Deep Learning
Homework 1

October 23, 2024

BAIM Mohamed Jalal
313551810

---

1 Regression - Energy efficiency data

Introduction In the first part of this homework, we explore the use of a neural network to model and predict the heating load based on various features.

1.1 Data preprocessing

Before implementing our neural network, we need to deal with our data and prepare it for further processing:

- **Data Loading:**
  - Load data from a CSV file.
  - Convert categorical features to numerical representations using one-hot encoding.
- **Data Splitting and Shuffling:**
  - Define Features and targets
  - Randomly shuffle the dataset.
  - Divide the dataset into training and testing sets.
- **Feature Normalization**

1.2 Network Architecture

For this regression task, a neural network with the following architecture was implemented:

- **Input Layer:** The input layer consists of *input_dim* units. This is determined by the number of features in the preprocessed data. In our case, it likely includes the original 7 features along with the one-hot encoded representations of categorical features.
- **Hidden Layers:** The network utilizes 3 hidden layers with **10** and **5** neurons each. These hidden layers employ the **sigmoid** activation function.

- **Output Layer:** The final layer, the output layer, has *output_dim* unit (set to 1) as we are performing regression for a single target variable (heating load). Since it's a regression task, the output layer uses no activation function to provide the predicted heating load value.

1.3 Learning Curve

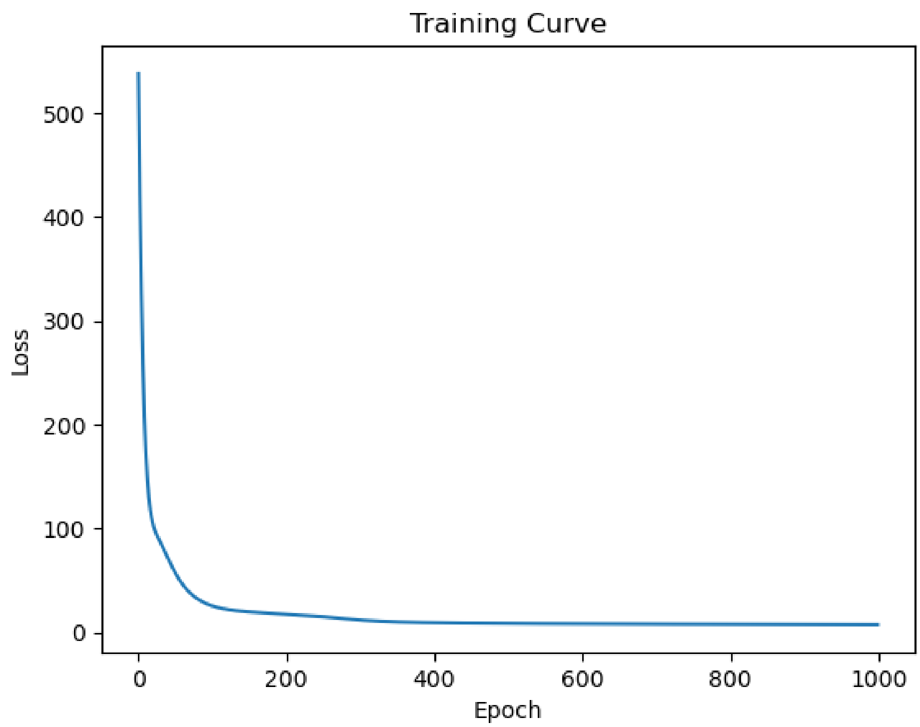

Figure 1: Learning Curve (Loss vs epochs)

The learning curve demonstrates how the model's error changes over epochs. As the number of epochs increases, the error tends to decrease, indicating that the model is learning and adjusting its weights to minimize the error.
In this case, The **learning rat**e was set to 0.001, with **batch size** of 32 and 1000 epochs.

Network Architecture	[10 , 5]
Selected features	[0 ,..., 7]
Training RMS error	2.561
Test RMS error	2.691

1.4 Regression Results with labels

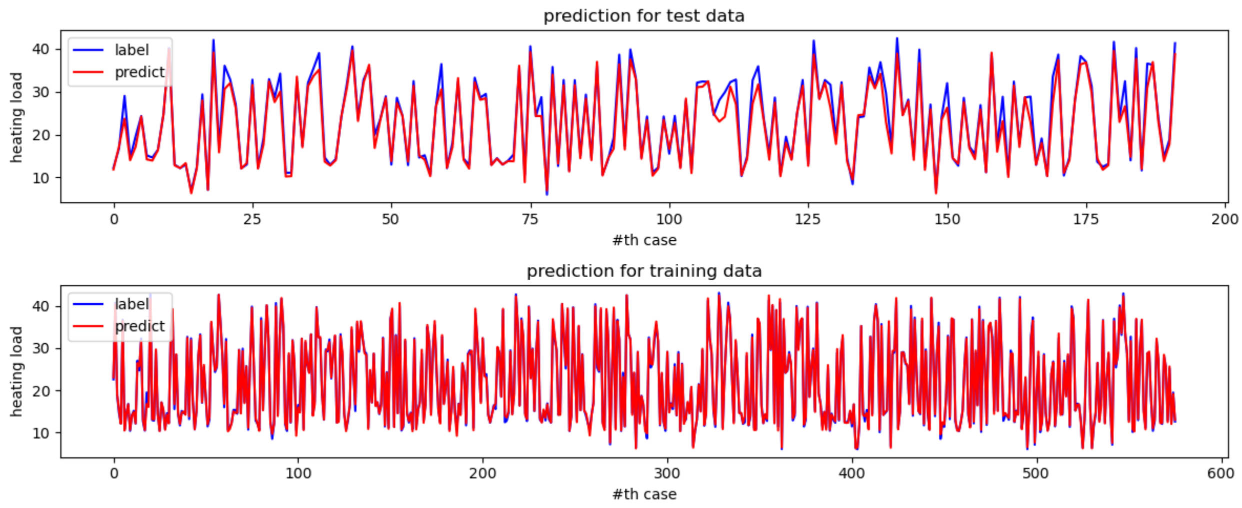

Figure 2: Regression results for both Train & Test Labels

This plot showcases the model's predictions against the true heating load values for both train & test data. A closer alignment of the predictions with the true labels indicates better model performance.

1.4.1 Source of limitations

As observed in the predictions, there's a notable range constraint. The model struggles to accurately predict the extreme values of the target variable, ***'Heating Load.'***:

- **Sigmoid Activation:** The sigmoid activation function can lead to vanishing gradient problems, particularly in deeper architectures. This can hinder training convergence and limit the model's ability to learn complex patterns.While the network isn't deep, the sigmoid function's nature of squashing values between 0 and 1 might limit the model's ability to predict extreme values of 'Heating Load'.
- **Network Simplicity:** With a two hidden layers and only 10 and 5 neurons, the network's architecture may be insufficiently complex to capture the intricate relationships within the data.

1.5 Feature Selection

1.5.1 Methodology

We conducted a feature importance analysis by iteratively removing each feature from the dataset, training a new model, and observing the impact on performance. We compared the Mean Squared Error (MSE) of the modified models to the baseline model(our first configuratuon) to quantify the importance of each feature.

1.5.2 Results

The most important feature, based on the increase in MSE when removed, was **Glazing Area** with an increase of **1.426** in MSE. Other notable features include **Orientation_3.0** and **Glazing Area Distribution_3.0**, with MSE increases of **0.353** and **0.345**, respectively.

Feature	**MSE Change**
Glazing Area	1.426
Orientation_3.0	0.353
Glazing Area Distribution_3.0	0.345
Glazing Area Distribution_2.0	0.313
Surface Area	0.296
Orientation_4.0	0.269
Glazing Area Distribution_1.0	0.260
Orientation_2.0	0.231
Orientation	0.210
Wall Area	0.171
Glazing Area Distribution_0.0	0.108
# Relative Compactness	-0.205
Roof Area	-0.313
Glazing Area Distribution_5.0	-0.365
Glazing Area Distribution	-0.466
Orientation_5.0	-0.548
Overall Height	-0.556
Glazing Area Distribution_4.0	-0.599

Table 1: Feature Importance based on MSE change

2 Classification - Ionosphere dataset

Introduction In this part, we explore the Ionosphere dataset using a simple classification task. We'll detail the network's structure, its learning progression, and how changing the hidden layer's node count affects its performance.

2.1 Data preprocessing

Before implementing our neural network, we need to deal with our data and prepare it for further processing:

- **Data Loading:**
  - Load data from CSV file.
  - Transform the target column to 0 (good) and 1 (bad)
- **Data Splitting and Shuffling:**
  - Shuffle data
  - Split the dataset into training (80%) and testing (20%)

2.2 Network architecture

For the classification task, I implemented the following neural network:

- **Input layer:** 34 units (features).
- **Hidden layers:** One hidden layer with 2 neurons, using Sigmoid activation function.
- **Output layer:** 2 units (for binary classification).

2.3 Learning Curve

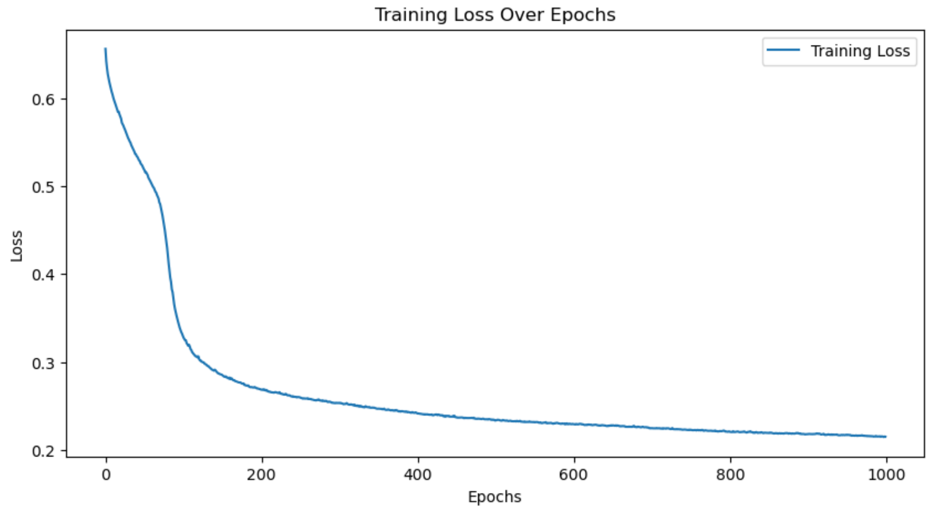

Figure 3: Learning curve showing cross-entropy loss over epochs

Network Architecture	[2] (1 hidden layer)
Selected features	[0,...,33]
Training RMS error	0.121
Test RMS error	0.186

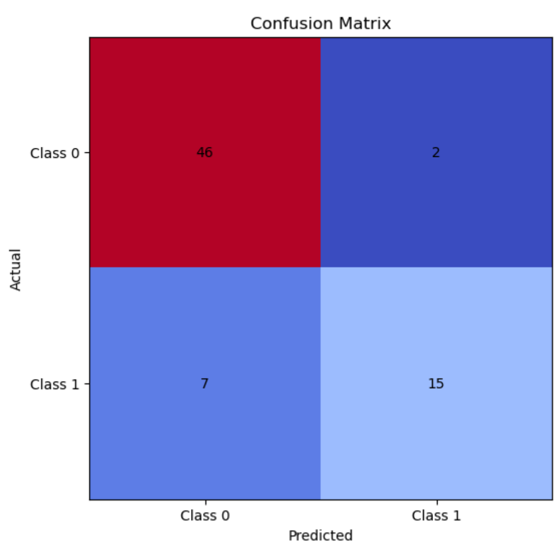

Figure 4: Confusion Matrix of our Model

2.4 Visualization of 10th and 390th Epochs for Different Numbers of Nodes

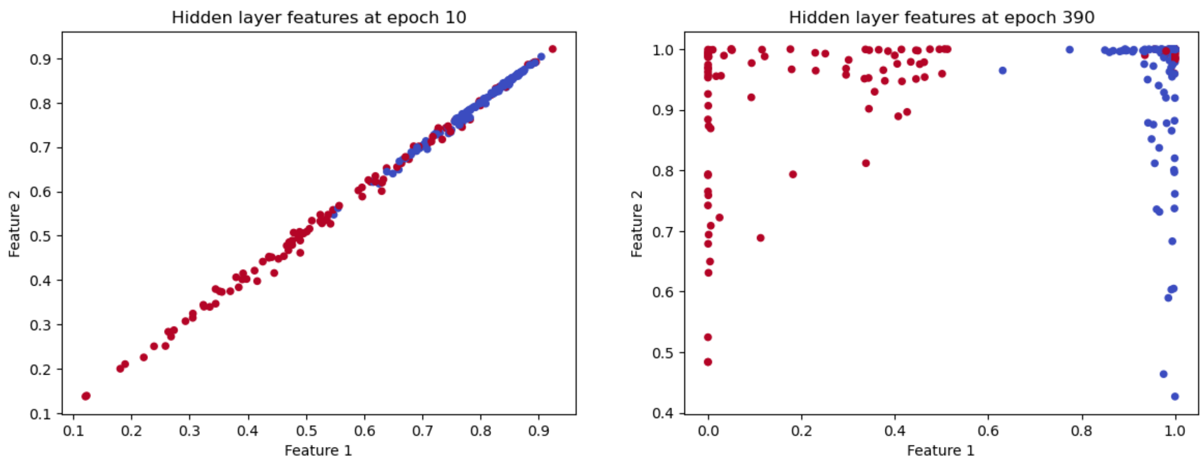

Figure 5: Latent space distribution.

The latent features offer a compact representation of the original data, capturing the key variations essential for classification. By projecting these features into a 2D space, we can visually assess the network's ability to differentiate between the two classes: "good" and "bad."

- **10th Epoch:** At this early stage of training, the latent features display a scattered distribution, with significant overlap between the two classes. The neural network is still adjusting its weights, leading to a less distinct differentiation between classes.
- **390th Epoch:** As training progresses, a clear separation between the two classes becomes visible. This reflects the network's enhanced ability to distinguish between "good" and "bad" based on latent feature representations.

This analysis highlights the critical influence of the number of nodes in the layer just before the output layer on the network's ability to differentiate between classes. A larger number of nodes allows the network to capture more complex patterns, while a smaller number tends to produce a more generalized representation.