CS156 (Introduction to AI), Fall 2022

Homework 7 submission

Roster Name: Preet LNU

Student ID: 014755741

Email address: preet.lnu@sjsu.edu

References and sources

https://www.geeksforgeeks.org/violinplot-using-seaborn-in-python/

Solution

Load libraries and set random number generator seed

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier

from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import cross_val_score


np.random.seed(42)
```

Code the solution

Load the Dataset

```
digits = datasets.load_digits()
X = digits.data
X = X.astype("float32") / 255
Y = digits.target
class_names = digits.target_names
X.shape, Y.shape, class_names
```

```
((1797, 64), (1797,), array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]))
```

```
digits_df = pd.DataFrame(X, columns=digits.feature_names)
digits_df['output_digit'] = Y
digits_df.head()
```

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pi
0	0.0	0.0	0.019608	0.050980	0.035294	0.003922	0.0	
1	0.0	0.0	0.000000	0.047059	0.050980	0.019608	0.0	
2	0.0	0.0	0.000000	0.015686	0.058824	0.047059	0.0	
3	0.0	0.0	0.027451	0.058824	0.050980	0.003922	0.0	
4	0.0	0.0	0.000000	0.003922	0.043137	0.000000	0.0	

5 rows × 65 columns

Training and testing data

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

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

6 MLP models

```
model1 = MLPClassifier(random_state=1, max_iter=10000).fit(X_train, Y_train)
model2 = MLPClassifier(hidden_layer_sizes=(400,150,50), activation = 'relu', random_st
model3 = MLPClassifier(hidden_layer_sizes=(64,32,8), activation = 'relu', random_state
model4 = MLPClassifier(hidden_layer_sizes=(32,16), activation = 'relu', random_state=1
model5 = MLPClassifier(hidden_layer_sizes=(120,64,16), activation = 'relu', random_sta
model6 = MLPClassifier(hidden_layer_sizes=(320,120,32), activation = 'relu', random_st
```

Stratified 5-fold cross-val prediction accuracy per fold

```
cross_vals1 = cross_val_score(model1, X_train, Y_train, cv=5)
cross_vals2 = cross_val_score(model2, X_train, Y_train, cv=5)
cross_vals3 = cross_val_score(model3, X_train, Y_train, cv=5)
cross_vals4 = cross_val_score(model4, X_train, Y_train, cv=5)
cross_vals5 = cross_val_score(model5, X_train, Y_train, cv=5)
cross_vals6 = cross_val_score(model6, X_train, Y_train, cv=5)

print('Individual cross-validation accuracies for Model 1: ' + str(cross_vals1))
print('Individual cross-validation accuracies for Model 2: ' + str(cross_vals2))
print('Individual cross-validation accuracies for Model 3: ' + str(cross_vals3))
print('Individual cross-validation accuracies for Model 4: ' + str(cross_vals4))
print('Individual cross-validation accuracies for Model 5: ' + str(cross_vals5))
print('Individual cross-validation accuracies for Model 6: ' + str(cross_vals6))
```

```
Individual cross-validation accuracies for Model 1: [0.96180556 0.98263889 0.958
Individual cross-validation accuracies for Model 2: [0.95486111 0.97569444 0.951
Individual cross-validation accuracies for Model 3: [0.93402778 0.95833333 0.923
Individual cross-validation accuracies for Model 4: [0.9375     0.95486111 0.912
Individual cross-validation accuracies for Model 5: [0.94097222 0.96527778 0.940
Individual cross-validation accuracies for Model 6: [0.94097222 0.97222222 0.944
```

Predicting accuracy

```
model1accuracy = model1.score(X_test, Y_test)
model2accuracy = model2.score(X_test, Y_test)
model3accuracy = model3.score(X_test, Y_test)
model4accuracy = model4.score(X_test, Y_test)
model5accuracy = model5.score(X_test, Y_test)
model6accuracy = model6.score(X_test, Y_test)

print('Accuracy of MLPClassifier Model1 on test set: {:.2f}'.format(model1accuracy))
print('Accuracy of MLPClassifier Model2 on test set: {:.2f}'.format(model2accuracy))
print('Accuracy of MLPClassifier Model3 on test set: {:.2f}'.format(model3accuracy))
print('Accuracy of MLPClassifier Model4 on test set: {:.2f}'.format(model4accuracy))
print('Accuracy of MLPClassifier Model5 on test set: {:.2f}'.format(model5accuracy))
print('Accuracy of MLPClassifier Model6 on test set: {:.2f}'.format(model6accuracy))
```

```
Accuracy of MLPClassifier Model1 on test set: 0.97
Accuracy of MLPClassifier Model2 on test set: 0.95
Accuracy of MLPClassifier Model3 on test set: 0.94
Accuracy of MLPClassifier Model4 on test set: 0.93
Accuracy of MLPClassifier Model5 on test set: 0.94
Accuracy of MLPClassifier Model6 on test set: 0.95
```

Setting things up for plotting

```
models = ['Model1', 'Model1', 'Model1', 'Model1', 'Model1',
          'Model2', 'Model2', 'Model2', 'Model2', 'Model2',
          'Model3', 'Model3', 'Model3', 'Model3', 'Model3',
          'Model4', 'Model4', 'Model4', 'Model4', 'Model4',
          'Model5', 'Model5', 'Model5', 'Model5', 'Model5',
          'Model6', 'Model6', 'Model6', 'Model6', 'Model6']
models_for_score = ['Model1', 'Model2', 'Model3', 'Model4', 'Model5', 'Model6']
modelnumber = [1, 2, 3, 4, 5, 6]
model_accs = [model1accuracy, model2accuracy, model3accuracy, model4accuracy, model5ac
vals = np.array([cross_vals1, cross_vals2, cross_vals3, cross_vals4, cross_vals5, cros
vals = vals.flatten()
```

Plotting the cross vals and test set accuracy

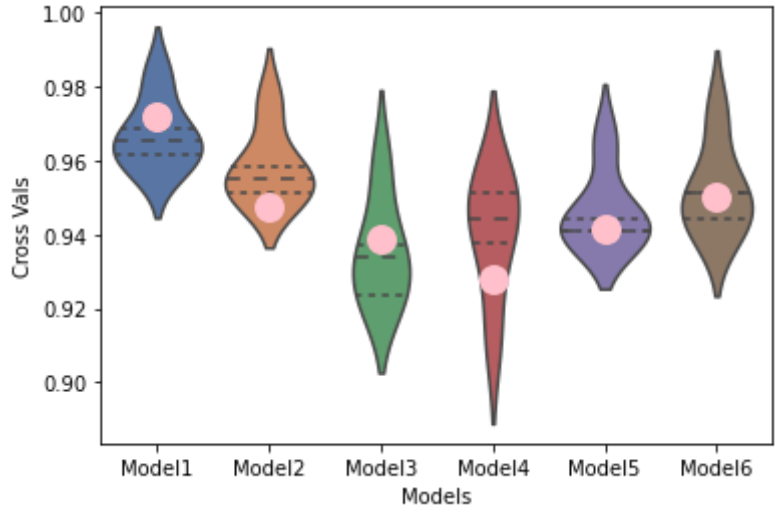
```
fig, ax = plt.subplots()

sns.set(style = 'whitegrid')
vplot = sns.violinplot(x = models, y = vals, inner = "quartile", ax = ax)
vplot.set_xlabel("Models")
vplot.set_ylabel("Cross Vals")

vplot = sns.swarmplot(x = models_for_score, y = model_accs, color = 'pink', edgecolor

print(vplot)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```



Colab paid products - Cancel contracts here

✓ 3m 21s completed at 5:02 PM