CS156 (Introduction to AI), Spring 2021

Homework 6 submission

Roster Name: Neeval Kumar

Preferred Name (if different): Chosen Name

Student ID: 011877086

Email address: kumar.neeval@gmail.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

In [131…
```
import numpy as np
import pandas as pd
from sklearn import datasets
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Perceptron
from sklearn.metrics import plot_confusion_matrix
```

In [132…
```
np.random.seed(42)
```

Load the dataset and flatten numbers

In [133…
```
digits = datasets.load_digits()

X = digits.data
Y = digits.target
class_names = digits.target_names
X.shape, Y.shape, class_names
```

Out[133…
```
((1797, 64), (1797,), array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]))
```

In [134…
```
X = X.astype("float32") / 255
```

Just a method to replace the variables in Y based on the class label we are looping through. Basically the 0 or not 0 aspect in our confusion matrix. If it is the label, then we replace the element with 1, otherwise 0.

In [135…
```
def replaceLabels(Y, label):
    for index, value in enumerate(Y):
        if value == label:
            Y[index] = 1
        else:
            Y[index] = 0
    return Y
```

Loop through each unique class label, and replace the Y with the labels from the method before. Then split and create a model and perform a confusion matrix on that specific model.

In [136…
```
digits_df = pd.DataFrame(X,columns=digits.feature_names)
for index, label in enumerate(class_names):
    scaled_Y = replaceLabels(list(Y),label)
    X_train, X_test, Y_train, Y_test  = train_test_split(digits_df, scaled_Y, test_si

    model = Perceptron(tol=1e-3, random_state=0)
    model.fit(X_train, Y_train)

    title = 'Accuracy of perceptron on training set: {:.2f}'.format(model.score(X_tra
    display_labels = ['Not ' + str(label), str(label)]
    disp = plot_confusion_matrix(model, X_test, Y_test,
                                 display_labels= display_labels,
                                 cmap=plt.cm.Blues)
    disp.ax_.set_title(title)
plt.show()
```