CS 6140 Machine Learning - Assignment 4

(200 points)

1 Perceptron and Dual Perceptron

1.1 Data and Pre-processing

- PerceptronData: This is a binary classification dataset consisting of four features and the classes are linearly separable.
- Two Spirals: This dataset has two features and it is non-linearly separable (Figure-1).

1.2 Implementation(*30 points*)

1. Implement the perceptron algorithm and test it on the PerceptronData dataset using ten fold cross validation.
2. Implement the dual perceptron algorithm and test it on the PerceptronData dataset using ten fold cross validation.

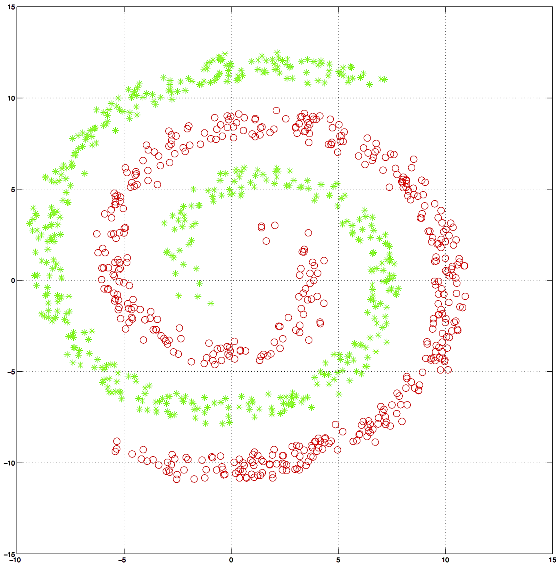

Figure 1: The two spirals dataset, the colors represent class labels.

3. Compare the performance of the two algorithms on the PerceptronData dataset and make sure that they have (almost) identical performance.

1.3 Kernelizing Dual Perceptron(*20 points*)

1. Run the dual perceptron with the linear kernel on the Two Spiral dataset and show that the data is not separable using ten-fold cross validation.
2. Run the dual perceptron with the Gaussian (RBF) kernel on the Two Spiral dataset and show that the data is separable using ten-fold cross validation.

Note: You would need to select an appropriate value for the scale γ of the RBF kernel. Design a search strategy to set the value of γ that searches for possible values of $\gamma \in [0, 0.25]$.) For both the perceptron and dual perceptron be sure to set an upper limit on the number of iterations.

2 Regularized Logistic Regression(*50 points*)

In this question you will develop a regularized version of Logistic regression. Let (x_i, y_i), $i \in \{1, 2, \ldots, N\}$ represent the training data where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. Under the logistic regression model, the probability of an instance belonging to either class is given as:

$$P(y_i|w, x_i) = \frac{1}{1 + e^{-y_i w^T x_i}} \tag{1}$$

where, w are the model parameters. The maximum likelihood solution can be obtained by minimizing the negative log-likelihood. For the regularized case, we add an L2 penalty on the norm of w which results in the objective function for regularized Logistic regression:

$$\min_{w} \quad \frac{\lambda}{2}\|w\|^2 + \sum_{i=1}^{N} \ln(1 + e^{-y_i w^T x_i}) \tag{2}$$

1. Do you think that w_0 should be included in the regularization?
2. Calculate the gradient of the objective with respect to the model parameters.
3. Develop a gradient descent algorithm for learning the parameters from given training data.
4. Contrast the performance of Logistic Regression with Regularized Logistic Regression for the Spambase, Diabetes and Breast Cancer datasets using ten fold cross validation.
5. Is it possible to kernelize Equation-2? If yes, then what would be the dual objective function for regularized logistic regression?

Note: If you google Regularized Logistic Regression, Andrew Ng has a video lecture that details the development of Regularized Logistic Regression from simple Logistic Regression. I would recommend you to watch the video after you have spent some time trying to develop the solution yourself.

3 Determining Model Hyper-parameters(*60 points*)

In this section we will look at a strategy for setting model hyper-parameters. An example of a model hyper-parameter is λ in Equation 2 that we need to set before we can train and assess the performance of logistic regression on a given dataset. Most of the models we have worked with require only one hyper-parameter and therefore we test different values of the parameter over a fixed range. This is known as grid search, beacuse we are gridding the possible values of a parameter and then selecting a value that gives us the best peformance on our validation set. But, what if our model requires multiple hyper-parameters to be set prior to training. In this case, we cannot treat each parameter as independent and need to try different combinations of all hyper-parameters.

Usually, we use k-fold cross-validation to assess the performance of our model. But when we also have to determine model hyper-parameters, we need to select the best values using only our training data, and not include the test data while searching for the hyper-parameters as this will cause "leakage" where the training phase has information about the test data which will bias our results (the results will be optimistic).

In order to facilitate this, we use a nested cross-validation strategy. In this strategy we will partition our training data (determined through the outer k-fold cross-validation loop) further into m-folds (most commonly used values are: $k = 10, m = 5$). For each iteration of the inner cross-validation we train on $m - 1$ folds, and try different combinations of model hyper-parameters, and select the combination that gives us the highest performance i.e., recall, precision or accuracy (ties can be broken randomly) on the internal test set. We will then use this combination to train a model using all m folds (original training data produced by the outer cross-validation loop), and test its performance on the outer test set. Since, we have m-fold cross-validation nested within a k-fold cross-validation, therefore we call this strategy nested cross-validation.

In this question you will be working with libSVM or the SVM classifier in sklearn. You will be using the linear and RBF kernels, and will need to set two model-hyperparamters (for the RBF kernel) i.e., γ for the RBF kernel, and C the cost of misclassification for the SVM. Determine a good range for both hyperparamters, usually these are set as powers of two i.e., $C \in \{2^{-5}, 2^{-4}, ..., 2^{10}\}$ and $\gamma \in \{2^{-15}, 2^{-14}, ..., 2^{5}\}$, you can adjust these ranges as you think appropriate for the given datasets.

3.1 Deliverables:

1. Report the training and test perfomance of both kernels on the Diabetes, Breast Cancer, and Spambase datasets in a table (include mean and standard deviation of recall, precision, and accuracy). List the best hyper-parameters that you found by optimizing the accuracy over your parameter grid, for each fold.

2. Repeat the grid search in the first part but instead of optimizing accuracy, optimize AUC. *Note:* In order calculate AUC will need the SVM to predict class probabilities, instead of predicting labels.

3. For each dataset, provide a ROC-AUC curve across all k-folds.

4 SVMs vs Multiclass Problems(*40 points*)

Since SVMs are binary classifiers, how do they deal with multiclass dataset? We will design a 1-vs-all strategy where we train m models for an m-class multiclass problem. In this strategy if we have three classes we will train three models: class-1 (positive) vs all other classes (negative), class-2 (positive) vs all other classes (negative), and class-3 (positive) vs all other classes (negative). Each of these three problems will be solved independently.

4.1 Deliverables:

1. Report the training and test perfomance of both kernels on the Wine dataset in a table (include mean and standard deviation of recall, precision, and accuracy per class). List the best hyper-parameters that you found by optimizing the accuracy over your parameter grid, for each fold per class.

2. Provide a ROC-AUC curve for each class, across all k-folds.

3. *Extra-credit:* Download the digits dataset which is an OCR dataset i.e., recognition of handwritten digits from USPS. It has 256 features corresponding to pixels from a 16 x 16 image of digits. Use a linear and an RBF kernel based SVM model, and report the recall, precision and accuracy per class using cross-validation to set hyper-parameters. Provide ROC-AUC curves for every class separtely. (*Note:* the dataset is pre-partitioned into a training and a test set, so you don't have to do nested cross-validation (the outer cross-validation loop).)