Machine Learning - Lab

Iva Jorgusheska, UID: 11114620

April 30, 2024

1 Linear Classification via Gradient Descent

1.1 Step-by-step derivation of the gradient for the objective function

1. **Derivative of the Hinge Loss Term with respect to:**

$$w^T x_i + w_0$$

$$L_i = \max(0, 1 - y_i(w^T x_i + w_0))$$

$$\frac{\partial L_i}{\partial (w^T x_i + w_0)} = \begin{cases} -y_i & \text{if } 1 - y_i(w^T x_i + w_0) > 0 \\ 0 & \text{otherwise} \end{cases}$$

If the prediction is good, it should not affect the change, hence we have 0 in that case.

2. **Derivative of the Regularization Term:**

$$\frac{\partial}{\partial w}\left(\frac{1}{2} w^T w\right) = w$$

3. **Combine the Derivatives to with respect to w:**

$$\frac{\partial w}{\partial O} = \sum_{i=1}^{N} \frac{\partial (w^T x_i + w_0)}{\partial L_i} \cdot \frac{\partial w}{\partial (w^T x_i + w_0)} + w$$

$$\frac{\partial w}{\partial O} = \sum_{i=1}^{N} (-y_i) x_i \cdot 1_{(1 - y_i(w^T x_i + w_0) > 0)} + w$$

4. **Vectorized Form:**

$$\frac{\partial O}{\partial w} = -C \left(\sum_{i=1}^{N} y_i x_i \cdot 1_{(y_i(w^T x_i + w_0) < 1)} \right) + w$$

Result:
The gradient $\nabla O(w)$ of the function $O = C \sum_{i=1}^{N} \max(0, 1 - y_i(w^T x_i + w_0)) + \frac{1}{2} w^T w$ with respect to w is:

$$\nabla O(w) = -C \left(\sum_{i=1}^{N} y_i x_i \cdot 1_{(y_i(w^T x_i + w_0) < 1)} \right) + w$$

1.2 The updating equation of the model weights based on gradient descent

The gradient points to the direction in which the function ascends the fastest. We update the weight in the opposite direction of the gradient to reach the global minimum.

$$w- = \text{learning\_rate} \times \text{gradient}$$

The learning rate determines the step size. If it is too small, the algorithm takes longer to converge to the minimum, as it would take many small steps. On the other hand, if the learning rate is too large, the algorithm might overshoot the minimum and diverge, failing to converge.

1.3 The indication of the classification accuracies of the training and testing sets based on figure from section 2.1

1.3.1

In the initial iterations, the model learns from the training data leading to a rapid increase in classification accuracy. As iterations pass, the model has learned the most from the training data and can make accurate predictions on the training set. Following the drastic improvement in the accuracy, the model reaches stability. This indicates that the model has converged to a solution where further iterations do not significantly improve the accuracy of the training set. Further iterations can only increase the computational cost with no significant benefit. Similarly, the cost decreases with iterations, first rapidly then reaching stability.

1.3.2

A high training set accuracy suggests that the model has learned the patterns and relationships within the training data. I obtained high testing accuracy similar to the training one suggesting that the model is generalizing well with no overfitting.

1.4 The effect of the learning rate on model training, and on the model performance during testing, based on the results observed in Section 2.2.

If the learning rate is too big, the accuracy and the cost start to get closer to the global minimum, but then converge since the model overshoots. If the learning rate is too small, it takes more iterations to reach the minimum, but it eventually reaches it. Interestingly, the biggest learning rate has the biggest test accuracy. The dataset might have imbalanced classes, where one class is much more prevalent than the other, and hence the overshooting makes the correct decision.

2 Air Quality Analysis by Neural Network

2.1 Conclusions drawn based on the model selection results in Section 3.1

The selected model with a single hidden layer of 100 neurons suggests that the dataset most probably does not have complex non-linear relationships. Since the task is binary classification, the 100 neurons in the single layer are enough to capture the underlying patterns in the data. ReLU is often favored in deep learning for its ability to mitigate vanishing gradient problems and speed up training, which seems beneficial here as well. The MSE of 0.1923 on the testing set is close to the cross-validated MSE, indicating that the model's performance on unseen data is consistent. The high $R\hat{2}$ score of 0.9099 suggests that the model explains about 90.99% of the variance, which is a strong indication of its predictive power.

2.2 Comment on the two training algorithms in Section 3.2.

Both optimizers exhibit converging to a stable point over iterations, with SGD demonstrating quicker convergence. This implies SGD's efficiency in reaching convergence in fewer iterations for this dataset and model architecture. While both show reasonable performance on unseen data, SGD's superior results suggest slightly better generalization. ADAM's default hyperparameters may not have been optimal for this dataset.

3 Experiment

Data was preprocessed using Z-score method for outlier detection. I used Iterative Imputer to get advantage of the data in other features, and Standard Scaler. The MLPRegressor, adept at capturing non-linear relationships, served as the base model. GridSearchCV explored various hyperparameters: hidden layer sizes (100, 50, etc.), 'relu' and 'sigmoid' activations, alpha (0.0001, 0.001, 0.01), learning rates ('constant', 'invscaling', 'adaptive'), momentum (). Cross-validation ensured generalization, vital for datasets with variable distributions, leaving out the need for validation set. After the best parameters were chosen, the whole data set was used to train the model using them.