8), activation = 'relu', max_iter=2500).fi
res4 = cross_val_score(model4, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 4 : ')
```

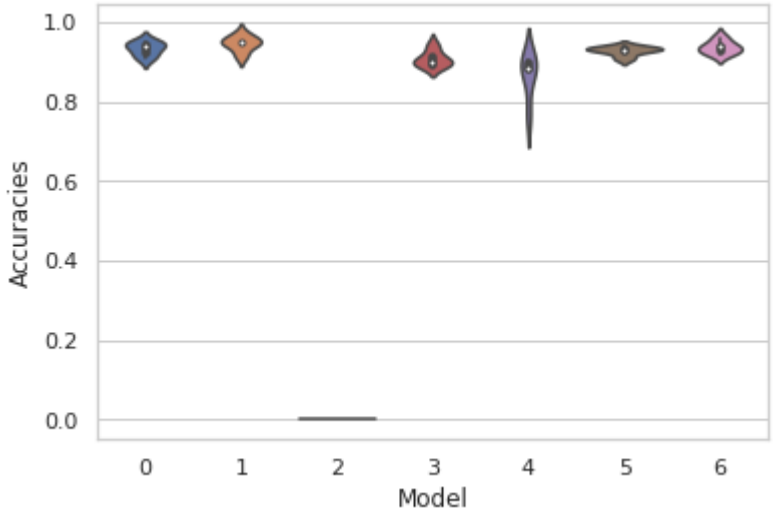
```
print (res4)
print('Mean Cross-Validation Accuracies 4 : ')
print (res4.mean())
model4.score(X_test, Y_test)
model5 = MLPClassifier(hidden_layer_sizes=(32, 16), activation = 'relu', max_iter=2500).fit(X
res5 = cross_val_score(model5, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 5 : ')
print (res5)
print('Mean Cross-Validation Accuracies 5 : ')
print (res5.mean())
model5.score(X_test, Y_test)
model6 = MLPClassifier(hidden_layer_sizes=(120, 64, 16), activation = 'relu', max_iter=2500).
res6 = cross_val_score(model6, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 6 : ')
print (res6)
print('Mean Cross-Validation Accuracies 6 : ')
print (res6.mean())
model6.score(X_test, Y_test)
model7 = MLPClassifier(hidden_layer_sizes=(320, 120, 32), activation = 'relu', max_iter=2500)
res7 = cross_val_score(model7, X_train, Y_train, cv=5, n_jobs=-1)
print('Individual Cross-Validation Accuracies 7 : ')
print (res7)
print('Mean Cross-Validation Accuracies 7 : ')
print (res7.mean())
model7.score(X_test, Y_test)
```

```
Individual Cross-Validation Accuracies 1 :
[0.94097222 0.95138889 0.90940767 0.92334495 0.94425087]
Mean Cross-Validation Accuracies 1 :
0.9338729190863336
Individual Cross-Validation Accuracies 2 :
[0.95138889 0.96875    0.91637631 0.94773519 0.94773519]
Mean Cross-Validation Accuracies 2 :
0.946397115756872
Individual Cross-Validation Accuracies 3 :
[0. 0. 0. 0. 0.]
Mean Cross-Validation Accuracies 3 :
0.0
Individual Cross-Validation Accuracies 4 :
[0.89930556 0.94097222 0.89198606 0.89547038 0.91289199]
Mean Cross-Validation Accuracies 4 :
0.9081252419667052
Individual Cross-Validation Accuracies 5 :
[0.90277778 0.76736111 0.8989547  0.88501742 0.8815331 ]
Mean Cross-Validation Accuracies 5 :
0.8671288230739449
Individual Cross-Validation Accuracies 6 :
[0.93055556 0.93055556 0.93379791 0.93728223 0.90940767]
Mean Cross-Validation Accuracies 6 :
0.928319783197832
Individual Cross-Validation Accuracies 7 :
[0.96180556 0.94097222 0.91986063 0.93728223 0.92682927]
Mean Cross-Validation Accuracies 7 :
```

```
0.9373499806426636
0.9416666666666667
```

```
seaborn.set(style = 'whitegrid')
ax = seaborn.violinplot(data = [res1, res2, res3, res4, res5, res6, res7])
ax.set_ylabel("Accuracies")
ax.set_xlabel("Model")
```

```
Text(0.5, 0, 'Model')
```



✓ 0s completed at 4:54 PM