FAI HW3

b08902045 資工四 袁紹奇

Hand-written Part

Problem 1

Consider $1 - y\mathbf{w}^T\mathbf{x} > 0$, then the gradient of a term is

$$\nabla\big(\max(1 - y_n\mathbf{w}^T\mathbf{x}_n, 0)\big)^2 = \nabla(1 - y_n\mathbf{w}^T\mathbf{x}_n)^2 = 2(1 - y_n\mathbf{w}^T\mathbf{x}_n)\cdot(-y_n\mathbf{x}_n)$$

If $1 - y\mathbf{w}^T\mathbf{x} \leq 0$, then the gradient is simply 0.

Combine the above scenarios and summing over N terms, we have the following:

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N}\sum_{n=1}^{N}\big(\max(1 - y_n\mathbf{w}^T\mathbf{x}_n, 0)\big)\cdot(-y_n\mathbf{x}_n)$$

Problem 2

Since the function $f(x) = x$ and $g(x) = \log x$ are both strictly increasing, and $f(x_1) > f(x_2) \iff g(x_1) > g(x_2)$, we can take log to the right side.

$$\mathbf{u}^* = \arg\max_{\mathbf{u}\in\mathbb{R}^d}\prod_{n=1}^{N} p_{\mathbf{u}}(\mathbf{x}_n) = \arg\max_{\mathbf{u}\in\mathbb{R}^d}\sum_{n=1}^{N}\log p_{\mathbf{u}}(\mathbf{x}_n)$$

To find the maximum, find $\mathbf{u}^*$ that makes the derivative 0. Let $g(\mathbf{u}) = \log p_{\mathbf{u}}(\mathbf{x}_n)$, we have

$$\frac{d}{d\mathbf{u}}\log p_{\mathbf{u}}(\mathbf{x}_n) = \frac{1}{p_{\mathbf{u}}(\mathbf{x}_n)}\cdot\frac{d}{d\mathbf{u}}p_{\mathbf{u}}(\mathbf{x}_n)$$

Since $p_{\mathbf{u}}(\mathbf{x}_n)$ is the probability density function of $\mathcal{N}(\mathbf{u}, \mathbf{I})$, where $\mathbf{I}$ is the identity matrix, we have

$$\frac{d}{d\mathbf{u}}p_{\mathbf{u}}(\mathbf{x}_n) = \frac{1}{(2\pi)^{d/2}}\frac{d}{d\mathbf{u}}\exp\left(-\frac{1}{2}(\mathbf{x}_n - \mathbf{u})^T(\mathbf{x}_n - \mathbf{u})\right) = \frac{\mathbf{x}_n - \mathbf{u}}{(2\pi)^{d/2}}\exp\left(-\frac{1}{2}(\mathbf{x}_n - \mathbf{u})^T(\mathbf{x}_n - \mathbf{u})\right) = (\mathbf{x}_n - \mathbf{u})\cdot p_{\mathbf{u}}(\mathbf{x}_n)$$

Combine the term we have

$$\frac{d}{d\mathbf{u}}\log p_{\mathbf{u}}(\mathbf{x}_n) = \mathbf{x}_n - \mathbf{u}$$

The original problem would be

$$0 = \sum_{n=1}^{N}\mathbf{x}_n - \mathbf{u} = \left(\sum_{n=1}^{N}\mathbf{x}_n\right) - N\cdot\mathbf{u}$$

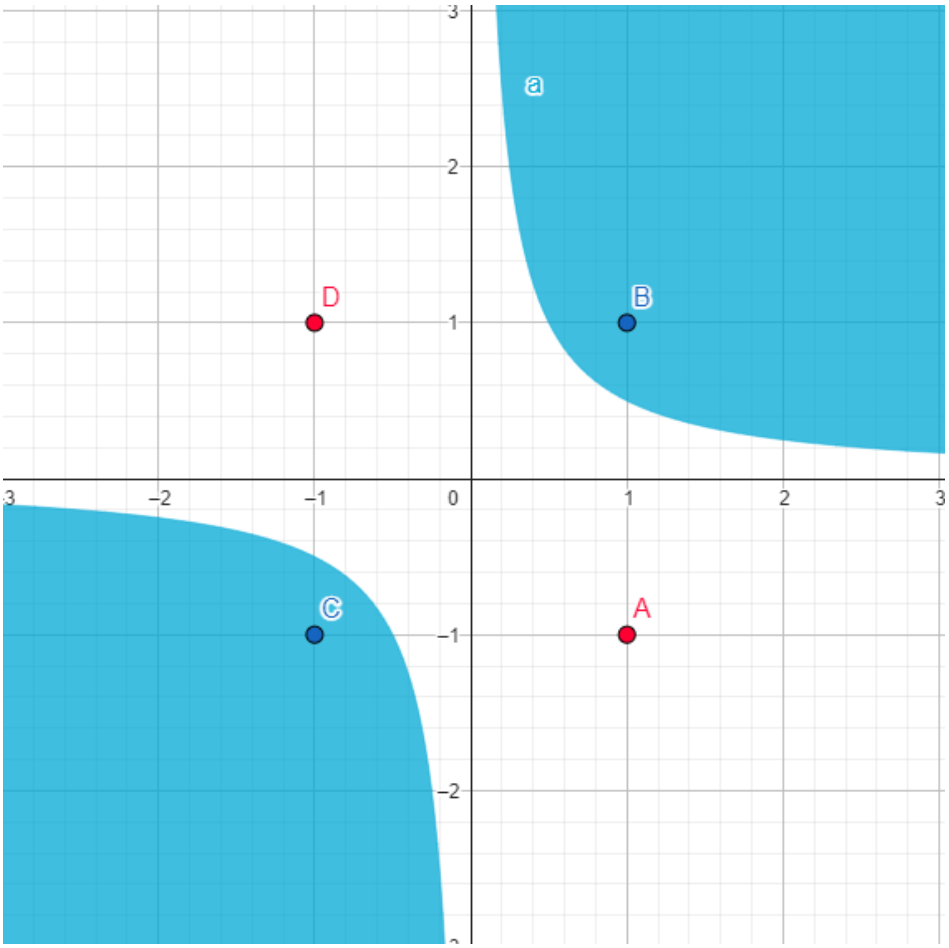
By selecting $\mathbf{u}^* = \frac{1}{N}\cdot\sum_{n=1}^{N}\mathbf{x}_n$ gives us the maximum, hence proved.

Problem 3

Select $\tilde{\mathbf{w}}^T = [0.5, 0, 0, 0, -1, 0]$, we have

$$\tilde{\mathbf{w}}^T \mathbf{z}_n = 0.5 \cdot 1 - 1 \cdot x_1 x_2$$

The selection makes the data perfectly separate the data. (red=+1, blue=-1)



Problem 4

Since all the positive samples where $g_t(\mathbf{x}_n) = y_n$ makes $\delta(g_t(\mathbf{x}_n), y_n) = 0$. Consider the negative samples, where $g_t(\mathbf{x}_n) \neq y_n$, we have

$$\sum_{n=1}^{N} w_n^{t+1} \delta(g_t(\mathbf{x}_n), y_n) = d_t \cdot \epsilon_t \cdot \sum_{n=1}^{N} w_n^t = \sqrt{\epsilon_t \cdot (1-\epsilon_t)} \cdot \sum_{n=1}^{N} w_n^t$$

Since $d_t = \sqrt{(1-\epsilon_t)/\epsilon_t}$, we have

$$\sum_{n=1}^{N} w_n^{t+1} = d_t \cdot \epsilon_t \cdot \sum_{n=1}^{N} w_n^t + \frac{1}{d_t} \cdot (1-\epsilon_t) \cdot \sum_{n=1}^{N} w_n^t = \sqrt{(1-\epsilon_t) \cdot \epsilon_t} \cdot \sum_{n=1}^{N} w_n^t + \sqrt{\epsilon_t \cdot (1-\epsilon_t)} \cdot \sum_{n=1}^{N} w_n^t = 2\sqrt{\epsilon_t \cdot (1-\epsilon_t)} \cdot \sum_{n=1}^{N} w_n^t$$

Combining the above 2 terms, we have

$$\frac{\sum_{n=1}^{N} w_n^{t+1} \delta(g_t(\mathbf{x}_n), y_n)}{\sum_{n=1}^{N} w_n^{t+1}} = \frac{\sqrt{\epsilon_t \cdot (1-\epsilon_t)} \cdot \sum_{n=1}^{N} w_n^t}{2\sqrt{\epsilon_t \cdot (1-\epsilon_t)} \cdot \sum_{n=1}^{N} w_n^t} = 0.5$$

Programming Part

(a)

Accuracy for classification task

Logistic Regression	Decision Tree	Random Forest
0.9556	0.8222	0.8444

MSE for regression task

Linear Regression	Decision Tree	Random Forest
22.6624	19.3996	10.7194

In the classification task, the logistic regression model performs better in this case because linear models are already capable for this task. On the other hand, the decision tree and random forest models have a larger set of possible solutions and can easily overfit the training data, resulting in lower performance on new, unseen data. This is supported by the fact that the decision tree and random forest models achieve 100% accuracy on the training set, indicating overfitting.

In the regression task, linear models may not be powerful enough to capture complex relationships between the features and the target variable. The random forest regressor performs better in this case because it can capture both linear and nonlinear relationships between the features and the target variable. It combines multiple decision trees, allowing it to handle more complex patterns and reduce overfitting compared to a single decision tree.

Generally, random forest tends to perform better than a single decision tree because it leverages the ensemble technique resulting in robustness under noises.

(b)

Normalization scales the features to a range between 0 and 1. This can be beneficial when the features have different scales or units. It ensures that each feature contributes equally to the model and prevents one feature from dominating the others. If the min or max values of each feature are unstable or the data contains outliers, this approach may result in worse performance.

Standardization transforms the features to have zero mean and unit variance. It can be useful when the features have different distributions. It helps the model by centering the features around zero and scaling them to a comparable range, making the optimization process more efficient. If the original data was assumed to sample from a Gaussian distribution, then utilize this approach tends to perform well.

The following table compared random forest (RF) for classification and regression tasks with normalization and standardization.

Classification

only RF	normalization, RF	standardization, RF
0.9556	0.8444	0.9111

Regression

only RF	normalization, RF	standardization, RF
10.1533	10.7194	23.3727

From the experiments, it is observed that applying both normalization and standardization techniques does not significantly improve the performance of the RF model in both classification and regression tasks. The reasons for using normalization or standardization are typically to increase the robustness of the model and prevent the problem of gradient vanishing. However, the RF model does not rely on gradient descent to update the model parameters. Instead, it combines multiple decision trees, making it less sensitive to the scale and distribution of the features. Therefore, the impact of normalization and standardization on the RF model is minimal, and the model performs well even without applying these preprocessing techniques.

(c)

Based on the provided configurations, here are the performance results for the logistic regression model:

Configuration 1:

Learning Rate: 0.001
Iterations: 100000
Accuracy: 0.9556

Configuration 2:

Learning Rate: 0.05
Iterations: 1000
Accuracy: 0.9333

Config 1, with a larger number of iterations, allows the model more time to update its parameters and reach convergence. It explores the solution space more thoroughly, which can lead to better performance. However, a higher number of iterations also increases the computational time required for training.

Config 2, with a higher learning rate, takes larger steps to update the parameters. This can result in faster convergence but also increases the risk of overshooting the optimal solution. In this case, Config 2 achieved a slightly lower accuracy compared to Config 1, indicating that it may have overshot the optimal solution.

In conclusion, a smaller learning rate allows for finer parameter updates, while a larger number of iterations provides more opportunities for convergence.

(d)

The number of trees determines how many individual decision trees are combined in the random forest. Increasing the number of trees can improve the model's performance up to a certain point, but it also increases computational complexity and training time. It helps to reduce overfitting and enhance generalization as the ensemble of trees provides more robust predictions.

The maximum depth controls the depth of each decision tree in the random forest. A deeper tree can capture more complex relationships in the data, but it can also lead to overfitting. Limiting the maximum depth helps to prevent overfitting and makes the model less complex. It balances the trade-off between model complexity and generalization.

In general, I tuned the parameters based on performance considerations. Given the relatively small dataset used in this homework, it was unnecessary to set a high maximum depth for the random forest. The number of trees was determined by the desired computational cost.

(e)

Linear models are simple and interpretable. They work well when there is a linear relationship between the features and the target variable. They are computationally efficient, especially with large datasets. However, they may struggle to capture complex, nonlinear patterns and can be limited by their assumptions.

Nonlinear models offer a larger solution space. They are more flexible and can handle complex patterns with better performance. Random forests, in particular, reduce overfitting by combining multiple decision trees. However, they can be computationally expensive and less interpretable than linear models.

If the problems or the dataset have linear relationship between features and labels, I would choose linear models. However, in cases where the problem is intricate and the relationship between features and labels is difficult to capture, I would choose nonlinear models even it requires more computational resources. If the dataset contains noises or the variance is large, I would choose random forest over decision tree to prevent overfitting and improve generalization.

Ref:
https://www.quora.com/Why-is-squared-hinge-loss-differentiable
https://ithelp.ithome.com.tw/articles/10226726
https://chat.openai.com/