Assignment 6 (Sol.)

Introduction to Data Analytics

Prof. Nandan Sudarsanam & Prof. B. Ravindran

1. Is it possible to use neural networks to perform compression? If so, will the compression be lossy (i.e, exact input cannot be recovered) or lossless (decompression gives back the exact input)?

   (a) no, compression is not possible using neural networks

   (b) yes, compression is possible, but only lossy compression

   (c) yes, compression is possible, but only lossless compression

   (d) yes, compression is possible, and depending upon the data and the network, both lossy and lossless compression may be performed

   Sol. (d)

2. Assume that you are given a data set and a neural network model trained on the data set. You are asked to build a decision tree model with the sole purpose of understanding/interpreting the built neural network model. In such a scenario, which among the following measures would you concentrate most on optimising?

   (a) Accuracy of the decision tree model on the given data set

   (b) F1 measure of the decision tree model on the given data set

   (c) Fidelity of the decision tree model, which is the fraction of instances on which the neural network and the decision tree give the same output

   (d) Comprehensibility of the decision tree model, measured in terms of the size of the corresponding rule set

   Sol. (c)

3. Which of the following is/are true about bagging?

   (a) bagging reduces variance of the classifier

   (b) bagging increases the variance of the classifier

   (c) bagging can help make robust classifiers from unstable classifiers

   (d) majority is one way of combining outputs from various classifiers which are being bagged

   Sol. (a), (c), (d)

4. Can the boosting technique be applied on regression problems? Can bagging be applied on regression problems?

   (a) no, no
   (b) no, yes
   (c) yes, no
   (d) yes, yes

   Sol. (d)
   Ensemble methods are not tied to the classification problem, and can be used for regression as well.

5. In the general context of classification, re-weighting the data points (relative to an original training data set where the points are un-weighted) can lead to

   (a) change in the underlying optimisation problem that is solved
   (b) change in the positions of data points in the feature space
   (c) change in the decision surface generated by the classifier
   (d) change in the nature of the data set from being linearly separable to becoming linearly non-separable (in case the original data was linearly separable)

   Sol. (a), (c)
   Consider the case of support vector machines. Suppose a specific data point was re-weighted so that its new weight is 2 (originally all data points were un-weighted, i.e., had weight of 1). This is as good as there being two instances of the same data point, and assuming that this data point falls on the wrong side of the separating hyperplane, the error contributed by this point doubles. Hence, the underlying optimisation problem changes, with their being more of an emphasis to reduce the error on this specific point. Solving this modified optimisation problem may lead to a change in the separating hyperplane (compared to the one for the original data set). Note however, that re-weighting does not result in any change in the positions of the points in the feature space, hence if the original training data was linearly separable, the re-weighted data set remains linearly separable.

6. If one feature (compared to all others) is a very strong predictor of the class label of the output variable, then all of the trees in a random forest will have this feature as the root node.

   (a) false
   (b) true

   Sol. (a)
   In random forests, due to the random subset of features selected at each split, while constructing individual trees, many of the trees will not have this feature as the root even if it is the most important feature for predicting the label of the target variable.

7. Which of the following statements are true about ensemble classifiers?

   (a) The different learners in boosting based ensembles can be trained in parallel
   (b) The different learners in bagging based ensembles can be trained in parallel

(c) Boosting based algorithms which iteratively re-weight training points, such as AdaBoost, are more sensitive to noise than bagging based methods.
(d) Boosting methods generally use strong learners as individual classifiers
(e) Boosting methods generally use weak learners as individual classifiers.
(f) An individual classifier in a bagging based ensemble is trained with every point in the training set
(g) An individual classifier in a boosting based ensemble is trained with every point in the training set.

Sol. (b), (c), (e), (g)

8. By using a linear activation function in the output layer of a neural network for solving regression tasks, are we constraining the resultant model to be linear in the input features?

(a) no
(b) yes

Sol. (a)
Note that in this case, the linear functions in the output layer still take as input non-linear outputs of the hidden layer units.

9. In case of limited training data, which technique, bagging or stacking, would be preferred, and why?

(a) bagging, because we can combine as many classifier as we want by training each on a different sample of the training data
(b) bagging, because we use the same classification algorithms on all samples of the training data
(c) stacking, because each classifier is trained on all of the available data
(d) stacking, because we can use different classification algorithms on the training data

Sol. (c)

10. In the lectures, we saw how to train a 7 layer auto encoder network. In case we wanted to perform classification on the data used for training this network, while making use of the trained network, a suitable approach would be to

(a) add an additional eighth layer on top of the 7 layers as the output layer and train the entire network for the classification task
(b) add an additional eighth layer on top of the 7 layers as the output layer and only modify the weights between layers 7 and 8 in training for the classification task
(c) discard the top 3 layers, add an additional layer on top of the 4^{th} layer as the output layer and train the entire network for the classification task
(d) discard the top 3 layers, add an additional layer on top of the 4^{th} layer as the output layer and only modify the weights between layers 4 and 5 in training for the classification task

Sol. (c)
Counting from the bottom, the fourth layer of the network contains the compact representation of the input data that we used the greedy unsupervised layerwise pretraining technique to obtain. Post finetuning, we can discard the top 3 layers, and add an additional layer on top of layer 4 which will act as our output layer. Next we train the network as we would normally do using backpropogation and modifying the weights of all the layers.

Note that just modifying the weights between layers 4 and 5 may not allow us to obtain a classifier with satisfactory performance. This is because up till this stage, the previous layers have been trained only for the auto-encoder task, and may need to be modified further to allow for good performance in the classification task.