CA Assignment 1
Data Classification
Implementing Perceptron algorithm

ASSIGNMENT REPORT SUBMITTED FOR
ASSSESMENT OF COMP527
(DATA MINING AND VISUALISATION)

BY
SANTHOSH ARUNAGIRI

UNIVERSITY OF
LIVERPOOL

DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF LIVERPOOL
LIVERPOOL L69 3BX

PERCEPTRON ALGORITHM

The perceptron algorithm is a simple binary classification algorithm. The algorithm tries to learn a way to draw a straight line between the two groups that separates them well as possible. Given a set of training examples, each consisting of a feature vector and a binary class label (+1 or -1), the algorithm learns a weight vector that defines the decision boundary, with positive weights pushing the decision boundary towards positive examples and negative weights pushing it towards negative examples.

PERCEPTRON PSEUDO CODE

Training

1) *Initialize weights and bias to zero.*

```
weights = [0] * numfeatures
bias = 0
```

2) *Train the perceptron for a fixed number of iterations*

```
for iterations in range(n_iterations):
```

3) *Loop over all training data*

```
for x, y in training examples:
```

4) *Calculate predicted output*

```
Predicted output = sum (weights[i] * x[i] for i in range(numfeatures)) + bias
```

5) *Make prediction based on sign of output.*

```
y_pred = 1 if predicted output >= 0 else -1
```

6) *Update weights and bias if prediction is incorrect.*

```
if y_pred != y:
for i in range(numfeatures):
weights[i] += learning rate * y * x[i]
bias += learning rate * y
```

Testing

1) *Compute the predicted label for a new DATA*

```
y_pred = sign(dot product(weights, newfeatures))
```

BINARY CLASSIFICATION RESULTS

The train and test classification accuracies for each of the three classifiers after

training for 20 iterations.

Coeff = 0, Learning rate = 1, Iterations = 20

	TRAIN ACCURACY (%)	TEST ACCURACY (%)
CLASS-1 VS CLASS-2	99.38	100
CLASS-2 VS CLASS-3	90.68	92.5
CLASS-1 VS CLASS-3	94.38	89.74

Pair of classes is most difficult to separate are *CLASS-1 VS CLASS-3*

1-VS-REST APPROACH

The 1-vs-rest approach is a technique used in multi classification to classify more than two labels. In this technique, we train a binary classifier for each class, treating one class as positive and the rest as negatives. During training, we train the binary classifier for each class separately and create a decision boundary that separates the instances of that class from the instances of all other classes. This results in n binary classifiers, where n is the number of classes in the dataset.

Once the training is complete, to classify a new instance, we pass the values through each binary classifier, and the classifier with the maximum value (i.e., the highest confidence that the instance belongs to the positive class) is chosen as the predicted class for the instance.

The 1-vs-rest approach is commonly used when we have a greater number of classes or when the classes are imbalanced.

With respect to the code in .py,

The code splits the data into features and target variables for both training and testing sets.

Data is converted to NumPy arrays and the target variable is converted to numerical values using a dictionary that maps class labels to numerical values.

Next, a one vs rest perceptron model is instantiated with desired hyperparameters. The learning rate is set to 1, iterations is set to 20, and regularization coefficient is set to required value (0, 0.01, 0.1, 1.0, 10.0, 100.0)

The perceptron is then fitted to the training data using the fit() method.

three models are created as each models is based like one class is made positive and the rest are negative and alternatively for all three class. Each i in the data is trained for all three models to get three different z values. Comparing all the z values and finding the maximum is done by argmax. It is then compared with the y to get accuracy.

The performance of the model is then evaluated using the np.mean() function to

calculate the accuracy. The accuracy is printed for each pair of classes.

ONE VS REST CLASSIFICATION RESULTS

Coeff = 0, Learning rate = 1, Iterations = 20

	TRAIN ACCURACY (%)	TEST ACCURACY (%)
ONE VS REST	33.38	33.3

After setting coefficients as given values,

Learning rate = 1, Iterations = 20

COEFF	TRAIN ACCURACY (%)	TEST ACCURACY (%)
0.01	33.61	33.9
0.1	33.61	33.9
1	33.2	33.47
10	33.2	32.2
100	33.2	32.2

CONCLUSION

After comparing all the accuracies in one vs rest classification technique with different L2 coefficients (0, 0.01, 0.1, 1.0, 10.0, and 100.0), we can conclude that the model is not performing well on this dataset. The accuracy of the model is around 33%, i.e.. it is not able to classify the data correctly. Moreover, increasing the L2 coefficient does not seem to improve the performance of the model. Therefore, we need to explore other algorithms or techniques to improve the accuracy of the model.

REFERENCES

1) Raschka, S. (2015). Python Machine Learning. [online] Google Books. Packt Publishing Ltd. Available at: https://books.google.co.uk/books?id=GOVOCwAAQBAJ&lpg=PP1&ots=NdeCPcXS_F&dq=perceptron%20algorithm%20python%20&lr&pg=PA365#v=onepage&q&f=false [Accessed 3 Mar. 2023].
2) Aggarwal, C.C. (2018). Neural Networks and Deep Learning. [online] Cham: Springer International Publishing. doi: https://doi.org/10.1007/978-3-319-94463-0.
3) Brownlee, J. (2016). How To Implement The Perceptron Algorithm From Scratch In Python. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/implement-perceptron-algorithm-scratch-python/.