Air Quality Analysis Report

Wongyo CHOI

April 30, 2024

1 Linear Classification via Gradient Descent

1.1 Model Training and Testing - Objective Function O

(a) Objective Function:

$$O = C \sum_{i=1}^{N} \max(0, 1 - y_i(w^T x_i + w_0)) + \frac{1}{2} w^T w$$

The objective function O minimises the regularised hinge loss, balancing the classification margin maximisation with model complexity.

(b) Derivative of the Hinge Loss Part: When the hinge loss is active (i.e., $1 - y_i(w^T x_i + w_0) > 0$), the gradient of the hinge loss component w.r.t. w is:

$$\nabla_w \left(1 - y_i(w^T x_i + w_0)\right) = -y_i x_i$$

Otherwise, the gradient is zero.

(c) Derivative of the Regularisation Term: The gradient of the regularisation term $\frac{1}{2} w^T w$ is:

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

(d) Total Gradient: Combining these, the gradient of the objective function O w.r.t. w, considering only the samples where the hinge loss is active, is:

$$\nabla_w O = -C \sum_{i:1-y_i(w^T x_i + w_0)>0} y_i x_i + w$$

(e) Gradient Descent Update: The update rule for the weight vector w in gradient descent is:

$$w_{t+1} = w_t - \eta \nabla_w O_t$$

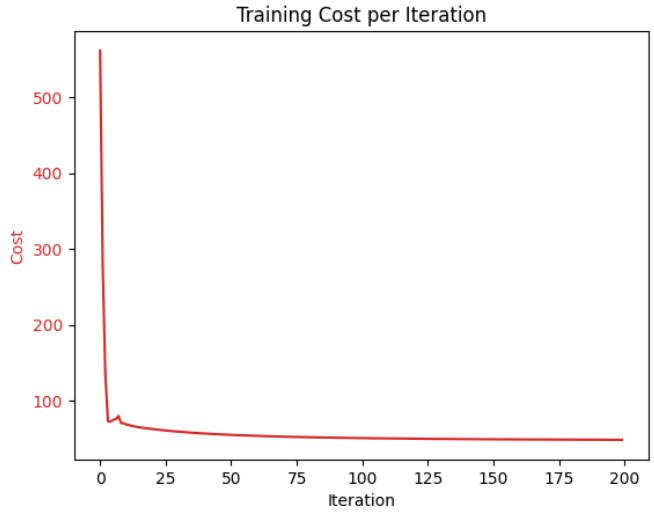
where η is the learning rate.

1.2 Model Training and Testing - Cost, Accuracy, and F1 Score

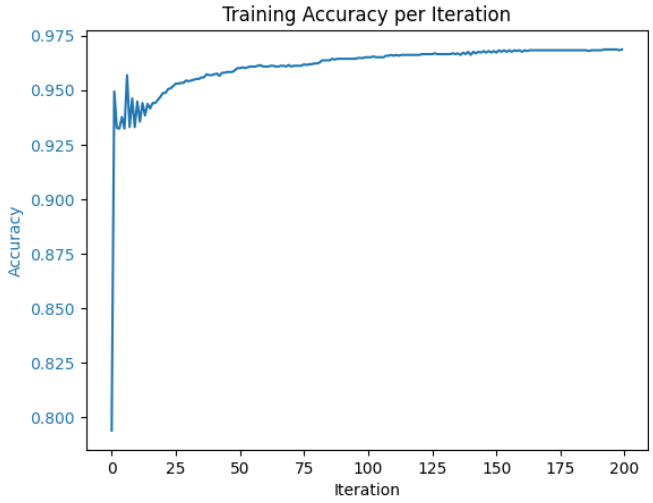
In the early iterations of training, the cost and accuracy graphs exhibit signs of overfitting, with notable fluctuations observed in both metrics. However, as the number of iterations increases, the model demonstrates a stabilised performance, effectively managing the overfitting concern and maintaining consistent accuracy and cost reductions.

For datasets with imbalanced classifications, such as the air quality dataset which predominantly contains 'False' results, it is crucial to consider metrics beyond accuracy. In scenarios where a model might predict every instance as 'False', a high accuracy rate could be misleadingly achieved, yet the F1 Score would significantly diminish due to poor precision and recall. The high F1 Score achieved by this model indicates a balanced

performance with respect to precision and recall, underscoring its capability to accurately predict across different classes without bias.



(a) Training cost reduction over iterations showing rapid initial learning followed by stabilisation.

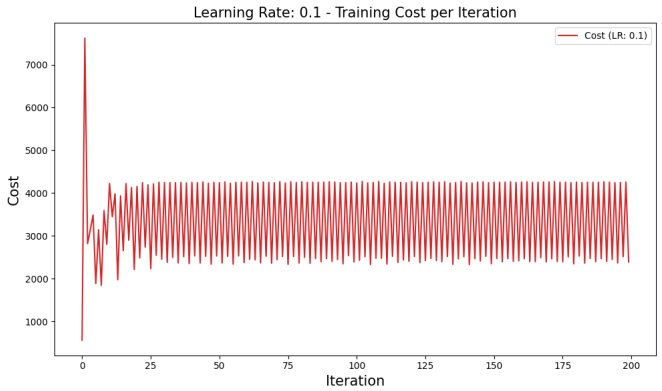


(b) Training accuracy improvement over iterations with early gains leveling off to a steady state.

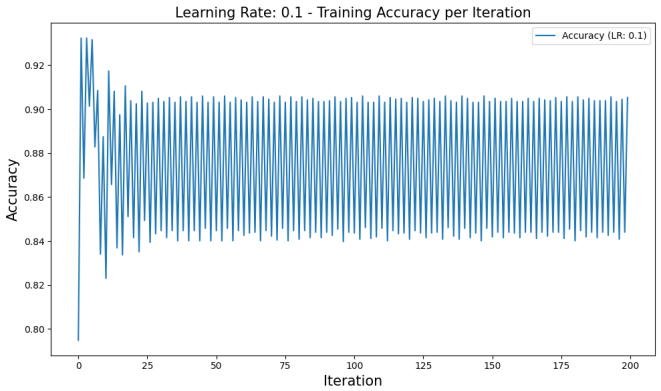
Figure 1: Model optimisation trends depicting the efficacy of the gradient descent algorithm in minimising cost and maximising accuracy through successive training iterations.

1.3 Learning Rate Analysis

The graphs demonstrate the effect of different learning rates (η) on model training and testing performance. Lower learning rates lead to smoother convergence but may require more iterations to minimise the cost. Higher rates accelerate initial learning but can overshoot the minimum, causing oscillations in cost. During testing, an optimal η balances fast convergence with stability, achieving high accuracy and F1 scores.

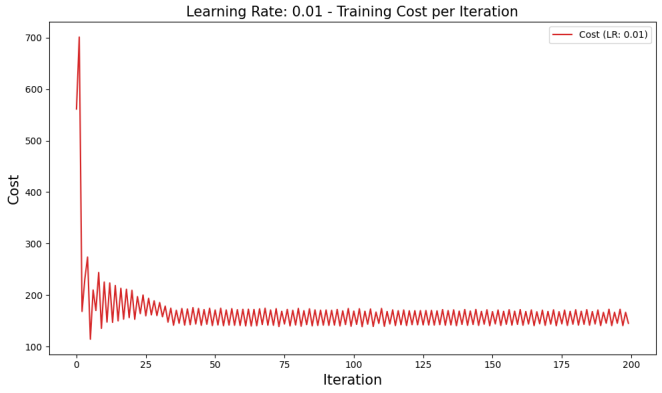


(a) Fluctuating cost over iterations at a learning rate of 0.1, indicating potential overshooting.

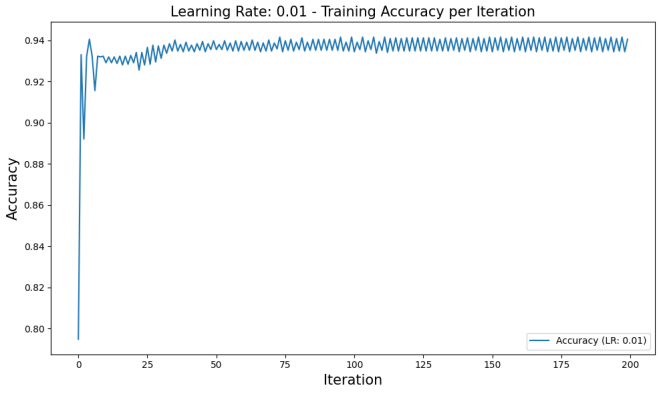


(b) Training accuracy per iteration at a learning rate of 0.1, showing high volatility.

Figure 2: Training performance metrics at a learning rate of 0.1.

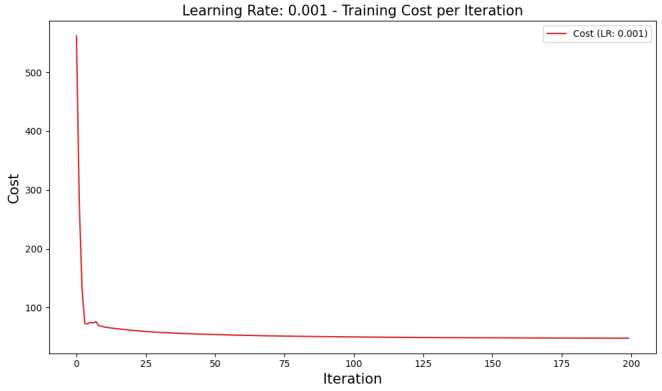


(a) Steady decrease in cost with reduced oscillations at a learning rate of 0.01.

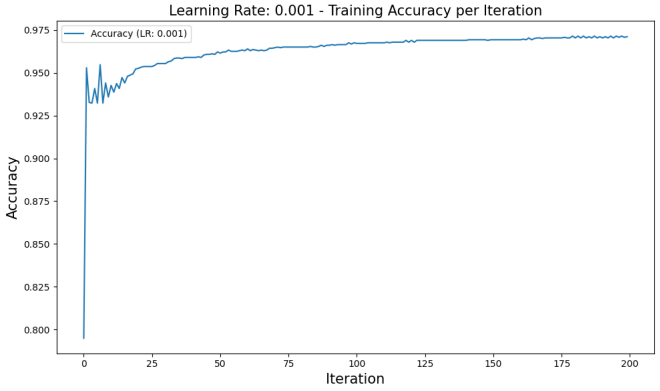


(b) More stable training accuracy per iteration at a learning rate of 0.01.

Figure 3: Training performance metrics at a learning rate of 0.01.

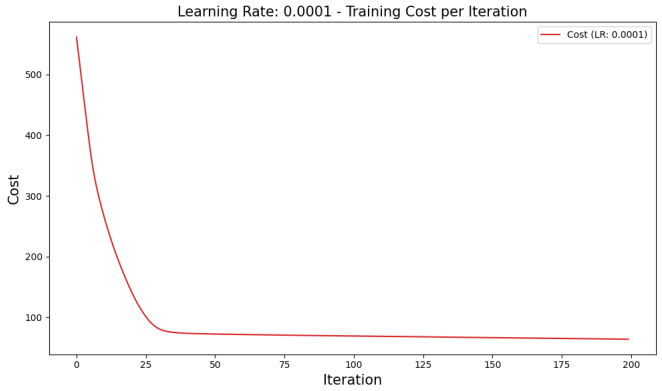


(a) Gradual cost reduction per iteration at a learning rate of 0.001, indicating stable learning.

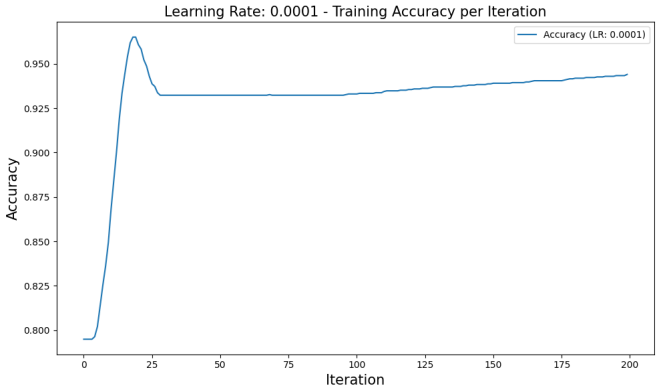


(b) Sustained improvement in training accuracy at a learning rate of 0.001.

Figure 4: Training performance metrics at a learning rate of 0.001.



(a) Smooth and consistent cost descent at a learning rate of 0.0001, optimal for convergence.



(b) Steady and high training accuracy with minimal fluctuation at a learning rate of 0.0001.

Figure 5: Training performance metrics at a learning rate of 0.0001.

2 Air Quality Analysis by Neural Network

2.1 Simple MLP Model Selection

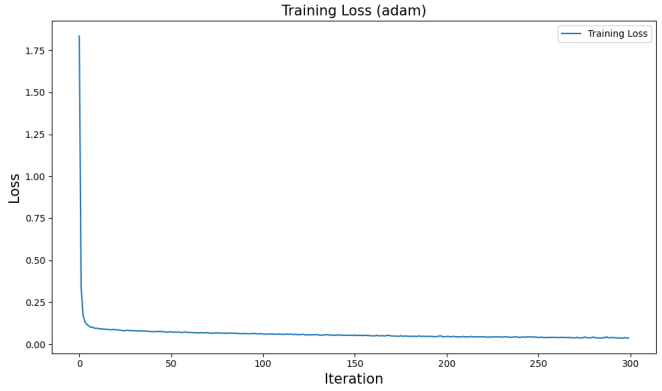
The grid search confirmed 'relu' as the superior activation function over 'logistic', likely due to its ability to effectively address vanishing gradients and support the learning of complex patterns. The chosen hidden layer size of (100, 100) demonstrates an optimal complexity that captures data nuances while preventing overfitting.

This setup, coupled with imputing missing values with the mean and standardising data, ensures that the model is not skewed by outliers or scale variations within the dataset. The consistency of performance is reinforced by 5-fold cross-validation, indicating the model's reliability.

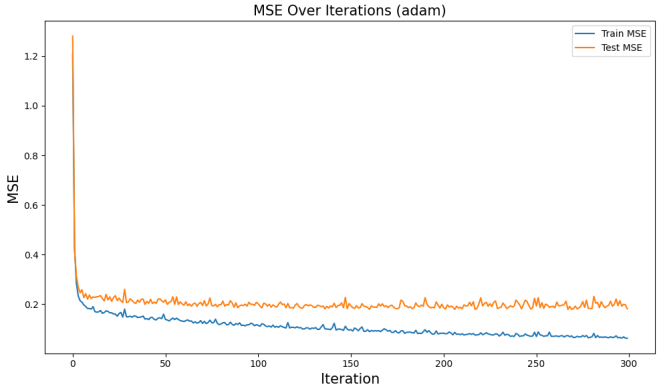
The test results underscore the model's capacity to generalise well to new data, exhibiting both high-quality fit and predictive accuracy. These outcomes highlight the model's adeptness at effectively capturing and generalising the underlying patterns in the data.

2.2 Training Algorithm Comparison: SGD and ADAM

In most case, the 'ADAM' algorithm outperforms 'SGD' in training loss and on both training and test sets for MSE and R^2 scores. The training loss graph for 'ADAM' shows a steeper decrease, indicating more effective optimisation. One key reason 'ADAM' demonstrates superior performance is its lower likelihood of getting trapped in local minima, due to its automated learning rate adjustments, which can navigate more efficiently towards the global minimum. The fluctuations observed in 'ADAM's' graph are indicative of this dynamic optimisation process, where the model avoids local minima to seek the global minimum, a process that may cause variability but ultimately leads to better convergence.

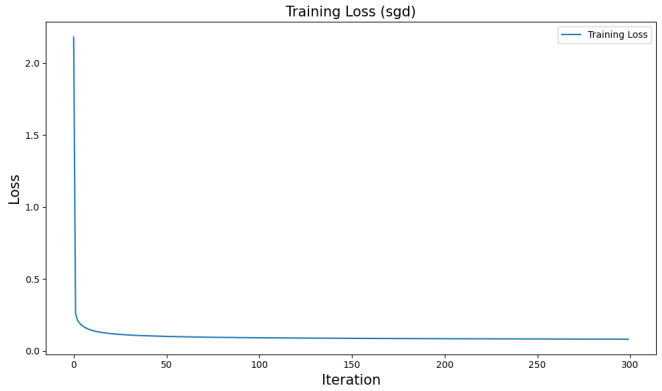


(a) Rapid convergence of training loss using Adam optimiser, showcasing its efficiency.

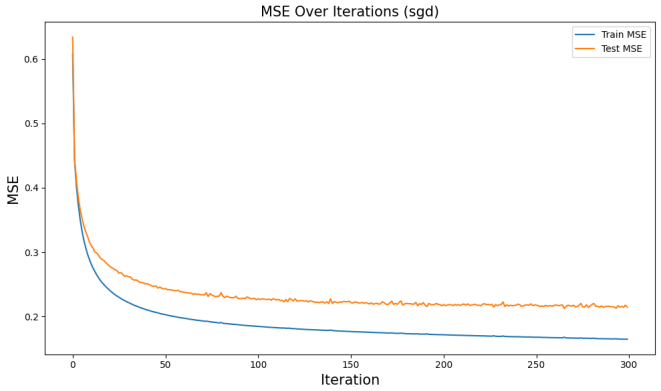


(b) Adam optimiser maintains lower Mean Squared Error, indicating better model fit over iterations.

Figure 6: Performance metrics for the Adam optimiser illustrating quick and stable convergence.



(a) Relatively gradual reduction of training loss with SGD optimiser, indicating steady learning.



(b) Stable Mean Squared Error through iterations with the SGD optimiser, showing consistency.

Figure 7: Performance metrics for the SGD optimiser, demonstrating consistent error minimisation.

3 Build A Robust MLP Regressor

3.1 Model Development

In model development, -999 values were replaced using KNN imputation with $k = 3$, and 'SimpleImputer' was swapped for 'RobustScaler' for better outlier handling. Hyperparameters were dynamically optimised using 'hyperopt' over a range, with potential for three-layer configurations. Early stopping was set to terminate training after 10 iterations without improvement, enhancing efficiency and preventing overfitting. This approach significantly refined the model's predictive accuracy and robustness.

```
space = {
    'hidden_layer_sizes': hp.choice('num_layers', [
        (hp.randint('neurons_in_layer2_1', 50, 150), hp.randint('neurons_in_layer2_2', 50, 150)),
        (hp.randint('neurons_in_layer3_1', 50, 150), hp.randint('neurons_in_layer3_2', 50, 150), hp.randint('neurons_in_layer3_3', 50, 100)),
        ]),
    'activation': 'relu',
    'solver': 'adam',
    'alpha': hp.loguniform('alpha', np.log(1e-4), np.log(1e-2)),
    'early_stopping': True,  # Activate early stopping
    'validation_fraction': 0.1,   # Validation data fraction for early stopping
    'n_iter_no_change': 10  # Iterations to stop if no improvement
}
```

Listing 1: Hyperparameter optimisation setup using the Hyperopt library.