LMP1210H Winter 2025 Assignment 1

Yuliang Xiao – yl.xiao@mail.utoronto.ca

January 29, 2025

Instructors: Rahul G. Krishnan & Bo Wang

By turning in this assignment, I agree by the UofT honor code and declare that all of this is my own work.

Problem 1 - Basics about Machine Learning

(a) Answers

Definition:
overfitting means that the model is trained to have very good and detailed understanding of hidden patterns on training data but does not have good generalization on the test or unseen data. This usually comes with good learning of the noises of the data.

Techniques to prevent overfitting:

1. Introduce cross-validation strategy to the training so that model can be evaluated on the validation dataset.
2. Add the regularization loss to the cost function.
3. Reduce the complexity of the model architecture.
4. Add dropout layers to the model to reduce the dependencies of features.

(b) Answers

Impacts:
The magnitude of k controls the performance of the model.

1. k is small, the model learns a good underlying pattern of the data and thus predicts strong boundaries to label the data, but this may cause the model to over-fitting because it is sensible to a random sample in the training data.
2. k is large, it makes a stable/smooth prediction by averaging over many examples, but the model may be under-fitting since it does not capture important regularities.

How to choose k:

1. Choose optimal k based on the number of data points or the size of data pool n, use the rule of thumb: $k = n^{\frac{2}{2+d}}$ (d refers to dimensionality).
2. Measure the generalization error on the unseen data set with different choices of k.

(c) Answers

- **Regression category**
  - **Goal:** explore the relationship between independent variables (age, gender, BMI, height, etc) with the volume of hippocampus.
  - **Method:** we can use linear regression to model the continuous outcome which represents the volume of hippocampus and thus provides better judgment for neurodegenerative diseases.
  - **Risk:**
    * The linear regression may not capture accurate relationship between independent variables and volume of hippocampus, since volume of hippocampus may have non-linear relationship with independent variables.
    * The model may be over-fitting if there are too many independent variables or features using to predict outcomes while the data size is small.
    * The model may be poor and has bias if the variances of errors among each independent variables in data are not constant.
    * The model may be sensitive to outliers for both independent and dependent variables.
- **Classification category**
  - **Goal:** make a judgment if the patient will have the stroke given the current and previous medical records like (systolic blood pressure, BMI, age, etc).
  - **Method:** we can use logistic regression to model the binary outcomes which tells whether or not the stroke will happen.
  - **Risk:**
    * The logistic regression results may not be accurate since it relies on the threshold which is a hyper-parameter to predict binary results. Picking an optimal threshold is a challenge work.
    * The model may be over-fitting if there are too many independent variables or features using to predict outcomes while the data size is small.
    * The data set is imbalanced. The distribution of stroke patients is not approximate to that of non-stroke one. The model could predict the majority class (no stroke) more often, ignoring the minority class (stroke).

Problem 2 - Python Familiarity

Answers: please click and check this Google Colab Notebook.

Problem 3 - Regularized Linear Regression

(a) Answers

$$\frac{\partial \mathcal{J}_{\text{reg}}^{\beta}}{\partial w_j} = \frac{\partial \mathcal{J}}{\partial w_j} + \frac{\partial \mathcal{R}}{\partial w_j} = \frac{\partial \mathcal{J}}{\partial y}\frac{\partial y}{\partial w_j} + \frac{\partial \mathcal{R}}{\partial w_j} \tag{1}$$

$$\frac{\partial \mathcal{J}}{\partial y}\frac{\partial y}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})x_j{}^{(i)} = \frac{1}{N}\sum_{i=1}^{N}(\sum_{j=1}^{D} w_j x_j{}^{(i)} - t^{(i)})x_j{}^{(i)} \tag{2}$$

$$\frac{\partial \mathcal{R}}{\partial w_j} = \frac{\partial}{\partial w_j}(\frac{1}{2}\sum_{j=1}^{D}\beta_j {w_j}^2) = \beta_j w_j \tag{3}$$

Place Eq. (2) and Eq. (3) to Eq. (1), we will get

$$\frac{\partial \mathcal{J}_{\text{reg}}^{\beta}}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(\sum_{j=1}^{D} w_j x_j{}^{(i)} - t^{(i)})x_j{}^{(i)} + \beta_j w_j \tag{4}$$

To update the gradient with respect to w_j, we can use the equation

$$w_j \leftarrow w_j - \eta\frac{\partial \mathcal{J}_{\text{reg}}^{\beta}}{\partial w_j} \tag{5}$$

Place Eq. (4) to Eq. (5) and we get final answer

$$w_j \leftarrow w_j - \eta(\frac{1}{N}\sum_{i=1}^{N}(\sum_{j=1}^{D} w_j x_j{}^{(i)} - t^{(i)})x_j{}^{(i)} + \beta_j w_j) \tag{6}$$

Where η is the learning rate which is a hyper-parameter defined by users.

If we know that j-th feature is less important which means this feature should be removed or ignored in the forward process to predict outcomes. To achieve this, changing the value of w_j to a very small value is rational. Therefore, we can assign a large value to β_j to penalize the w_j toward to 0 and thus reduce the importance of that variable contributing to the model.

(b) Answers

Given the Eq. (4) we got in question (a), we can make some mathematical transformation to obtain expected format,

$$\begin{aligned}\frac{\partial \mathcal{J}_{\text{reg}}^{\beta}}{\partial w_j} &= \frac{1}{N}\sum_{i=1}^{N}(\sum_{j=1}^{D} w_j {x_j}^{(i)} - t^{(i)}){x_j}^{(i)} + \beta_j w_j = 0 \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)} x_j^{(i)} - \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} + \beta_j w_j = 0\end{aligned} \tag{7}$$

Transfer the term $(\frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)})$ in Eq. (7) to right hand side

$$\frac{1}{N}\sum_{i=1}^{N}\sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)} x_j^{(i)} + \beta_j w_j = \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} \tag{8}$$

Since variable $w_{j'}$ is only dependent on parameter j' instead of i, we can swap the order of the summation in Eq. (8).

$$\sum_{j'=1}^{D} w_{j'}(\frac{1}{N}\sum_{i=1}^{N} x_{j'}^{(i)} x_j^{(i)}) + \beta_j w_j = \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} \tag{9}$$

For the regularization term $\beta_j w_j$, we know that this term only contribute when $j' = j$. Therefore, we can create a diagonal matrix $\delta_{jj'}$ where $\delta_{jj'} = 1$ only when $j' = j$ otherwise it is 0. Indeed, we can combine the $\beta_j w_j$ with first term in Eq. (9),

$$\sum_{j'=1}^{D} w_{j'}(\frac{1}{N}\sum_{i=1}^{N} x_{j'}^{(i)} x_j^{(i)} + \beta_j \delta_{jj'}) = \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} \tag{10}$$

Finally we can write the Eq. (10) in the expected format,

$$\sum_{j'=1}^{D}(\frac{1}{N}\sum_{i=1}^{N} x_{j'}^{(i)} x_j^{(i)} + \beta_j \delta_{jj'}) w_{j'} - \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} = 0 \tag{11}$$

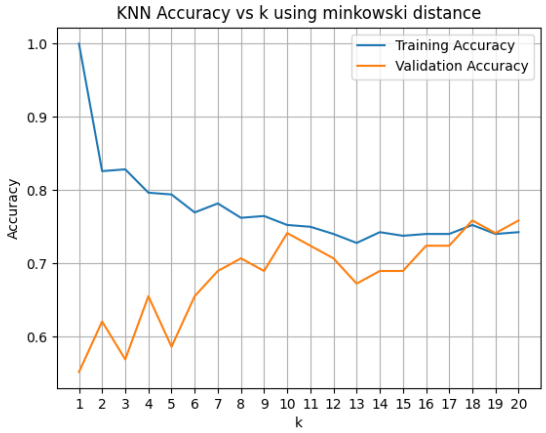
where

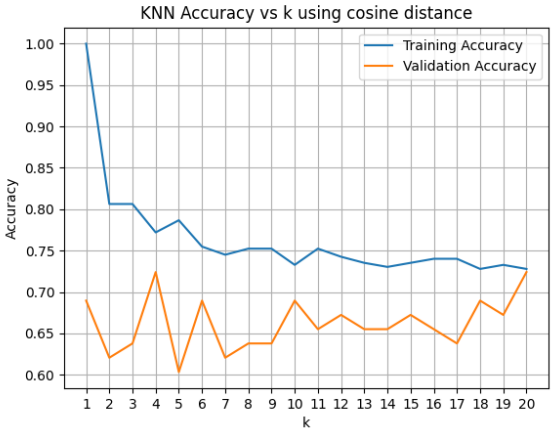
$$\begin{aligned} A_{jj'} &= \frac{1}{N}\sum_{i=1}^{N} x_{j'}^{(i)} x_j^{(i)} + \beta_j \delta_{jj'} \\ c_j &= \frac{1}{N}\sum_{i=1}^{N} t^{(i)} x_j^{(i)} \end{aligned} \tag{12}$$

Problem 4 - Classification with Nearest Neighbours

Please check the README.md and run the code A1_code.py file.

The plots and accuracy report of training and validation accuracy with different k selections and metrics are showed:





KNN Report		
metric	best k	accuracy
minkowski	4	0.684
cosine	20	0.709

Table 1: Test accuracy of KNN model with different metrics (All values are rounded to 3 significant figures).

From the figures of "minkowski" distance, when k reaches to 18 and 20, they have the same validation accuracies, and we will choose k=18 for the final answer because it gives us small size of model. Also, for "cosine" distance, we find the two same validation accuracy at k=4 and k=20, finally, we will choose k=4 for the same reason.

Problem 5 - Classification with Decision Tree

Please check the README.md and run the code A1_code.py file.
The screenshots and accuracy report of first 2 decision tree layers with different min_sample_leaf are showed, given the accuracy table, we can find that model is overfitting for min_sample_leaf=1.

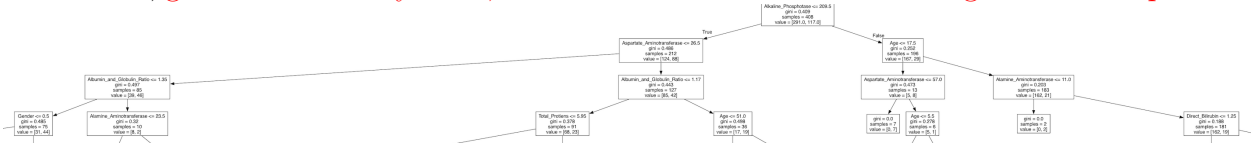


Figure 1: Screenshot of first 2 decision tree layers with min_sample_leaf=1

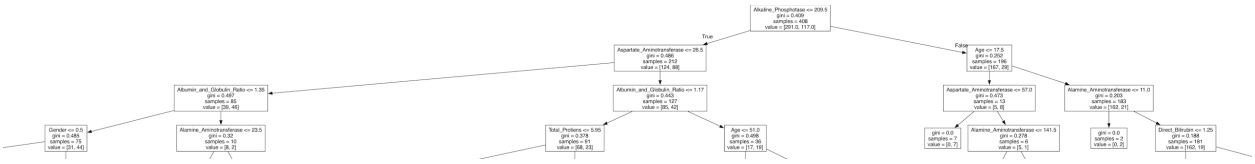


Figure 2: Screenshot of first 2 decision tree layers with min_sample_leaf=2

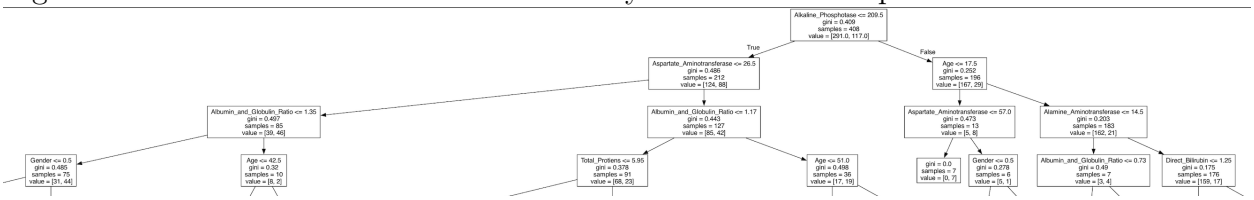


Figure 3: Screenshot of first 2 decision tree layers with min_sample_leaf=3

Decision Tree Report			
min_sample_leaf	acc_train	acc_valid	acc_test
1	1.00	0.638	0.607
2	0.946	0.621	0.632
3	0.926	0.569	0.615

Table 2: Train, validation and test accuracy of decision tree model with different min_sample_leaf (All values are rounded to 3 significant figures).

From the Table 2, we can find that when `min_sample_leaf`=1, the train accuracy becomes 1 and others are not which fits the definition of over-fitting, so we increase the `min_sample_leaf`=2 and `min_sample_leaf`=3, the train accuracy becomes lower but still high, the validation loss follows the changes of train accuracy, this may indicate that the model is under-fitting.

Problem 6 - Classification with Logistic Regression

Please check the README.md and run the code A1_code.py file.
The accuracy report of logistic regression is showed:

Logistic Regression Report		
acc_train	acc_valid	acc_test
0.740	0.741	0.718

Table 3: Train, validation and test accuracy of logistic regression model (All values are rounded to 3 significant figures).

From the Table 3, we can see that the validation accuracy is very closed to the train accuracy and its value is not too low. This shows that our model has low variance and is generalizing well to unseen data, which can be further validated by test accuracy.