UNIVERSITY OF WATERLOO

CHERITON SCHOOL OF COMPUTER SCIENCE

CS 480 - Homework 1

Prepared by:

Jianxiang (Jack) Xu [20658861]

12 February 2021

CS480/680: Introduction to Machine Learning
Homework 2
Due: 11:59 pm, February 12, 2021

Exercise 1: k-Nearest Neighbours (5 pts)

The KNN-dataset provides a 4x4 representation of the handwritten digits 5, 6, and 8. Using the notebook provided, train a KNN classifier using 5-fold cross validation to determine the optimal k-value for this task.

1. (3 pt) Create a graph that shows the average accuracy based on 5-fold cross validation when varying the number of neighbours from 1 to 30.
   Ans: **Answer 1.1**
2. (1 pt) Report the best hyperparameter found by 5-fold cross-validation and the cross-validation accuracy.
   Ans: **Answer 1.2**
3. (1 pt) Report the test accuracy based on the best hyperparameter.
   Ans: **Answer 1.3**

Answer 1.1: KNN classifier with 5-fold cross validation

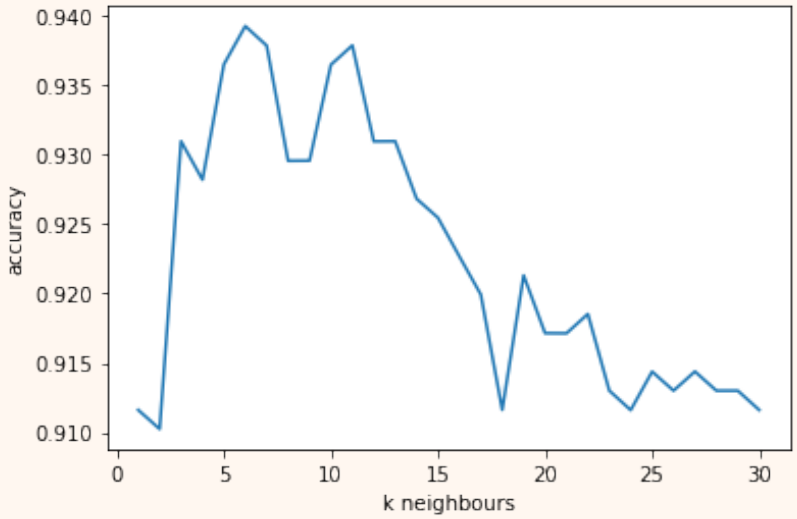

Figure 1-1. KNN Accuracy vs. K-size

See code in the jupyter submission.

Answer 1.2: Best Hyperparameter

From the output below, we may find the best hyperparamter is $k = 5$, with 93.92% cross-validation accuracy and 92.31% test accuracy.

```
best # of neighbours k: 5
best cross validation accuracy: 0.9392432950191572
test accuracy: 0.9230769230769231

```

Answer 1.3: Test Accuracy

As reported in **Answer 1.2**, the final test accuracy is 92.31%.

Exercise 2: Poisson Regression (4 pts)

Recall that in logistic regression we assumed the *binary* label $Y_i \in \{0,1\}$ follows the Bernoulli distribution: $\Pr(Y_i = 1|X_i) = p_i$, where p_i also happens to be the mean. Under the independence assumption we derived the log-likelihood function:

$$\sum_{i=1}^{n} (1-y_i)\log(1-p_i) + y_i \log(p_i). \tag{1}$$

Then, we parameterized the mean parameter p_i through the logit transform:

$$\log \frac{p_i}{1-p_i} = \mathbf{w}^\top \mathbf{x}_i + b, \quad \text{or equivalently} \quad p_i = \frac{1}{1+\exp(-\mathbf{w}^\top \mathbf{x}_i - b)}. \tag{2}$$

Lastly, we found the weight vector $\mathbf{w}$ and b by maximizing the log-likelihood function.

In the following we generalize the above idea to the case where $Y_i \in \mathbb{N}$, i.e., Y_i can take any natural number (for instance, when we are interested in predicting the number of customers or network packages).

1. (1 pt) Naturally, we assume $Y_i \in \mathbb{N}$ follows the Poisson distribution (with mean $\mu_i > 0$):

$$\Pr(Y_i = k|X_i) = \frac{\mu_i^k}{k!}\exp(-\mu_i), \quad k = 0,1,2,\ldots. \tag{3}$$

   Given a dataset $\mathscr{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, what is the log-likelihood function (of μ_i's) given $\mathscr{D}$?
   Ans: **Answer 2.1**
2. (1 pt) Can you give some justification of the parameterization below?

$$\log \mu_i = \mathbf{w}^\top \mathbf{x}_i + b. \tag{4}$$

   Ans: **Answer 2.2**
3. (1 pt) Based on the above, write down the objective function for Poisson regression. Please specify the optimization variables and whether you are maximizing or minimizing. [Constants can be dropped.]
   Ans: **Answer 2.3**
4. (1 pt) Compute the gradient of your objective function above and formulate a gradient algorithm for finding the weight vector $\mathbf{w}$ and b.
   Ans: **Answer 2.4**

Answer 2.1: Log-likelihood Function

Since $Y_i = y_i \in \mathbb{N}$ are i.i.d. Poisson random variables, we may obtain the likelihood function ($L(\mu_i)$):

$$L(\mu_i) = \Pr(Y_1 = y_1, \ldots, Y_n = y_n | X_1 = x_1, \ldots X_n = x_n) \tag{5}$$

$$= \prod_{i=1}^{n} \Pr(Y_i = y_i | X_i) \tag{6}$$

$$= \prod_{i=1}^{n} \frac{\mu_i^{y_i}}{y_i!} \exp(-\mu_i) \tag{7}$$

Since maximizing the likelihood function is equivalent to maximizing its log function, hence we may take log of the entire likelihood function $L(\mu_i)$ from above to compute the log-likelihood:

$$l(\mu_i) = \log(L(\mu_i)) = \log\left(\prod_{i=1}^{n} \frac{\mu_i^{y_i}}{y_i!} \exp(-\mu_i)\right) \tag{8}$$

$$= \sum_{i=1}^{n} \left(\log(\exp(-\mu_i)) + \log\left(\mu_i^{y_i}\right) - \log(y_i!)\right) \tag{9}$$

$$= \sum_{i=1}^{n} \left(-\mu_i + y_i \log(\mu_i) - \log(y_i!)\right) \tag{10}$$

Answer 2.2: Justification of the parameterization

The goal in linear regression is to find some best fit factor $\mathbf{w}$ and bias b that will allow us to map $\mathbf{x}_i$ to some model y. Similarly, we transfer the space into a distribution model, where the output y_i is sampled from a normal distribution (Poisson distribution in this case).

Hence,

$$y \sim \mathcal{N}(y; \mu, \sigma^2), \quad \text{where } \mu_i = \mathbf{w}^\top \mathbf{x}_i + b. \tag{11}$$

However, in our case, the distribution is Poisson and $\mu_i > 0$. Since $\mathbf{w}^\top \mathbf{x}_i + b$ has to be mapped to $(-\infty, \infty)$, $log(\mu_i)$ is a reasonable mapping strategy to transfer $\mu_i > 0$ to $(-\infty, \infty)$.

Hence,

$$y \sim Po(y; \mu, \sigma^2), \quad \text{where } \log(\mu_i) = \mathbf{w}^\top \mathbf{x}_i + b. \tag{12}$$

Answer 2.3: Objective Function for Poisson Regression

From Equation (4) (justified in Answer 2.2), we may now complete the log-likelihood function as:

$$l(\mu_i) = \sum_{i=1}^{n} (-\mu_i + y_i \log(\mu_i) - \log(y_i!)) \tag{13}$$

$$= \sum_{i=1}^{n} \left(-e^{(\mathbf{w}^\top \mathbf{x}_i + b)} + y_i(\mathbf{w}^\top \mathbf{x}_i + b) - \log(y_i!) \right) \tag{14}$$

Since $\log(y_i!)$ term in the equation could be regarded as constant, since it is provided in training data, we may drop it.

$$\because \quad \frac{\partial \sum_{i=1}^{n} -\log(y_i!)}{\partial \mu_i} = 0 \tag{15}$$

$$\therefore \quad l(w,b) = \sum_{i=1}^{n} \left(-e^{(\mathbf{w}^\top \mathbf{x}_i + b)} + y_i(\mathbf{w}^\top \mathbf{x}_i + b)) \right) \tag{16}$$

There is no closed form solution to maximizing the log-likelihood function, but we can convert it to minimization and apply gradient descent.

$$\max_{w,b} l(w,b) = \min_{w,b} -l(w,b) \tag{17}$$

$$= \min_{w,b} \sum_{i=1}^{n} \left(e^{(\mathbf{w}^\top \mathbf{x}_i + b)} - y_i(\mathbf{w}^\top \mathbf{x}_i + b)) \right) \tag{18}$$

Objective Function

$$\min_{w,b} \sum_{i=1}^{n} \left(e^{(\mathbf{w}^\top \mathbf{x}_i + b)} - y_i(\mathbf{w}^\top \mathbf{x}_i + b)) \right) \equiv \min_{\mu_i} \sum_{i=1}^{n} (\mu_i - y_i \log(\mu_i)) \tag{19}$$

Answer 2.4: Gradient Algorithm

Let's find the gradient step of the objective function (Equation (19)):

$$\frac{\partial}{\partial w}\sum_{i=1}^{n}\left(e^{(\mathbf{w}^\top \mathbf{x}_i+b)}-y_i(\mathbf{w}^\top \mathbf{x}_i+b)\right)=\sum_{i=1}^{n}x_i\,e^{w^T x_i+b}-\sum_{i=1}^{n}y_i x_i \tag{20}$$

$$\frac{\partial}{\partial b}\sum_{i=1}^{n}\left(e^{(\mathbf{w}^\top \mathbf{x}_i+b)}-y_i(\mathbf{w}^\top \mathbf{x}_i+b)\right)=\sum_{i=1}^{n}e^{w^T x_i+b}-\sum_{i=1}^{n}y_i \tag{21}$$

$$\tag{22}$$

Hence, the gradient algorithm for weight vector $\mathbf{w}$ and b:

Gradient Descent Update Function

$$\mathbf{w}\leftarrow \mathbf{w}-\eta\left(\sum_{i=1}^{n}x_i\,e^{w^T x_i+b}-\sum_{i=1}^{n}y_i x_i\right) \tag{23}$$

$$b\leftarrow b-\eta\left(\sum_{i=1}^{n}e^{w^T x_i+b}-\sum_{i=1}^{n}y_i\right) \tag{24}$$

Exercise 3: Fun with Classification (5 pts)

For this problem, you are allowed to use `statsmodels` and `sklearn` as directed.

1. (2 pts) Run logistic regression, SVM with ℓ_2 regularization with parameter 1 (soft-margin SVM), and SVM with regularization parameter `float('inf')` (hard-margin SVM) on Mystery Dataset A (note that there is only a training dataset and no test dataset). Use `Logit` from `statsmodels` and `SVC` (with linear kernel) from `sklearn`. One of these methods will not work -- mathematically explain why (that is, give an explanation which is more informative than any error message you encounter). How could the associated problems be remedied? Discuss similarities and differences between the solution obtained via these three methods.
   Ans: **Answer 3.1**
2. (3 pts) Take your solution for the soft-margin SVM from the previous part. For each point in the dataset, take its inner product with the produced coefficient vector, and scale the result by the sign of each point's label (replace 0's with -1's). How many of these values are ≤ 1? [Be sure you're getting all of them -- there may be numerical precision issues, so if in doubt, err on the side of counting a point.] Based on your answer to these questions, sketch a 2D caricature of what the points and the hyperplane defined by the SVM solution look like. Write the parameter vector solution to the SVM problem as a linear combination of some points in your dataset. How many points did you require? [You may find built-in functions useful for this purpose.]
   Compute the solution for the same three methods on Mystery Dataset B. You will again run into issues with one of them -- explain why. [The answer is likely to be simpler than last time.] Find a way to write the parameter vector solution to the SVM problem as a linear combination of some points in your dataset -- do not report the solution itself, but report how many points you used, and how you arrived at this answer. Compare the empirical prediction accuracy (i.e., using 0-1 loss) of the successfully-trained classifiers on the test set.
   Ans: **Answer 3.2**

Answer 3.1: Logit and SVM

It appears `'Logit'` from `'statsmodels'` terminated with error message of "PerfectSeparationError: Perfect separation detected, results not available", whereas soft-margin and hard-margin SVM are both capable to perform the fitting.

This error happens when a linear combination of the predictors yield a perfect prediction of the response variable or the binary outcome in this case. Simply, $\mathbf{x}_i$ predicts y_i perfectly. If we perform l2 norm on every single x dataset, we may realize every two alternating points with opposing labels are equal distance from the origin, and we may suspect, there exists such plane through the origin that can perfectly separate the dataset. Hence, the maximum likelihood on x_i would approach to infinity, therefore, we cannot find a suitable coefficient to maximize the conditional likelihood.

A solution to resolve this is to utilize a form of penalized regression to penalize the likelihood (the whole point of SVM).

The final trained results are shown in Answer 3.1, and we observed that both hard and soft margin SVM provides the exact same hyperplane through the origin as expected. Whereas, the logistic regression failed to compute.

```
w_hard =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
   1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
   1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
  -9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
  -1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
  -6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
   7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
   1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
  -1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
   1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
  -1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
   1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
   2.91490606e-01 -3.25700173e-02]]
b_hard =  [-0.]
w_soft =  [[-1.21665426e-01  3.62093715e-04 -8.14547914e-02 -1.30461544e-01
   1.03679744e-01 -1.52182440e-01  1.04733167e-01  2.37085565e-02
   1.88904300e-01  4.14259231e-01 -5.74918926e-02  2.53091455e-02
  -9.57967331e-02  2.78213946e-01  1.26402871e-02  4.24952941e-02
  -1.08058239e-01 -3.17898534e-02  3.32288614e-03  1.71401564e-01
  -6.93185355e-02 -2.85727264e-01 -1.11431511e-01 -2.81551300e-02
   7.14262962e-02  1.98803054e-01 -1.89996218e-01  9.63349060e-02
   1.20110591e-01 -5.53355480e-02 -4.29452659e-03 -1.10216493e-01
  -1.57449276e-01  3.64106600e-02 -4.35440021e-02 -1.42084333e-01
   1.30750877e-01  2.68330435e-02  6.77532548e-02 -3.01716444e-01
  -1.22263636e-02  2.15382941e-01  1.26537049e-01  2.24429271e-02
   1.34745230e-01  3.23138170e-02  1.47580798e-01  5.67770933e-02
   2.91490606e-01 -3.25700173e-02]]
b_soft =  [-0.]
|w_hard|_2 - |w_soft|_2 =  0.0
```

Answer 3.2: Conclusion

How many of these values are ≤ 1?

All the values (2000/2000) are ≤ 1

2D caricature sketch of SVM solution

Parameter vector solution

2 points needed.

$y = -0.5x_{1999} + 0.5x_{1998}$

Mystery Dataset B

We ran into problem (infinity loop) with Hard-margin SVM, while Soft-margin SVM and Logistic regression has been successfully fitted. This is caused by the dataset that is inseparable with hard-margin, hence the program ran into infinity loop trying to separate the dataset that is not separable.

192 points needed.

To compare the SVM and logistic regression performance, the empirical metric accuracy is calculated based on test data: accuracy $= \frac{\text{\# right predictions}}{\text{\# of test data}}$

We find:

Soft-SVM Test Performance (Empirical Accuracy): 97.15%

Logit Test Performance (Empirical Accuracy): 96.95%

Hence, the Soft-SVM has better performance.

Exercise 4: Support Vector Regression (8 pts)

Let us consider support vector regression:

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n} \max\{|y_i - (\mathbf{w}^\top \mathbf{x}_i + b)| - \varepsilon, 0\}, \tag{25}$$

where $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and $\|\mathbf{w}\|_2 := \sqrt{\sum_{j=1}^d w_j^2}$ is the Euclidean norm.

1. (2 pts) Derive the Lagrangian dual of the support vector regression loss function (25). Please include intermediate steps so that you can get partial credits.
   Ans: **Answer 4.1**

   In the following you will complete and implement the following gradient algorithm for solving support vector regression in Equation (25):

   Algorithm 1.1: GD for SVR.

   Input: $X \in \mathbb{R}^{n\times d}$, $\mathbf{y} \in \mathbb{R}^n$, $\mathbf{w} = \mathbf{0}_d$, $b = 0$, max_pass $\in \mathbb{N}$, step size η
   Output: $\mathbf{w}, b$

   ```
   1 for t = 1,2,...,max_pass do
   2     for i = 1,2,...,n do
   3         choose step size η
   4         if |y_i − (⟨x_i, w⟩ + b)| ≥ ε then
   5             w ←                                   // x_i is the i-th row of X
   6             b ←
   7         w ←                                       // proximal step
   ```

   Note that this differs a bit from what you've seen so far, in terms of gradient descent. Rather than taking steps based on the entire loss function, we instead take a step based on the unregularized loss, and then perform a projection step based on the regularizer.
2. (2 pts) Compute the gradient w.r.t. $\mathbf{w}$ and b for each second term in Equation (25). Note that in places where the function is non-differentiable, you might have to compute a sub-gradient.

$$C\sum_{i=1}^{n} \max\{|y_i - (\mathbf{w}^\top \mathbf{x}_i + b)| - \varepsilon, 0\} \tag{26}$$

   Ans: **Answer 4.2**
3. (1 pt) Find the closed-form solution of the following proximal step:

$$\mathsf{P}^\eta(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{z}} \frac{1}{2\eta}\|\mathbf{z} - \mathbf{w}\|_2^2 + \frac{1}{2}\|\mathbf{z}\|_2^2 \tag{27}$$

   Ans: **Answer 4.3**
4. (3 pts) Implement Algorithm 1.1. You should use Ex 1.3 to complete lines 5-6, and Ex 1.4 for line 7. Run it on Mystery Dataset C (this is Mystery Dataset A from Assignment 1, but reused), and report your training error, training loss, and test error. Use $C = 1$ and $\varepsilon = 0.1$.
   Ans: **Answer 4.4**

Answer 4.1: Lagrangian Dual Derivation

Let's substitute the regulation (second) term with a complement slack variables (ξ_i, ξ_i') as suggested in lecture to account the delta error outside the ε margin.

$$\textbf{Condition:} \tag{28}$$
$$y_i - \hat{y}_i \leq \xi_i + \varepsilon \tag{29}$$
$$\hat{y}_i - y_i \leq \xi_i' + \varepsilon \tag{30}$$
$$\text{where } \xi_i, \xi_i' > 0, \quad \hat{y}_i = \mathbf{w}^T \mathbf{x}_i + b \tag{31}$$

Hence, we may simplify the formulation Equation (25) into:

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \tag{32}$$
$$\textbf{where} \tag{33}$$
$$y_i - \hat{y} \leq \xi_i + \varepsilon \tag{34}$$
$$\hat{y} - y_i \leq \xi_i' + \varepsilon \tag{35}$$
$$\text{where } \xi_i, \xi_i' > 0, \quad \hat{y} = \mathbf{w}^T \mathbf{x}_i + b \tag{36}$$

We may now apply Lagrangian with complement (α, α') and (β, β') multipliers:

$$\min_{w,b,\xi,\xi'} \max_{\alpha,\alpha'\geq 0 \beta,\beta'\leq 0} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') + \sum_{i=1}^{n}\alpha_i(y_i - \hat{y}_i - \varepsilon - \xi_i) + \sum_{i=1}^{n}\alpha_i'(-y_i + \hat{y}_i - \varepsilon - \xi_i') - \sum_{i=1}^{n}\beta_i\xi_i - \sum_{i=1}^{n}\beta_i'\xi_i' \tag{37}$$

Swap min and max:

$$\max_{\alpha,\alpha'\geq 0 \quad \beta,\beta'\leq 0} \min_{w,b,\xi,\xi'} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') + \sum_{i=1}^{n}\alpha_i(y_i - (\mathbf{w}^T\mathbf{x}_i + b) - \varepsilon - \xi_i) + \sum_{i=1}^{n}\alpha_i'(-y_i + (\mathbf{w}^T\mathbf{x}_i + b) - \varepsilon - \xi_i') - \sum_{i=1}^{n}\beta_i\xi_i - \sum_{i=1}^{n}\beta_i'\xi_i' \tag{38}$$

Take derivative $w.r.t. w, b, \xi_i, \xi_i'$:

$$\frac{\partial}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i \mathbf{x}_i + \sum_{i=1}^{n} \alpha_i' \mathbf{x}_i \tag{39}$$

$$\frac{\partial}{\partial b} = -\sum_{i=1}^{n} \alpha_i + \sum_{i=1}^{n} \alpha_i' \tag{40}$$

$$\frac{\partial}{\partial \xi_i} = nC - n\alpha_i - n\beta_i \tag{41}$$

$$\frac{\partial}{\partial \xi_i'} = nC - n\alpha_i' - n\beta_i' \tag{42}$$

, and $\equiv 0$:

$$\sum_{i=1}^{n} (\alpha_i - \alpha_i') \mathbf{x}_i = \mathbf{w} \tag{43}$$

$$\sum_{i=1}^{n} (\alpha_i' - \alpha_i) = 0 \tag{44}$$

$$\alpha_i + \beta_i = \alpha_i' + \beta_i' = C \tag{45}$$

Substitute back to Lagrangian (Equation (38)):

$$\begin{aligned} \max_{\substack{\alpha,\alpha' \geq 0 \\ \beta,\beta' \leq 0}} \quad & \frac{1}{2} \| \sum_{i=1}^{n} (\alpha_i - \alpha_i') \mathbf{x}_i \|_2^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i') + \sum_{i=1}^{n} \alpha_i (y_i - (\mathbf{w}^T \mathbf{x}_i + b) - \varepsilon - \xi_i) \\ & + \sum_{i=1}^{n} \alpha_i' (-y_i + (\mathbf{w}^T \mathbf{x}_i + b) - \varepsilon - \xi_i') - \sum_{i=1}^{n} \beta_i \xi_i - \sum_{i=1}^{n} \beta_i' \xi_i' \end{aligned} \tag{46}$$

$$\begin{aligned} \max_{\substack{\alpha,\alpha' \geq 0 \\ \beta,\beta' \leq 0}} \quad & \frac{1}{2} \| \sum_{i=1}^{n} (\alpha_i - \alpha_i') \mathbf{x}_i \|_2^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i') + \sum_{i=1}^{n} \alpha_i y_i - \cancel{\sum_{i=1}^{n} \alpha_i (\mathbf{w}^T \mathbf{x}_i + b)} - \sum_{i=1}^{n} \alpha_i \varepsilon - \sum_{i=1}^{n} \alpha_i \xi_i \\ & - \sum_{i=1}^{n} \alpha_i' y_i + \cancel{\sum_{i=1}^{n} \alpha_i' (\mathbf{w}^T \mathbf{x}_i + b)} - \sum_{i=1}^{n} \alpha_i' \varepsilon - \sum_{i=1}^{n} \alpha_i' \xi_i' - \sum_{i=1}^{n} \beta_i \xi_i - \sum_{i=1}^{n} \beta_i' \xi_i' \end{aligned} \tag{47}$$

$$\begin{aligned} \max_{\substack{\alpha,\alpha' \geq 0 \\ \beta,\beta' \leq 0}} \quad & \frac{1}{2} \| \sum_{i=1}^{n} (\alpha_i - \alpha_i') \mathbf{x}_i \|_2^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i') - \sum_{i=1}^{n} (\alpha_i + \alpha_i') \varepsilon + \sum_{i=1}^{n} (\alpha_i' - \alpha_i) y_i \\ & - \cancel{\sum_{i=1}^{n} (\alpha_i + \beta_i) \xi_i} - \cancel{\sum_{i=1}^{n} (\alpha_i' + \beta_i') \xi_i'} \end{aligned} \tag{48}$$

$$\max_{\alpha,\alpha' \geq 0} \quad \frac{1}{2} \| \sum_{i=1}^{n} (\alpha_i - \alpha_i') \mathbf{x}_i \|_2^2 - \sum_{i=1}^{n} (\alpha_i + \alpha_i') \varepsilon + \sum_{i=1}^{n} (\alpha_i' - \alpha_i) y_i \tag{49}$$

Convert to minimization problem:

The Dual

$$\min_{C \geq \alpha, \alpha' \geq 0} \quad \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i - \alpha_i')(\alpha_j - \alpha_j') \mathbf{x}_i^T \mathbf{x}_j + \varepsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i') - \sum_{i=1}^{n} (\alpha_i' - \alpha_i) y_i \tag{50}$$

$$\textbf{s.t.} \quad \sum_{i=1}^{n} (\alpha_i' - \alpha_i) = 0 \tag{51}$$

Answer 4.2: Gradient

$$\frac{\partial}{\partial \mathbf{w}} = \sum_{i=1}^{n} \begin{cases} 0 & |y_i - \hat{y}_i| \leq \varepsilon \\ -Cx_i & y_i > \hat{y}_i, |y_i - \hat{y}_i| > \varepsilon \\ Cx_i & \text{otherwise} \end{cases} \tag{52}$$

$$\frac{\partial}{\partial b} = \sum_{i=1}^{n} \begin{cases} 0 & |y_i - \hat{y}_i| \leq \varepsilon \\ -C & y_i > \hat{y}_i, |y_i - \hat{y}_i| > \varepsilon \\ C & \text{otherwise} \end{cases} \tag{53}$$

$$\text{where} \qquad \hat{y}_i = \mathbf{w}^T \mathbf{x}_i + b \tag{54}$$

Answer 4.3: Closed Form Solution of the Proximal Step

$$\frac{\partial}{\partial z}\mathsf{P}^{\eta}(\mathbf{w}) = \frac{\partial}{\partial z}\left(\frac{1}{2\eta}\|\mathbf{z}-\mathbf{w}\|_2^2 + \frac{1}{2}\|\mathbf{z}\|_2^2\right) \tag{55}$$

$$0 = \frac{1}{\eta}(\mathbf{z}-\mathbf{w}) + \mathbf{z} \tag{56}$$

$$z = \frac{w}{1+\eta} \tag{57}$$

Closed Form Solution

$$z = \frac{w}{1+\eta} \tag{58}$$

Answer 4.4: Implementation

Algorithm 2.2: Seudo implementation of GD for SVR.

Input: $X \in \mathbb{R}^{n \times d}$, $\mathbf{y} \in \mathbb{R}^n$, $\mathbf{w} = \mathbf{0}_d$, $b = 0$, max_pass $\in \mathbb{N}$, step size η

Output: $\mathbf{w}, b$

for $t = 1, 2, \ldots,$ max_pass **do**
  for $i = 1, 2, \ldots, n$ **do**
    choose step size η
    if $|y_i - (\langle \mathbf{x}_i, \mathbf{w} \rangle + b)| \geq \varepsilon$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} - \eta C x_i \times \mathbf{sgn}(< \mathbf{w}, \mathbf{x} > + b - y_i)$ // $\mathbf{x}_i$ is the i-th row of X
      $b \leftarrow b - \eta C \times \mathbf{sgn}(< \mathbf{w}, \mathbf{x} > + b - y_i)$
    $\mathbf{w} \leftarrow \frac{w}{1+\eta}$ // proximal step

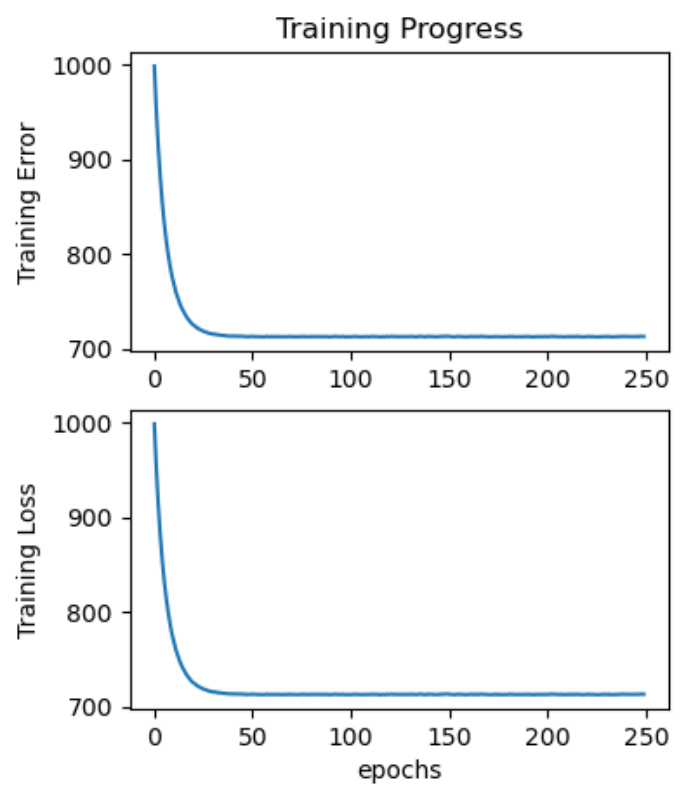

Figure 4-2. Training Progress

```
> [result   ] T: 249 | Training Error: 712.84065 | Training Loss: 713.37650 | Test Error: 891.05659

```