HW 5

黃熙漢

5. $E_{PR}(w) = \frac{1}{N}\left(\sum^{N} (w^T x_n - y_n)^2 + \lambda \sum_{i=0}^{d} \alpha_i w_i^2\right)$

$\nabla E_{PR}(w) = \frac{2}{N} X^T (Xw - y) + \frac{2\lambda}{N} Dw = 0$

$D : \text{diag}(\alpha_0, \alpha_1 \dots)$

$X^T X w + \lambda D w = X^T y$

$(X^T X + D\lambda) w = X^T y \dots\dots ①$

$\nabla E_{PV}(w) = \frac{2}{N+K}(X^T(Xw - y) + \tilde{X}^T \tilde{X} w) = 0$

$(X^T X + \tilde{X}^T \tilde{X}) w = X^T y \dots\dots ②$

For prove that ① = ②, we need to prove that $\lambda D = \tilde{X}^T \tilde{X}$

$\tilde{x}_k = \begin{bmatrix} 0 \\ \vdots \\ \sqrt{\lambda \alpha_{k-1}} \\ \vdots \\ 0 \end{bmatrix} \in R^{1+d}$

$\tilde{X}^T \tilde{X} = \sum_{k=1}^{K} \tilde{x}_k \tilde{x}_k^T$

$\tilde{x}_k \tilde{x}_k^T = (\sqrt{\lambda \alpha_{k-1}})^2 e_k e_k^T = \lambda a_{k-1} e_k e_k^T$

$\tilde{X}^T \tilde{X} = \sum_{k=1}^{K} \lambda \alpha_{k-1} e_k e_k^T = \lambda \cdot \text{diag}(\alpha_0 .. \alpha_1, \dots \alpha_d) = \lambda D$

$\tilde{X}^T \tilde{X} = \lambda D$

$(X^T X + \lambda D) w = X^T y$

$\therefore w_{PR}^* = w_{PV}^*$ Both approaches serve to prevent overfit by controlling the complexity of the model, ensuring that w do not become too large.

6. Given w^* minize $E_{in}(w)$, so $\nabla E_{in}(w^*) = 0$

$E_{in}(w) \approx E_{in}(w^*) + \frac{1}{2}(w-w^*)^T H (w-w^*)$

approximate $\tilde{E}_{aug}(w)$

$\tilde{E}_{aug} = E_{in}(w) + \frac{\lambda}{N}||w||^2 \approx E_{in}(w^*) + \frac{1}{2}(w-w^*)^T H(w-w^*) + \frac{\lambda}{N}||w||^2$

$E_{in}(w^*)$ = const (ignore)

optimize :

$$\min_{w \in R^{d+1}} \left[\frac{1}{2}(w-w^*)^T H (w-w^*) + \frac{\lambda}{N}||w||^2\right]$$

$\nabla \tilde{E}_{aug}(w) = H(w-w^*) + 2\frac{\lambda}{N} w = 0$

Rearrange: $H(w-w^*) + 2\frac{\lambda}{N} = 0$

$(H + 2\frac{\lambda}{N} I)w = Hw^*$, where I : $(d+1) \times (d+1)$

Solve w^*_{aug} (minize)

$w^*_{aug} = (H + 2\frac{\lambda}{N} I)^{-1} H w$

$\therefore w^*_{aug} = (I + 2\frac{\lambda}{N} H^{-1})^{-1} w^*$

expression demonstrates how L2 regularization adjusts the original minimicer w^* by incorporating the Hessian H and the regularization parameters λ and N.

7.

estimator: $\bar{y} = \frac{1}{N-k} \sum_{n=1}^{N-k} y_n$

$\bar{y}$ mean:

$$\mathbb{E}[\bar{y}] = \frac{1}{N-k} \sum_{n=1}^{N-k} \mathbb{E}[y_n] = \frac{1}{N-k} \cdot (N-k)\, 0 = 0$$

Variance:

$$\mathrm{Var}(\bar{y}) = \mathrm{Var}\left(\frac{1}{N-k} \sum_{n=1} (y_n)\right)$$

$$= \left(\frac{1}{N-k}\right)^2 \sum_{n=1}^{N} \mathrm{Var}(y_n)$$

$$= \frac{1}{(N-k)^2} \cdot (N-k) \cdot \sigma^2$$

$$= \frac{\sigma^2}{N-k}$$

interese Term!

$$(y_n - \bar{y})^2$$

Expectation:

$$\mathbb{E}[(y_n - \bar{y})^2] = \mathbb{E}[y_n^2] + \mathbb{E}[\bar{y}^2] - 2\mathbb{E}[y_n \bar{y}]$$

1. $\mathbb{E}[y_n^2] = \mathrm{Var}(y_n) + (\mathbb{E}[y_n])^2 = \sigma^2 + 0 = \sigma^2$

2. $\mathbb{E}[\bar{y}^2]$:

$\mathbb{E}[\bar{y}^2] = \mathrm{Var}(\bar{y}) + (\mathbb{E}[\bar{y}])^2 = \frac{\sigma^2}{N-k} + 0 = \frac{\sigma^2}{N-k}$

3. $\mathbb{E}[y_n \bar{y}]$:

$\because$ training data independene with Validation:

y_n and $\bar{y}$ independent

$\mathbb{E}[y_n \bar{y}] = \mathbb{E}(y_n] \cdot \mathbb{E}[\bar{y}] = 0$

$\mathbb{E}[(y_n - \bar{y})^2] = \sigma^2 + \frac{\sigma^2}{-k} - 0 = \sigma^2 (1 + \frac{1}{N-k})$

$\frac{1}{K} \sum_{n=N-k+1}^{N} \mathbb{E}[(y_n - \bar{y})^2]$

$= \frac{1}{k} \cdot k \cdot \sigma^2 (1 + \frac{1}{N-k}$

$= \sigma^2 (1 + \frac{1}{N-k})$ #

Fixed hypo with zero mean:

error purely due to the inherent variance of data sample mean estimate:

$\frac{\sigma^2}{N-k}$ (variance due to estimate the mean fom a finite training set),

training data:

$N-k \uparrow$, expect validation error approach σ^2

$N-k \downarrow$, expect validation err significantly increase

8.

$E_{in}(w_0^*)$

$$w_0^* = \frac{1}{N} \sum_{n=1}^{N} y_n$$

$$E_{in}(w_0^*) = \frac{1}{N} \sum_{n=1}^{N} (w_0^* - y_n)^2$$

$$E_{loocv}(A_{avg}) = \frac{1}{N} \sum_{n=1}^{N} E[(w_{0,-n} - y_n)^2]$$

$$w_{0,-n} = \frac{1}{N-1} \sum_{\substack{m=1 \\ m \neq n}}^{N} y_m$$

$$w_0^* = \frac{1}{N} y_n + \frac{N-1}{N} w_{0,-n}$$

$$w_{0,-n} = \frac{Nw_0^* - y_n}{N-1}$$

$$w_{0,-n} - y_n = \frac{Nw_0^* - y_n}{N-1} - y_n = \frac{Nw_0^* - y_n - (N-1)y_n}{N-1}$$

$$= \frac{N}{N-1}(w_0^* - y_n)$$

$$(w_{0,-n} - y_n)^2 = \left(\frac{N}{N-1}\right)^2 (w_0^* - y_n)^2$$

compute $\mathbb{E}[(w_0^* - y_n)^2]$

given $w_0^* = \frac{1}{N}\sum_{m=1}^{N} y_m$, y_m i.i.d, mean μ, Var σ^2

$$\mathbb{E}[(w_0^* - y_n)^2] = Var(w_0^* - y_n) + \mathbb{E}[w_0^* - y_n])^2$$

$\mathbb{E}[w_0^*] = \mu$ $\mathbb{E}[y_n] = \mu$:

$$\mathbb{E}[w_0^* - y_n] = \mu - \mu = 0$$

$$\mathbb{E}[(w_0^* - y_n)^2] = Var(w_0^* - y_n)$$

$$w_0^* - y_n = \frac{1}{N} \sum_{m \neq n} y_m - \frac{N-1}{N} y_n$$

all independent:

$$Var(w_0^* - y_n) = Var\left(\frac{1}{N} \sum_{m \neq n} y_m - \frac{N-1}{N} y_n\right)$$

$$= \frac{N-1}{N^2} \sigma^2 + \frac{(N-1)^2}{N^2} \sigma^2$$

$$E[(w_0^* - y_n)^2] = \frac{N-1}{N} \sigma^2$$

take expectation of $(w_{0,-n} - y_n)^2$

$$E[(w_{0,-n,} - y_n)^2] = \left(\frac{N}{N-1}\right)^2 \cdot \frac{N-1}{N} \sigma^2$$

$$= \frac{N}{N-1} \sigma^2$$

$$E_{in}(w_0^*) = \frac{1}{N} \sum_{n=1}^{N} (w_0^* - y_n)^2$$

$$E[E_{in}(w_0^*)] = \frac{N-1}{N} \sigma^2$$

$$E(w_{0,-n} - y_n)^2] = \frac{N}{N-1} \cdot E[E_{in}(w_0^*))$$

$$= \frac{N}{N-1} \cdot E_{in}(w_0^*)$$

$$E_{loocv}(A_{avg}) = \frac{1}{N}\sum^{n} E[(w_{0,-n} - y_n)^2]$$

$$= \frac{1}{N} \cdot N \cdot \frac{N}{N-1} \cdot E_{in}(w_0^*$$

$$= \frac{N}{N-1} \cdot E_{in}(w_0^*)$$

$$E_{loocv}(A_{avg}) = \frac{N}{N-1} \cdot E_{in}(w_0^*)$$

9. Uniform Distribution:

$$P(y=+1) = P(y=-1) = \frac{1}{2}$$

Expected error for g: $E_{out}(g) = \frac{1}{2}\epsilon_+ + \frac{1}{2}\epsilon_-$

Nonuniform Distribution:

$p(y=-1) = p$

$p(y=+1) = 1-p$

Expect error g: $E_{out}(g) = (1-p)\epsilon_+ + p\,\epsilon_-$

expect error g_c: $E_{out}(g_c) = p$

Find p which $E_{out}(g) = E_{out}(g_c) = p$

$(1-p)\varepsilon_+ + p\varepsilon_- = p$

$\varepsilon_+ - p\varepsilon_+ - p\varepsilon_- = p$

$p(\varepsilon_- - \varepsilon_+) - p = -\varepsilon_+$

$p[(\varepsilon_- - \varepsilon_+) - 1] = -\varepsilon_+$

$p(-\varepsilon_+ + \varepsilon_- - 1) = -\varepsilon_+$

$p = \frac{\varepsilon_+}{1+\varepsilon_+ - \varepsilon_-}$, $\because 0 \leq \varepsilon_+ \leq 1 , 0 \leq \varepsilon_- \leq 1$, $1 + \varepsilon_+ - \varepsilon_- > 0$ (valid)

$\therefore p < \frac{\varepsilon_+}{1+\varepsilon_+ - \varepsilon_-}$: classifier g is better than g_c

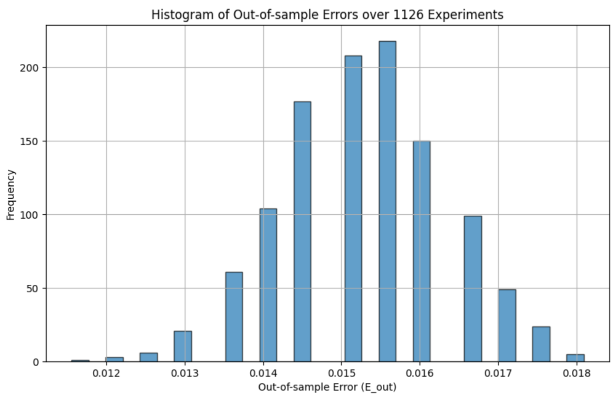
$p = \frac{\varepsilon_+}{1+\varepsilon_+ - \varepsilon_-}$: g equally good as g_c

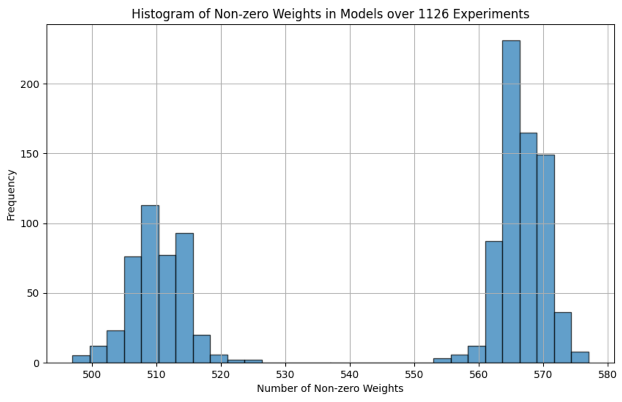
$p > \frac{\varepsilon_+}{1+\varepsilon_+ - \varepsilon_-}$: g is worse than g_c

Higher ϵ_+ : p increase, g perform good than g_c

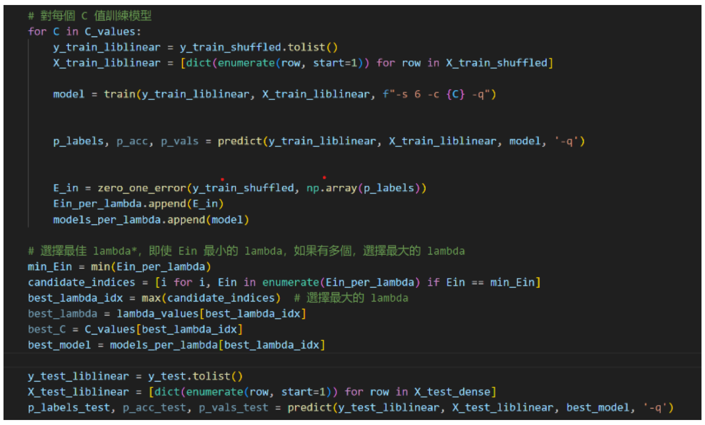
High ϵ_- : p decrease, reducing the range where g outperform g_c.

10. Result:





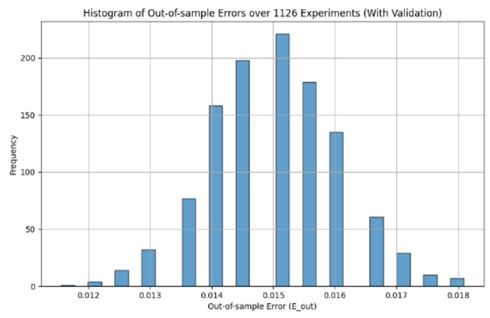
Code:

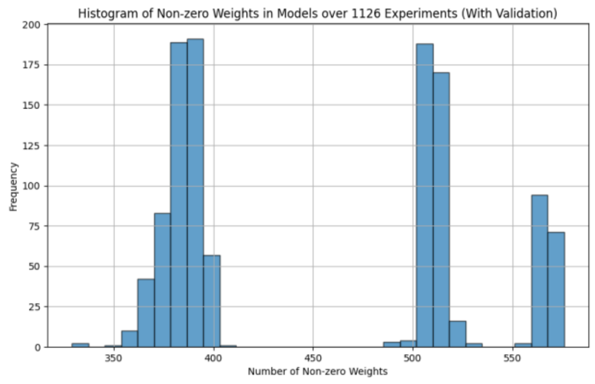


```python
# 對每個 C 值訓練模型
for C in C_values:
    y_train_liblinear = y_train_shuffled.tolist()
    X_train_liblinear = [dict(enumerate(row, start=1)) for row in X_train_shuffled]

    model = train(y_train_liblinear, X_train_liblinear, f"-s 6 -c {C} -q")


    p_labels, p_acc, p_vals = predict(y_train_liblinear, X_train_liblinear, model, '-q')


    E_in = zero_one_error(y_train_shuffled, np.array(p_labels))
    Ein_per_lambda.append(E_in)
    models_per_lambda.append(model)

# 選擇最佳 lambda*, 即使 Ein 最小的 lambda, 如果有多個, 選擇最大的 lambda
min_Ein = min(Ein_per_lambda)
candidate_indices = [i for i, Ein in enumerate(Ein_per_lambda) if Ein == min_Ein]
best_lambda_idx = max(candidate_indices)  # 選擇最大的 lambda
best_lambda = lambda_values[best_lambda_idx]
best_C = C_values[best_lambda_idx]
best_model = models_per_lambda[best_lambda_idx]

y_test_liblinear = y_test.tolist()
X_test_liblinear = [dict(enumerate(row, start=1)) for row in X_test_dense]
p_labels_test, p_acc_test, p_vals_test = predict(y_test_liblinear, X_test_liblinear, best_model, '-q')
```

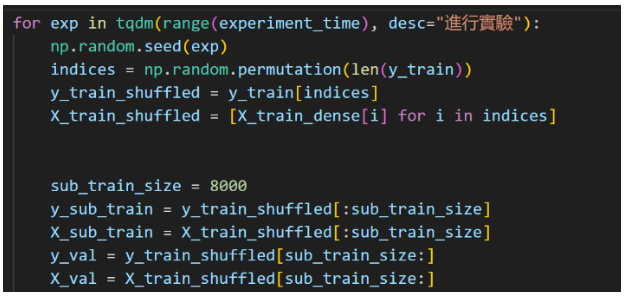
11 . Describe :Based on the results of Problems 10 and 11, we observe a slight shift in the distribution of Eout and non weight zero after introducing validation. Specifically, the distribution moves to the left, indicating lower error rates. This supports the theory that random sampling from the dataset to simulate real-world scenarios, combined with validation, helps achieve smaller Eout . By using a validation set, we can better select the optimal regularization parameter λ∗, leading to improved generalization and reduced overfitting. As a result, the model performs better on unseen data, validating the importance of these steps in achieving more robust results. Similarly, we observe a leftward shift in the distribution of non-zero weights. By comparing this with Eout, we find that models with the smallest number of non-zero weights tend to achieve better robustness and lower Eout. This highlights the importance of sparsity in the weight vector, as fewer non-zero weights often correspond to simpler models that generalize better to unseen data.

Result:





Code:

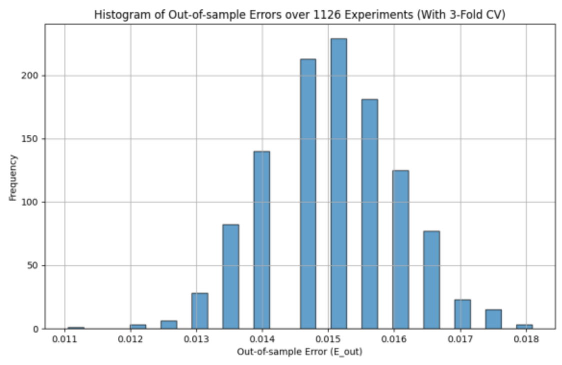


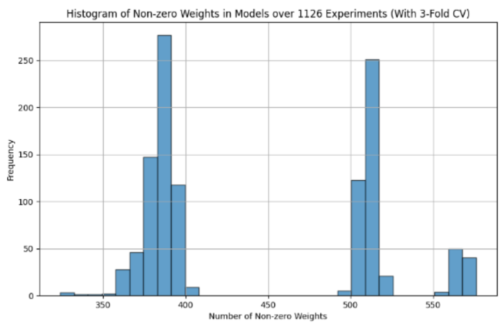
```python
for exp in tqdm(range(experiment_time), desc="進行實驗"):
    np.random.seed(exp)
    indices = np.random.permutation(len(y_train))
    y_train_shuffled = y_train[indices]
    X_train_shuffled = [X_train_dense[i] for i in indices]


    sub_train_size = 8000
    y_sub_train = y_train_shuffled[:sub_train_size]
    X_sub_train = X_train_shuffled[:sub_train_size]
    y_val = y_train_shuffled[sub_train_size:]
    X_val = X_train_shuffled[sub_train_size:]
```

12.

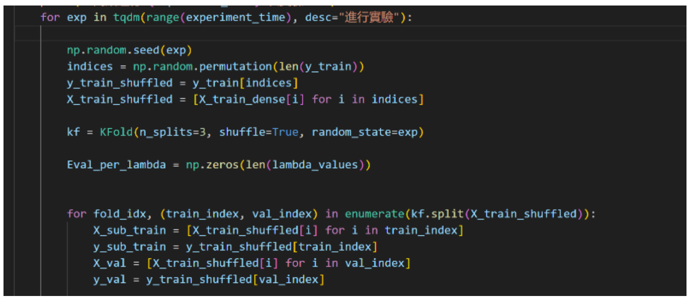
Describe: However, an unexpected observation emerged: despite the noticeable leftward shift in the distribution of non-zero weights, the EoutE_{\text{out}}Eout (out-of-sample error) remained unchanged after performing 3-fold cross-validation. This anomaly suggests that the validation process did not effectively capture patterns similar to those in the test data. In my opinion, this outcome may be attributed to a rare occurrence where the 3-fold validation subsets did not adequately represent the diversity of the entire dataset, leading to suboptimal parameter selection. Consequently, the chosen λ∗\lambda^*λ∗ did not enhance the model's generalization as anticipated. This scenario underscores the inherent variability in cross-validation results and highlights the importance of ensuring that validation folds are sufficiently representative of the overall data distribution to achieve reliable performance improvements.

Result:





Code:



```python
for exp in tqdm(range(experiment_time), desc="進行實驗"):

    np.random.seed(exp)
    indices = np.random.permutation(len(y_train))
    y_train_shuffled = y_train[indices]
    X_train_shuffled = [X_train_dense[i] for i in indices]

    kf = KFold(n_splits=3, shuffle=True, random_state=exp)

    Eval_per_lambda = np.zeros(len(lambda_values))


    for fold_idx, (train_index, val_index) in enumerate(kf.split(X_train_shuffled)):
        X_sub_train = [X_train_shuffled[i] for i in train_index]
        y_sub_train = y_train_shuffled[train_index]
        X_val = [X_train_shuffled[i] for i in val_index]
        y_val = y_train_shuffled[val_index]
```

13.

To handle non smooth L^1 norm, use coordinate descent. In each iterace, we update one coordinate w_i while keeping the others fixed.

for $j \neq i$ $w_j^{(t)}$ are const during update of w_i

$r_n = y_n - \sum_{j \neq i} w_j^T x_{n,j}$

$f(w_i) = \frac{1}{N} \sum (r_n - w_i x_{n,i})^2 + \frac{\lambda_1}{N} |w_i| + \frac{\lambda_2}{N} w_i^2 + C$

define: $a_i = \frac{1}{N} \sum_{n=1}^{N} x_{n,i} r_n$

$b_i = \frac{1}{N} \sum_{n=1}^{N} x_{n,i}^2$

$f(w_i) = A w_i^2 - 2a_i w_i + c|w_i| + C$, $A = b_i + \frac{\lambda_2}{N}$

$c = \frac{\lambda_1}{N}$

minimize eq above:

consider $w_i > 0$

$w_i = 0$

$w_i < 0$

$w_i > 0$:

$\frac{df}{dw_i} = 2Aw_i - 2a_i + c = 0$

$w_i = \frac{a_i - \frac{c}{2}}{A}$

so $w_i > 0$, $a_i > \frac{c}{2}$

$w_i < 0$:

$f(w_i) = Aw_i^2 - 2a_i w_i - Cw_i$

$$\frac{df}{dw_i} = 2Aw_i - 2a_i - c = 0$$

$$w_i = \frac{a_i + \frac{c}{2}}{A}$$

$w_i < 0 : a_i < -\frac{c}{2}$

$w_i = 0$:

subgradient condition :

$0 \in -2a_i + c \cdot s, \quad s \in [-1, 1]$, possible direction

$-2a_i + c \cdot s = 0$

$s = \frac{2a_i}{c}$

$\left|\frac{2a_i}{c}\right| \leq 1 \Rightarrow |a_i| \leq \frac{c}{2}$

set $w_i = 0$ if $|a_i| \leq \frac{c}{2}$

for $w_i^{(t+1)} = \alpha \cdot \max(\beta, 0)$

$\alpha = \text{sign}(a_i) = \begin{cases} +1, & a_i > 0 \\ -1, & a_i < 0 \end{cases}$

$$\beta = \frac{|a_i| - \frac{c}{2}}{A}$$

$$a_i = \frac{1}{N}\sum_{n=1}^{N} x_{n,i}\left(y_n - \sum w_j^{(t)} x_{n,j}\right)$$

$$b_i = \frac{1}{N}\sum_{n=1}^{N} x_{n,i}^2$$

$A = b_i + \frac{\lambda_2}{N}$

$c = \frac{\lambda_1}{N}$

$\alpha = \text{sign}(a_i)$

$$\beta = \frac{|a_i| - \frac{c}{2}}{A} = \left|\frac{1}{N}\sum_{n=1}^{N} x_{n,i}\left(y_n - \sum_{j \neq i} w_j^{(t)} x_{n,j}\right)\right| - \frac{\lambda_1}{2N}$$