AI6012: Machine Learning Methodologies & Applications Assignment

M.Sc. Artificial Intelligence

Adnan Azmat (G2303265K)

Question 1

Consider a multi-class classification problem of C classes. Based on the parametric forms of the conditional probabilities of each class introduced on the 39th Page ("Extension to Multiple Classes") of the lecture notes of L4, derive the learning procedure of regularized logistic regression for multi-class classification problems. Hint: define a loss function by borrowing an idea from binary classification, and derive the gradient descent rules to update w_c's.

Solution

In the multi-class classification problem of C classes, the parametric forms of the conditional probabilities of each class is given by the term:

$$P(y = c|x) = \frac{\exp(w^{(c)T}x)}{1 + \sum_{c=1}^{C-1} \exp(w^{(c)T}x)} \tag{1}$$

$$P(y = 0|x) = \frac{1}{1 + \sum_{c=1}^{C-1} \exp(w^{(c)T}x)} \tag{2}$$

where -

- $P(y = c|x)$: Conditional probability where the variable y takes the value c given x.
- c: Specific class or category in a classification problem.
- x: Input vector.
- $w^{(c)}$: Weight vector corresponding to class c.
- C: Total number of classes or categories.

This equation is obtained by assuming C-1 independent binary logistic regression models and choosing one class (zeroth in this case) as reference or pivot. Thus, the probability of the reference class can be calculated as:

$$P(y_i = 0) = 1 - \sum_{c=1}^{C-1} P(y_i = c)$$

The probability of the rest of the classes can be obtained using the logit of the probability p

$$\ln \frac{P(y_i = c)}{P(y_i = 0)} = w_c \cdot x_i \quad , 0 < c < C$$

$$\implies P(y_i = c) = P(y_i = 0)e^{w_c \cdot x_i}$$

$$\implies P(y_i = 0) = 1 - \sum_{c=1}^{C-1} P(y_i = c) = 1 - \sum_{c=1}^{C-1} P(y_i = 0)e^{w_c \cdot x_i}$$

$$\implies P(y_i = 0) = \frac{1}{1 + \sum_{c=1}^{C-1} e^{w_c \cdot x_i}}$$

$$\implies P(y_i = c) = \frac{e^{w_c \cdot x_i}}{1 + \sum_{c=1}^{C-1} e^{w_c \cdot x_i}} \quad , o < c < C$$

Once we have the probability of each class, the likelihood function can be written as:

$$L(w) = \prod_{i=1}^{N} \prod_{c=1}^{C} P(y_i = c|x_i; w)^{I\{y_i = c\}} \tag{3}$$

- $L(w)$: This is the likelihood function which measures the goodness of fit of a statistical model to a sample of data for given values of the unknown parameters.
- w: These are the parameters of the model. The goal is to find the values of these parameters that maximize the likelihood function.
- x_i: Data points or features.
- y_i: Class labels for each observation.
- N: total number of observations in the dataset.
- C: Total number of classes or categories.
- $P(y_i = c|x_i; w)$: Conditional probability of class c given observation x_i and parameters w. We model the probability that a given observation belongs to a certain class.
- $I\{y_i = c\}$: This is an indicator function that equals 1 if y_i equals c and 0 otherwise.

We use negative log likelihood as loss function for our multi-class logistic regression model:

$$J(w) = -\log L(w) = -\sum_{i=1}^{N} \sum_{c=1}^{C} I\{y_i = c\} \log P(y_i = c|x_i; w) \tag{4}$$

If we add a regularization term, we get:

$$J(w) = -\log L(w) = -\sum_{i=1}^{N} \sum_{c=1}^{C} I\{y_i = c\} \log P(y_i = c|x_i; w) + \frac{\lambda}{2} \sum_{c=1}^{C} \|w_c\|^2 \tag{5}$$

$$\implies J(w) = -\log L(w) = -\sum_{i=1}^{N} \sum_{c=1}^{C} I\{y_i = c\} \left(\frac{e^{w_c \cdot x_i}}{1 + \sum_{c=1}^{C-1} e^{w_c \cdot x_i}} \right) + \frac{\lambda}{2} \sum_{c=1}^{C} \|w_c\|^2$$

As part of the learning procedure, we need to minimise the cost function $J(w)$ by arriving at an optimal w. Since, this cost function is not convex, we can use the gradient descent learning approach. For gradient descent learning we first need to calculate the derivative of the cost function.

$$\implies \frac{\partial J(w)}{\partial w_c} = -\sum_{i=1}^{N} I\left\{y_i = c\right\} \frac{\partial}{\partial w_c}\big(\log P(y_i = c|x_i; w)\big) + \lambda w_c$$

$$\implies \frac{\partial J(w)}{\partial w_c} = -\sum_{i=1}^{N} I\left\{y_i = c\right\} \left(\frac{1}{P(y_i = c|x_i; w)}\right) \frac{\partial}{\partial w_c}\left(\frac{e^{w_c \cdot x_i}}{1 + \sum_{c=1}^{C} e^{w_c \cdot x_i}}\right) + \lambda w_c \tag{6}$$

Let's denote $f(w_c) = e^{w_c x_i}$ and $g(w_c) = 1 + \sum_{c=1}^{C} e^{w_c x_i}$.

Then, we have: $f'(w_c) = x_i e^{w_c x_i}$ and $g'(w_c) = x_i \sum_{c=1}^{C} e^{w_c x_i}$

Using quotient-rule we have,

$$\frac{\partial}{\partial w_c}\left(\frac{e^{w_c \cdot x_i}}{1 + \sum_{c=1}^{C} e^{w_c \cdot x_i}}\right) = \frac{x_i e^{w_c x_i}(1 + \sum_{c=1}^{C-1} e^{w_c x_i}) - e^{w_c x_i} x_i \sum_{c=1}^{C} e^{w_c x_i}}{(1 + \sum_{c=1}^{C} e^{w_c x_i})^2}$$

$$= x_i(P(y_i = c|x_i; w))(1 - P(y_i = c|x_i; w))$$

Substituting this value in (6), we have:

$$\implies \frac{\partial J(w)}{\partial w_c} = -\sum_{i=1}^{N} I\left\{y_i = c\right\} \left(\frac{1}{P(y_i = c|x_i; w)}\right) x_i(P(y_i = c|x_i; w))(1 - P(y_i = c|x_i; w)) + \lambda w_c$$

$$\implies \frac{\partial J(w)}{\partial w_c} = -\sum_{i=1}^{N} I\left\{y_i = c\right\} x_i(1 - P(y_i = c|x_i; w)) + \lambda w_c \tag{7}$$

To minimize the loss function and learn the weight vectors w_c for each class, we'll use gradient descent. The update rule for each weight vector w_c is given by:

$$w_c = w_c - \alpha \frac{\partial J(w)}{\partial w_c} \tag{8}$$

OR

$$w_c = w_c - \alpha\left(\sum_{i=1}^{N} I\{y_i = c\} x_i(P(y_i = c) - 1) + \lambda w_c\right) \tag{9}$$

where:

α is a hyper parameter called the learning rate

λ is the regularization parameter used to control over fitting

$I\{y_i = c\}$ is the indicator function that equals 1 if y_i equals c and 0 otherwise.

and $P(y_i = c)$ is equal to $\dfrac{e^{w_c \cdot x_i}}{1 + \sum_{c=1}^{C} e^{w_c \cdot x_i}}$

This is the learning procedure of regularized logistic regression for multi-class classification problem using the gradient descent algorithm.

Question 2

This is a hands-on exercise to use the SVC API of scikit learn to train a SVM with the linear kernel and the rbf kernel, respectively, on a binary classification dataset.

1. Download the a9a dataset from the LIBSVM Dataset page.

2. Regarding linear kernel, show 3-fold cross-validation results in terms of classification accuracy on the training set with different values of parameter C in {0.01, 0.05, 0.1, 0.5, 1}

3. Regarding rbf kernel, show 3-fold cross-validation results in terms of classification accuracy on the training set with different values of the parameter gamma in {0.01, 0.05, 0.1, 0.5, 1} and different values of the parameter C in {0.01, 0.05, 0.1, 0.5, 1}

4. Based on the results shown, determine the best kernel and the best parameter setting. Use the best kernel with the best parameter setting to train a SVM using the whole training set and make predictions on test set.

Solution

1. We can use the *sklearn.datasets*'s *load_svmlight_file()* to load the a9a dataset. *load_svmlight_file* is a function in the *scikit-learn* library that loads a dataset in the svmlight/LIBSVM format and returns it as a sparse matrix.

The a9a dataset is a binary classification dataset containing demographic and employment-related features, used to predict whether a person's income is greater than $50,000 per year.

In this LIBSVM format, each line represents a single data point, and the first value on the line is the label (either +1 or -1). The remaining values on the line are feature-value pairs, where the feature is an integer index and the value is a floating-point number. Features that are not present in a data point are assumed to have a value of zero.

Example line (representing a single data point): -1 3:1 11:1 14:1 19:1 39:1 42:1 55:1 64:1 67:1 73:1 75:1 76:1 80:1 83:1

2.

C	0.01	0.05	0.1	0.5	1
Accuracy	0.844015	0.846104	0.846442	0.846933	0.847209

Table 1: 3-fold cross-validation mean accuracy for different values of the parameter C using linear kernel (Correct to 6 decimal spaces)

3.

C	**g = 0.01**	**g = 0.05**	**g = 0.1**	**g = 0.5**	**g = 1**
0.01	0.759190	0.819907	0.819846	0.759190	0.759190
0.05	0.831209	0.835755	0.834250	0.789165	0.759190
0.1	0.837720	0.839655	0.838764	0.806118	0.761985
0.5	0.842972	0.845766	0.846811	0.832161	0.789748
1	0.844415	0.846749	0.847425	0.836614	0.798286

Table 2: 3-fold cross-validation mean accuracy for different values of the parameter C and gamma (g) using Radial Basis Function (rbf) kernel (Correct to 6 decimal spaces)

4. Looking at the two tables (using linear and rbf kernels), we can observe that using the **rbf kernel** with parameters **C = 1** and **gamma (g) = 0.1**, performed the best with the 3-fold cross-validation mean accuracy of 0.847425

	rbf kernel (C = 1, g = 0.1)
Accuracy of SVM	0.850316

This accuracy was achieved by using the best kernel with the best parameter setting (C = 1, g = 0.1) to train a SVM on the whole training set and make predictions on test set of the a9a dataset.

Question 3

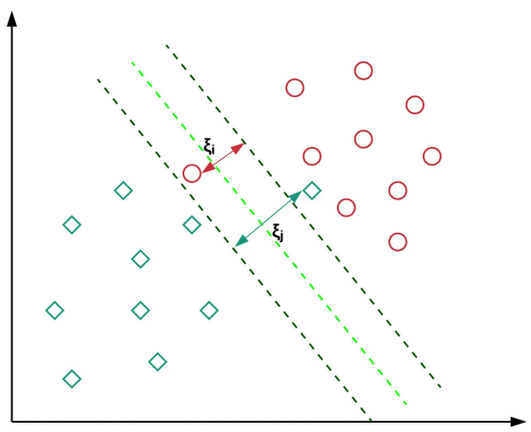
The optimization problem of linear soft-margin SVMs can be re-formulated as an instance of empirical structural risk minimization (refer to Page 37 on L5 notes). Show how to reformulate it. Hint: search reference about the hinge loss.

Solution

Soft margin SVM allows some mis-classification of data points in order to find a better separating hyperplane. This is done by introducing slack variables, which allow data points to fall within or on the wrong side of the margin, but with a penalty. The soft margin SVM finds the hyperplane that maximizes the margin while minimizing the penalty for misclassification.

Soft margin SVM is more robust than hard margin SVM, as it can handle data that is not perfectly linearly separable. It is also less prone to overfitting, as it is not trying to perfectly fit every data point.

Figure 1: Slack variables ϵ_i (when classes cannot be perfectly separated)



We add a penalty term in-corporating the slack variable ξ_i

$$\min_{w,b,\xi_i} \frac{\|w\|_2^2}{2} + C\sum_{i=1}^{N} \xi_i$$

$$s.t\ y_i[w^T\phi(x_i) + b] \geq 1 - \xi_i\ \forall i \geq 0$$

$$\xi_i \geq 0$$

C is a tradeoff parameter to balance the impact of margin maximization & tolerable errors This equation can be reformula