Aprendizagem 2023
Homework I – Group 21
(ist106127, ist107)

Part I: Pen and paper

1. Below is a training dataset D composed by two input variables and two output variables, one of which is numerical and the other categorical (y_{num}) (y_{class}). Consider a polynomial basis function $\phi(y_1, y_2) = y_1 y_2$ that transforms the original space into a new one-dimensional space.

Table 1: Provided data

D	y_1	y_2	y_{num}	y_{class}
x_1	1	1	1.25	B
x_2	1	3	7.0	A
x_3	3	2	2.7	C
x_4	3	3	3.2	A
x_5	2	4	5.5	B

Learn a regression model on the transformed feature space using the OLS closed form solution to predict the continuous output y_{num}.

Let's first apply ϕ to obtain the new one-dimensional space that holds the features of our observations.

Table 2: Variables x_1 through x_5 and their transformed features.

D	$\phi(y_1, y_2)$
x_1	1
x_2	3
x_3	6
x_4	9
x_5	8

Using the OLS closed form, learning the requested regression model is a matter of learning $\mathbf{w}$, using the formula:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{z} \tag{1}$$

Where:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 1 & 6 \\ 1 & 9 \\ 1 & 8 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} 1.25 \\ 7.0 \\ 2.7 \\ 3.2 \\ 5.5 \end{bmatrix}$$

Doing the math:

$$\left(\begin{bmatrix}1&1&1&1&1\\1&3&6&9&8\end{bmatrix}\begin{bmatrix}1&1\\1&3\\1&6\\1&9\\1&8\end{bmatrix}\right)^{-1}\begin{bmatrix}1&1&1&1&1\\1&3&6&9&8\end{bmatrix}\begin{bmatrix}1.25\\7.0\\2.7\\3.2\\5.5\end{bmatrix}\overset{\text{numpy}}{\approx}\begin{bmatrix}3.31593\\0.11372\end{bmatrix}$$

Therefore, the trained model is:

$$\hat{y_{num}} = w_0 + w_1 \cdot \phi(y_1, y_2) = 3.31593 + 0.11372 \cdot y_1 \cdot y_2 \tag{2}$$

2. Repeat the previous exercise, but this time learn a Ridge regression with penalty factor $\lambda = 1$. Compare the learnt coefficients with the ones from the previous exercise and discuss how regularization affects them.

In order to perform a Ridge Regression, we'll use the same $\mathbf{X}$ and $\mathbf{z}$, but a slightly different formula:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda \cdot \mathbf{I})^{-1}\mathbf{X}^T\mathbf{z} \tag{3}$$

Again, by plugging in the numbers:

$$\left(\begin{bmatrix}1&1&1&1&1\\1&3&6&9&8\end{bmatrix}\begin{bmatrix}1&1\\1&3\\1&6\\1&9\\1&8\end{bmatrix}+\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)^{-1}\begin{bmatrix}1&1&1&1&1\\1&3&6&9&8\end{bmatrix}\begin{bmatrix}1.25\\7.0\\2.7\\3.2\\5.5\end{bmatrix}\overset{\text{numpy}}{\approx}\begin{bmatrix}1.81809\\0.32379\end{bmatrix}$$

Giving us the model:

$$\hat{y_{num}} = w'_0 + w'_1 \cdot \phi(y_1, y_2) = 1.81809 + 0.32379 \cdot y_1 \cdot y_2 \tag{4}$$

The best way to visualize these coefficients and their meaning is to plot them, as we see below in Figure 1.

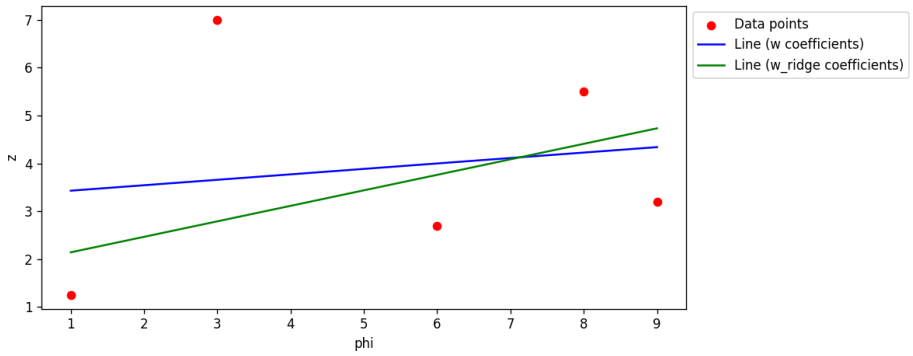

Figure 1: Provided data and Linear and Ridge Regression lines plotted.

Here, we can clearly see that the effect regularization had on the coefficients was: decrease w_0 and increase w_1.

Although the increase in w_1 is slightly counterintuitive when it comes to regularization, we can assume that this feature is important in explaining the data.

As for the decrease in the w_0 coefficient, this is more like a Ridge regression, and means the model is attributing a lesser relative importance to the y intercept in explaining the data (and, like we saw before, regarding the impact of the $\phi(y_1, y_2)$ variable as larger).

3. Given three new test observations and their corresponding y_{num} output $x_6 = (2, 2, 0.7)$, $x_7 = (1, 2, 1.1)$, and $x_8 = (5, 1, 2.2)$, compare the train and test RMSE of the two models obtained in 1) and 2). Explain if the results go according to what is expected.

In order to calculate the RMSE, we use the following formula:

$$\textbf{RMSE} = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} ||\mathbf{z} - \hat{\mathbf{z}}||} \tag{5}$$

On the data:

Table 3: Test set including x_6, x_7, and x_8

D	y_1	y_2	y_{num}
x_6	2	2	0.7
x_7	1	2	1.1
x_8	5	1	2.2

Let's start by computing the prediction given by each model.

These are given by inputting y_1 and y_2 of each test point in the Equations 2 and 4, which gives us, respectively, $\hat{z}$ and $\hat{z}_{ridge}$.

Table 4: Predicted values of z using the w and w_{ridge} coefficients.

D	$\hat{z}$	$\hat{z}_{ridge}$
x_1	3.42965	2.14184
x_2	3.65708	2.78936
x_3	3.99823	3.76064
x_4	4.33938	4.73191
x_5	4.22566	4.40816
x_6	3.77080	3.11312
x_7	3.54336	2.46560
x_8	3.88451	3.43688

$\mathbf{z}$ is presented in Table 3 and 1, and therefore we have all the parameters needed to apply the **RMSE** formula.

These are the results, when applied to each of the train/test sets and on the predictions of each model.

Table 5: Train and Test RMSE

	Train	Test
Normal	2.02650	2.46559
Ridge	2.15354	1.75289

The RMSE calculated do match what is expected. Naturally, the Ridge penalty damages the performance of the model when evaluating on the train set - RMSE for the Ridge model on the train set vs the OLS model. However, in the test set, the roles are inverted: now, the OLS model presents a much higher RMSE.

A lower RMSE in the test set vs the train set is synonymous with better generalization, which is precisely the goal of regularization techniques such as the Ridge regression.

4. Consider an MLP to predict the output y_{class} characterized by the weights

$$W^{[1]} = \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & 0.2 \\ 0.2 & 0.1 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 0.1 \\ 0 \\ 0.1 \end{bmatrix} \quad W^{[2]} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad b^{[2]} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

The output activation function

$$\text{softmax}(Z_C^{[out]}) = \frac{e^{Z_C^{[out]}}}{\sum_{l=1}^{|C|} e^{Z_l^{[out]}}}$$

no activations on the hidden layer(s) and the cross-entropy loss:

$$-\sum_{i=1}^{N}\sum_{l=1}^{|C|} t_l^{(i)} \log(X_l^{[out](i)})$$

Consider also that the output layer of the MLP gives the predictions for the classes A, B, and C in this order. Perform one stochastic gradient descent update to all the weights and biases with learning rate $\eta = 0.1$ using the training observation x_1.

The MLP model we pretend to update is pictured in Figure 4

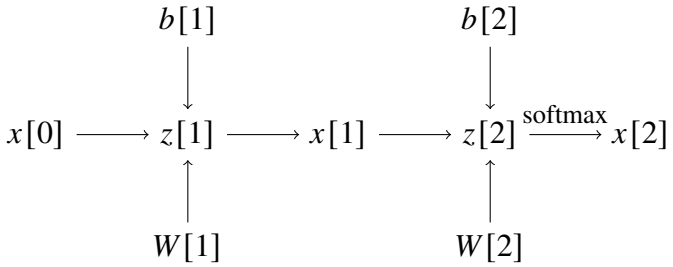

Figure 2: A representation of the forward propagation through the layers of this MLP

First, we perform propagation below.

Propagation

(a) First, we obtain $z^{[1]}$:

$$z^{[1]} = W^{[1]}x^{[0]} + b^{[1]}$$

$$z^{[1]} = \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & 0.2 \\ 0.2 & 0.1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.0 \\ 0.1 \end{bmatrix}$$

$$z^{[1]} = \begin{bmatrix} 0.3 \\ 0.3 \\ 0.4 \end{bmatrix}$$

(b) Then, as there is no activation function in this hidden layer, we get to $x^{[1]}$:

$$x^{[1]} = z^{[1]}$$

(c) We get $z^{[2]}$ through computations in the hidden layer:

$$z^{[2]} = W^{[2]}x^{[1]} + b^{[2]}$$

$$z^{[2]} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.3 \\ 0.3 \\ 0.4 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$z^{[2]} = \begin{bmatrix} 2.7 \\ 2.3 \\ 2.0 \end{bmatrix}$$

(d) And, then, we get to $x^{[2]}$, our output, by using the activation function:

$$x^{[2]} = \text{softmax}\left(z^{[2]}\right)$$

$$x^{[2]} = \text{softmax}\left(\begin{bmatrix} 2.7 \\ 2.3 \\ 2.0 \end{bmatrix}\right)$$

$$x^{[2]} = \begin{bmatrix} \dfrac{e^{2.7}}{e^{2.7} + e^{2.3} + e^{2.0}} \\ \dfrac{e^{2.3}}{e^{2.7} + e^{2.3} + e^{2.0}} \\ \dfrac{e^{2.0}}{e^{2.7} + e^{2.3} + e^{2.0}} \end{bmatrix} = \begin{bmatrix} 0.46149 \\ 0.30934 \\ 0.22917 \end{bmatrix}$$

Then, onto backpropagation.

Backpropagation

(a) Compute δs.

$$\delta^{[2]} = \frac{\partial \epsilon}{\partial z^{[2]}} = \frac{\partial \epsilon}{\partial x^{[2]}} \frac{\partial x^{[2]}}{\partial z^{[2]}} = x^{[2]} - t = \begin{bmatrix} 0.46149 \\ 0.30934 \\ 0.22917 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.46149 \\ -0.69066 \\ 0.22917 \end{bmatrix}$$

$$\delta^{[1]} = \left(\frac{\partial z^{[2]}}{\partial x^{[1]}}^T \delta^{[2]} \right) \circ \frac{\partial x^{[1]}}{\partial z^{[1]}} \overset{x^{[1]}=z^{[1]}}{=} \left(W^{[2]^T} \delta^{[2]} \right) = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 1 \\ 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.46149 \\ -0.69066 \\ 0.22917 \end{bmatrix} = \begin{bmatrix} -2.77556 \cdot 10^{-17} \\ -0.22917 \\ 0.46149 \end{bmatrix}$$

(b) Use δs to update weights and biases.

First, we update $W^{[2]}$

$$W^{[2]}_{new} = W^{[2]}_{old} - \eta \frac{\partial \epsilon}{\partial W^{[2]}} = W^{[2]}_{old} - \eta \left(\delta^{[2]} \frac{\partial z^{[2]}}{\partial W^{[2]}}^T \right) = W^{[2]}_{old} - \eta \left(\delta^{[2]} x^{[1]^T} \right) =$$

$$= \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} - 0.1 \cdot \begin{bmatrix} 0.46149 \\ -0.69066 \\ 0.22917 \end{bmatrix} \begin{bmatrix} 0.3 & 0.3 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.98616 & 1.98616 & 1.98154 \\ 1.02072 & 2.02072 & 1.02763 \\ 0.99313 & 0.99313 & 0.99083 \end{bmatrix}$$

Now, let us update $b^{[2]}$:

$$b^{[2]}_{new} = b^{[2]}_{old} - \eta \frac{\partial \epsilon}{\partial b^{[2]}} = b^{[2]}_{old} - \eta \left(\delta^{[2]} \frac{\partial z^{[2]}}{\partial b^{[2]}}^T \right) \overset{\frac{\partial z^{[2]}}{\partial b^{[2]}}=1}{=} b^{[2]}_{old} - \eta \delta^{[2]} =$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - 0.1 \cdot \begin{bmatrix} 0.46149 \\ -0.69066 \\ 0.22917 \end{bmatrix} = \begin{bmatrix} 0.95385 \\ 1.06907 \\ 0.97708 \end{bmatrix}$$

Then $W^{[1]}$

$$W^{[1]}_{new} = W^{[1]}_{old} - \eta \frac{\partial \epsilon}{\partial W^{[1]}} = W^{[1]}_{old} - \eta \left(\delta^{[1]} \frac{\partial z^{[1]}}{\partial W^{[1]}}^T \right) = W^{[1]}_{old} - \eta \left(\delta^{[1]} x^{[0]^T} \right) =$$

$$= \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & 0.2 \\ 0.2 & 0.1 \end{bmatrix} - 0.1 \cdot \begin{bmatrix} -2.77556 \cdot 10^{-17} \\ -0.22917 \\ 0.46149 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 \\ 0.12292 & 0.22292 \\ 0.15385 & 0.053851 \end{bmatrix}$$

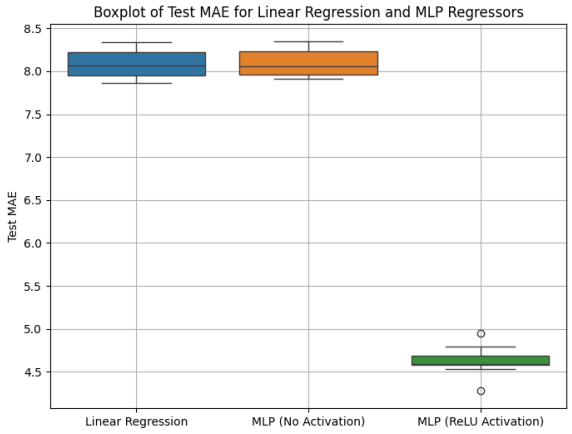

Figure 3: Boxplot with Linear Regression, MLP with no activation and MLP using ReLu.

And, finally, $b^{[1]}$

$$b^{[1]}_{new} = b^{[1]}_{old} - \eta \frac{\partial \epsilon}{\partial b^{[1]}} = b^{[1]}_{old} - \eta \left(\delta^{[1]} \frac{\partial z^{[1]}}{\partial b^{[1]}}^T \right) \overset{\frac{\partial z^{[1]}}{\partial b^{[1]}}=1}{=} b^{[1]}_{old} - \eta \delta^{[1]} =$$

$$= \begin{bmatrix} 0.1 \\ 0.0 \\ 0.1 \end{bmatrix} - 0.1 \cdot \begin{bmatrix} -2.77556 \cdot 10^{-17} \\ -0.22917 \\ 0.46149 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.02292 \\ 0.05385 \end{bmatrix}$$

Using the first established weigths and biases, we see that the MLP would classify this vector as being an instance of class A (through $\text{argmax}_c(x^{[2]})$). That is the wrong classification - the right one is B. By using backpropagation, we updated the weight, such that the new MLP output is $\begin{bmatrix} 0.40923 \\ 0.36617 \\ 0.22460 \end{bmatrix}$. It would still classify it wrong, however, the loss would now be smaller (1.0046 vs the 1.1733 we had before).

Part II: Programming

`The code and the discussion to questino 7. of Part II can be found in the notebook which is also included in the sent zip file, or in our` GitHub repo.

6. Compare a Linear Regression with a MLP with no activations, and explain the impact and the importance of using activation functions in a MLP. Support your reasoning with the results from the boxplots.

   Mathematically, if it has no activation functions, an MLP is essentially equivalent to a regression.

   A linear regression, in its essence, takes the input data, applies a linear transformation and adds a constant term (w_0).

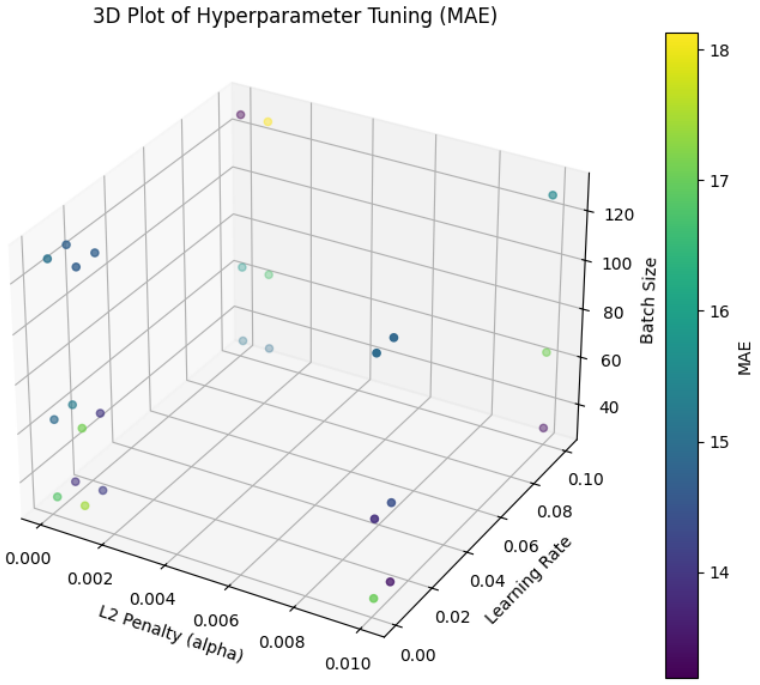

Figure 4: Plot with combinations of hyperparameters and learning rate

If it has no activation function, an MLP is essentially doing the same: it applies a linear transformation, adds a constant term; then, it takes that as a new observation and applies a linear transformation, adding a constant term, etc.

If we think about it carefully, it is almost the same as applying multiple linear transformations to the initial observation and summing to it a constant term - not quite, but as an abstraction it is close enough. As we know, composing multiple linear transformations is the same as applying a single linear transformation.

And so, taking this line of thought, an MLP with no activation function should be equivalent to a linear regression. In the boxplot, this is precisely what we see. The MAE of both models is quite similar (and drastically different from the MLP with an activation function, which distorts the linear relationships).

7. Plot the test MAE for each combination of hyperparameters, report the best combination, and discuss the trade-offs between the combinations

   From the plot, we can tell that higher batch sizes clearly perform better, (128 consistently led to better model performance) - this makes sense because larger batch sizes allow the model to calculate gradients based on more data, which results in smoother updates during learning.

   Moreover, larger batches are typically associated with improved generalization because the learning process is less noisy, and the model learns a more accurate representation of the underlying data distribution.

Smaller batch sizes perform especially bad with low L2 penalties: the effect of L2 regularization is large. Models trained with low L2 penalties, particularly around 0.0001, struggled to generalize, especially with higher learning rates.

One of the clearest patterns is that higher L2 penalties work best when paired with lower learning rates. These allow the model to learn more slowly, making smaller adjustments to the weights, contributing to more stable learning. When combined with regularization from the L2 penalty, this leads to significantly better performance, as the model is both learning carefully and avoiding overfitting.