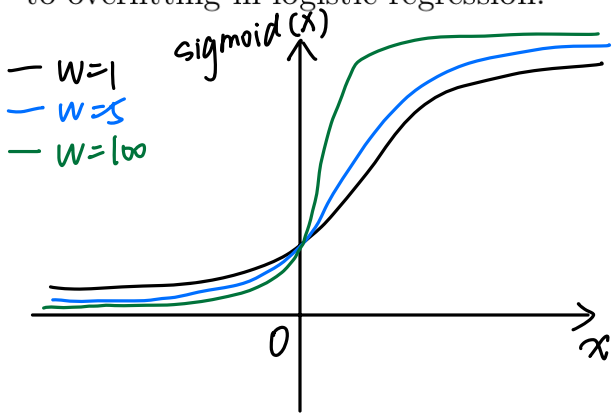
Jingquan Li 11901014S DDA3020 Homework 2

1 Written Problems (50 pts.)

1.1 Problem: Overfitting and Regularized Logisitc Regression (5 pts.)

1) (2 pts.) Plot the sigmoid function $1/(1+\exp^{-wX})$ for increasing weights $w \in \{1, 5, 100\}$ with $X \in \mathbb{R}$. A qualitative sketch will suffice. Utilize these plots to explain why large weights can lead to overfitting in logistic regression.



plots analyze:

(1) For $w=1$, the sigmoid function has a gentle S-shape, indicating a relatively smooth transition from 0 to 1.

(2) For $w=5$, the sigmoid function becomes steeper, resulting in a faster transition from 0 to 1 as x changes.

(3) For $w=100$, the sigmoid function approaches a step function, with a very sharp transition from 0 to 1 around $X=0$.

Explain why large weights can lead to overfitting in logistic regression:

- When the weight w is very large, the sigmoid function becomes highly sensitive to small changes in the input X. This means that even a slight change in the input can cause a drastic change in the output probability.
- In logistic regression, large weights can lead to complex decision boundaries that closely fit the training data. While this might reduce training error, it can result in poor generalization to unseen data. Large weights can lead to overfitting by making the model too sensitive to the training data.
- Consequently, the model becomes less robust and may perform poorly on new, unseen data because it has essentially memorized the training set rather than learning the underlying patterns that generalize well.

2) (3 pts.) To mitigate overfitting, it is preferable to have smaller weights. To accomplish this, rather than utilizing maximum conditional likelihood estimation M(C)LE for logistic regression:

$$\max_{w_0,\cdots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d), \tag{1}$$

we can consider maximum conditional a posterior M(C)AP estimation:

$$\max_{w_0,\cdots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d) P(w_0, \cdots, w_d), \tag{2}$$

where $P(w_0, \cdots, w_d)$ is a prior on the weights.
Given a standard Gaussian prior $\mathcal{N}(0, \mathbf{I})$ for the weight vector w, please derive the **gradient ascent** update rules for the weights and explain why M(C)AP can address overfitting issue.

Solution: $L(w) = \log\left(p(w) \prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d)\right)$

$\because w \sim \mathcal{N}(0, 1) \quad \therefore p(w) = \prod_{k=0}^{d} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{w_k^2}{2}\right)$

$\therefore w^* = \arg\max_w L(w) = \arg\max_w \left[\sum_{i=1}^{n} \log(P(Y_i|X_i, w_0, \cdots, w_d)) - \sum_{k=0}^{d} \frac{w_k^2}{2}\right]$

$$\frac{\partial L(w)}{\partial w_i} = \frac{\partial}{\partial w_i} \log p(w) + \frac{\partial}{\partial w_i} \log\left(\prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d)\right)$$

$$= -w_i + \frac{\partial}{\partial w_i} \log\left(\prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d)\right)$$

$$= -w_i + \frac{\partial}{\partial w_i} \left(\sum_{i=1}^{n} Y_i \log(f_w(x_i)) + (1 - Y_i) \log(1 - f_w(x_i))\right)$$

$\because \frac{\partial \log(f_w|x_i)}{\partial w} = \frac{1}{f_w(x_i)} \cdot \frac{x_i \exp(-w^T x_i)}{f_w(x_i)^{-2}} = f_w(x_i) \cdot x_i \cdot \exp(-w^T x_i) = \frac{x_i \exp(-w^T x_i)}{1 + \exp(-w^T x_i)}$

$\frac{\partial \log(1 - f_w(x_i))}{\partial w} = \frac{1}{1 - f_w(x_i)} \cdot \frac{-x_i \exp(-w^T x_i)}{f_w(x_i)^{-2}} = \frac{-x_i}{1 + \exp(-w^T x_i)}$

$\therefore \frac{\partial}{\partial w_i}(Y_i \log f_w(x_i) + (1 - Y_i) \log(1 - f_w|x_i)) = Y_i x_i - x_i \frac{1}{1 + \exp(-w^T x_i)}$

$\therefore \frac{\partial L(w)}{\partial w} = -w + \sum_{i=1}^{n} \left(Y_i x_i - \frac{x_i}{1 + \exp(-w^T x_i)}\right)$

$$= -w + \sum_{i=1}^{n} \left[x_i \left(Y_i - \frac{1}{1 + \exp(-w^T x_i)}\right)\right)$$

$\therefore$ gradient descent rule: $w \leftarrow w - \alpha\left[-w + \sum_{i=1}^{n} x_i \left(Y_i - \frac{1}{1 + \exp(-w^T x_i)}\right)\right]$

Explanation: MCAP addresses overfitting issue by maximizing the objective function, which incorperates both likelihood term and prior term. It penalizes large weights and helps in prevent overfitting by discouraging the model from fitting noise in the data too closely

1.2 Problem: Multi-class Logisitc Regression (16 pts.)

In this question, we will derive the "multi-class logistic regression" algorithm, assuming the dataset $\mathcal{D}$ is d-dimensional (with d features) and contains n entries.
Given a training set $\{(x^i, y^i)|i = 1, \cdots, n\}$ where $x^i \in \mathbb{R}^{d+1}$ is a feature vector and $y^i \in \mathbb{R}^k$ is a one-hot encoded binary vector with k entries representing classes. In a one-hot vector, the corresponding class label is 1, and all other entries are 0s. For example, if the label of x^i is 3, then the corresponding y^i should be $[0, 0, 1, \cdots, 0] \in \mathbb{R}^k$.

We aim to determine the parameters $\hat{w} \in \mathbb{R}^{k \times (d+1)}$ (representing one weight vector for each class) that maximize the likelihood for the training set, given a parametric model in the following form:

$$p(y_c^i = 1|x^i; w) = \frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}. \tag{3}$$

Note that $\frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}$ is always between 0 and 1, and $\sum_c p(y_c^i = 1|x^i; w)$ is always 1, which are desired properties of a probability distribution. Therefore, $\frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}$ is also known as the softmax function.

Since we know the probability sums to 1, we don't care about predicting the probability of the last (k^{th}) class, since we can calculate $p(y_k^i = 1|x^i; w)$ by:

$$p(y_k^i = 1|x^i; w) = 1 - \frac{\sum_{c'=1}^{k-1} \exp(w_{c'}^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}. \tag{4}$$

1) (4 pts.) Show the equivalence between the Eq. (5) and Eq. (6). Provide a short justification for why each line follows from the previous one in your derivation. (This is how we store less weights by making use of the fact that the probabilities sum to 1).

$$p(y_c^i = 1|x^i; w) = \frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)} \tag{5}$$

$$= \begin{cases} \frac{\exp(w_c^\top x^i)}{1 + \sum_{c'=1}^{k-1} \exp(w_c^\top x^i)}, & \text{if } c < k \\ \frac{1}{1 + \sum_{c'=1}^{k-1} \exp(w_c^\top x^i)}, & \text{if } c = k \end{cases} \tag{6}$$

Solution: When adding the same value Δ to all the weights w_i, the softmax function for any particular class i becomes: $\frac{e^{(w_i+\Delta)}}{\sum_{j=1}^{n} e^{(w_j+\Delta)}}$

$= \frac{e^{w_i} \cdot e^{\Delta}}{\sum_{j=1}^{n} e^{w_j} e^{\Delta}} = \frac{e^{w_i}}{\sum_{j=1}^{n} e^{w_j}}$.

Therefore, shifting all weights by the same amount does not alter the final probabilities produced by the softmax function.

Shift the kth weight to 0:

$p(y_c^i = 1|x^i; w) = \frac{\exp(w_c^T x^i)}{\sum_{c'} \exp(w_{c'}^T x^i)} = \frac{\exp((w_{c'} - w_k)^T x^i)}{\sum_{c'} \exp((w_{c'} - w_k)^T x^i)} = \frac{\exp((w_{c'} - w_k)^T x^i)}{1 + \sum_{c'=1}^{k-1} \exp((w_{c'} - w_k)^T x^i)}$

$= \begin{cases} \frac{\exp(w_c^T x^i)}{1 + \sum_{c'=1}^{k-1} \exp(w_{c'}^T x^i)}, & \text{if } c < k \\ \frac{1}{1 + \sum_{c'=1}^{k-1} \exp(w_{c'}^T x^i)}, & \text{if } c = k \end{cases}$

2) (4 pts.) Derive the conditional log likelihood for softmax regression. For the sake of simplicity, we only consider Eq. (5) as $p(y_c^j|x^j, w)$. Show the equivalence between the Eq. (7) and Eq. (8). Provide a short justification for why each line follows from the previous one in your derivation.

$$\mathcal{L}(w) \equiv \ln \prod_{j=1}^{n} p(y_c^j|x^j, w) \text{ [here } c \text{ is the true class of } x^j] \tag{7}$$

$$= \sum_{j=1}^{n} \sum_{c=1}^{k} \left[y_c^j (w_c^\top x^j) - y_c^j \ln \left(\sum_{c'} \exp(w_{c'}^\top x^j) \right) \right]. \tag{8}$$

Proof: $\mathcal{L}(w) = \ln \prod_{j=1}^{n} p(y_c^j|x^j, w) = \sum_{j=1}^{n} \sum_{c=1}^{k} y_c^j \ln p(y_c^j|x^j, w)$

$= \sum_{j=1}^{n} \sum_{c=1}^{k} \left[y_c^j \ln \frac{\exp(w_c^T x^j)}{\sum_{c'} \exp(w_{c'}^T x^j)} \right] = \sum_{j=1}^{n} \sum_{c=1}^{k} \left[y_c^j (w_c^T x^j) - y_c^j \ln \left(\sum_{c'} \exp(w_{c'}^T x^j) \right) \right]$

3) (4 pts.) Next, we will derive the gradient of the previous expression with respect to the c^{th} class of the weight matrix w_c, i.e., $\frac{\partial \mathcal{L}(w)}{\partial w_c}$, where $\mathcal{L}(w)$ denotes the log likelihood from Eq. (8). We will perform a few steps of the derivation, and then ask you to do one step at the end. If we task the derivative of Eq. (8) with respect to w_c, we get the following experssion:

$$\nabla_{w_c}\mathcal{L}(w) = \nabla_{w_c}\sum_{j=1}^{n}\sum_{c=1}^{k}\left[y_c^j(w_c^\top x^j) - y_c^j \ln\left(\sum_{c'}\exp(w_{c'}^\top x^j)\right)\right]. \tag{9}$$

The blue expression is linear in w_c, so it can be simplified to $\sum_{j=1}^{n} y_c^j x^j$. For the red expression,first we consider a fixed $j \in [1, n]$. Use the chain rule to verify that

$$\nabla_{w_c}\sum_{c=1}^{k} y_c^j \ln\left(\sum_{c'}\exp(w_{c'}^\top x^j)\right) \tag{10}$$

$$= \frac{\exp(w_c^\top x^j)}{\sum_{c'}\exp(w_{c'}^\top x^j)}x^j. \tag{11}$$

Proof: $\nabla_{w_c}\sum_{c=1}^{k} y_c^j \ln\left(\sum_{c'}\exp(w_{c'}^\top x^j)\right) = \nabla_{w_c}\left(\sum_{c=1}^{k} y_{c'}^j\right)\ln\left(\sum_{c'}\exp(w_{c'}^\top x^j)\right)$

$= \nabla_{w_c}\ln\left(\sum_{c'}\exp(w_{c'}^\top x^j)\right) = \frac{\exp(w_c^\top x^j)}{\sum_{c'}\exp(w_{c'}^\top x^j)}x^j.$

4) (2 pts.) Now use 11 (and the previous discussion) to show that overall, Eq. (9), i.e., $\nabla_{w_c}\mathcal{L}(w)$, is equal to

$$\nabla_{w_c}\mathcal{L}(w) = \sum_{j=1}^{n} x^j(y_c^j - p(y_c^j = 1|x^j; w)). \tag{12}$$

Proof: From question (3) and the previous discussion,

$$\nabla_{w_c}\mathcal{L}(w) = \sum_{j=1}^{n} y_c^j x^j - \sum_{j=1}^{n}\frac{\exp(w_c^\top x^j)}{\sum_{c'}\exp(w_{c'}^\top x^j)}x^j)$$

$$= \sum_{j=1}^{n} x^j(y_c^j - p(y_c^j = 1|x^j; w))$$

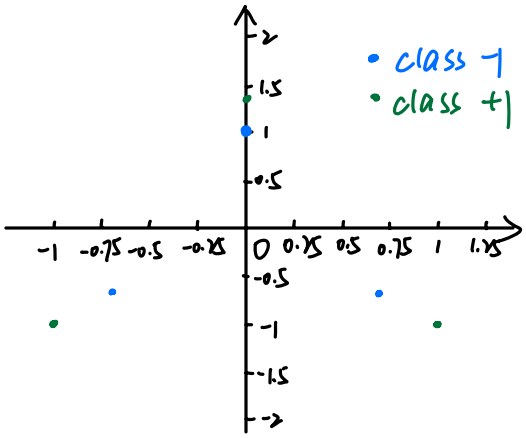
5) (2 pts.) Since the log likelihood is concave, it is easy to optimize using gradient ascent. Derive the update rule for **gradient ascent** with respect to learning rate for w_c w.r.t. η, y^j, x^j, and $p(y^j = 1|x^j; w^{(t)})$. Feel free to index into the vectors using subscripts.

Solution: $w_c^{t+1} = w_c^t + \eta\sum_{j=1}^{n} x^j(y_c^j - p(y_c^j = 1|x^j; w))$

1.3 Problem: Support Vector Machine 1 (6 pts.)

Given a binary data set:
Class -1: $\{(0,1), (\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}), (-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})\}$. Class +1: $\{(0,\sqrt{2}), (1,-1), (-1,-1)\}$.
1) (3 pts.) Could you find an SVM classifier (without slack variable) for this dataset? Please explain your reasoning, possibly with the help of a plot sketch.
2) (3 pts.) Use SVM by expanding the original feature vector $x = [x_1, x_2]$ to $x = [x_1^2, x_2^2]$, find the svm of this given data set and predict the label of $(-\frac{1}{2}, \sqrt{2})$.

(1) Solution: Since we don't have slack variables and we're looking for a linear decision boundary, we want to find the hyperplane that maximally separates the two classes.



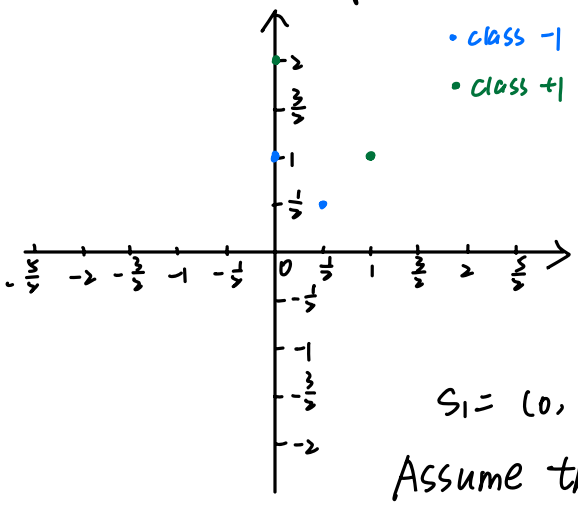
The plot shows that the data points are not linearly separable in the original feature space. Therefore, I cannot find an SVM classifier (without slack variable) for this dataset

(2) Expanding the original feature vector:

For class −1: $(0,1)\to(0,1)$, $(\frac{1}{\sqrt{2}},-\frac{1}{\sqrt{2}})\to(\frac{1}{2},\frac{1}{2})$, $(-\frac{1}{\sqrt{2}},-\frac{1}{\sqrt{2}})\to(\frac{1}{2},\frac{1}{2})$

For class +1: $(0,\sqrt{2})\to(0,2)$, $(1,-1)\to(1,1)$, $(-1,-1)\to(1,1)$

Plot the transformed data points:



We can see that, in this transformed space, the classes are linearly separable. We can draw a hyperplane to separate them.

It can be observed that the support vectors are $(0,1)$, $(\frac{1}{2},\frac{1}{2})$, $(0,2)$, $(1,1)$

$s_1=(0,1)^T$, $s_2=(\frac{1}{2},\frac{1}{2})^T$, $s_3=(0,2)^T$, $s_4=(1,1)^T$

Assume the weight vector as $w=(w_1,w_2)^T$ and the bias term as b.

- For class −1: $\begin{cases} w_1\cdot 0+w_2\cdot 1+b=-1 \\ w_1\cdot\frac{1}{2}+w_2\cdot\frac{1}{2}+b=-1 \end{cases}$ For class +1: $\begin{cases} w_1\cdot 0+w_2\cdot 2+b=1 \\ w_1\cdot 1+w_2\cdot 1+b=1 \end{cases}$

$\Rightarrow w_1=2,\ w_2=2,\ b=-3$

Expand the original data point $(-\frac{1}{2},\sqrt{2})$ to $(\frac{1}{4},2)$

$\therefore f(x)=\text{sign}(wx+b)=\text{sign}(2\times\frac{1}{4}+2\times 2-3)=\text{sign}(\frac{1}{2}+4-3)=\text{sign}(\frac{3}{2})$

Since $\text{sign}(\frac{3}{2})=1$ $\therefore$ The predicted label of $(-\frac{1}{2},\sqrt{2})$ is +1.

1.4 Problem: Support Vector Machine 2 (18 pts.)

In this question you have to derive the dual form of SV regression (SVR). Your training data is $\{(x_1,y_1),\cdots,(x_n,y_n)\}$, where $x_i\in\mathbb{R}^m$, $y_i\in\mathbb{R}$.

Since the hinge loss that we used in class is only designed for classification we cannot use that for regression. A frequently used loss function for regression is the epsilon sensitive loss:

$$\mathcal{L}_\epsilon(x,y,f)=|y-f(x)|_\epsilon=\max(0,|y-f(x)|-\epsilon). \tag{13}$$

Here x is the input, y is the output, and f is the function used for predicting the label. Using this notation, the SVR cost function is defined as

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\mathcal{L}_\epsilon(x_i, y_i, f), \tag{14}$$

where $f(x) = w^\top x$, and C, $\epsilon > 0$ are parameters.

1) (3 pts.) Introduce appropriate slack variables, and rewrite this problem as a quadratic problem (i.e. quadratic objective with linear constraints). This form is called the Primal form of support vector regression.

2) (2 pts.) Write down the Lagrangian function for the above primal form.

3) (3 pts.) Using the Karush Kunh Tucker conditions derive the dual form.

4) (1 pts.) Can we use quadratic optimization solvers to solve the dual problem?

5) (2 pts.) How would you define support vectors in this problem?

6) (2 pts.) Write down the equation that can be used for predicting the label of unseen sample X.

7) (1 pts.) Is it possible to kernelize this algorithm?

8) (2 pts.) What happens if we change ϵ?

9) (2 pts.) What happens if we change C?

Solution:

(1) For each data point, introduce two slack variables:

ξ_i^+: Represent the slack on the positive side of the regression line

ξ_i^-: Represent the slack on the negative side of the regression line

$\therefore L_\epsilon(x, y, f) = \max(0, |y - f(x)| - \epsilon) = \max(0, |y_i - w^T x_i| - \epsilon)$

$\therefore$ The SVR cost function can now be written as:

$$\min_{w, b, \xi_i^+, \xi_i^-} \left(\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-)\right)$$

Subject to the constraints
- ① $y_i - w^T x_i \le \epsilon + \xi_i^+$ for all i
- ② $w^T x_i - y_i \le \epsilon + \xi_i^-$ for all i
- ③ $\xi_i^+, \xi_i^- \ge 0$ for all i

(2) Suppose the Lagrange multipliers are $\alpha_i, \beta_i^+, \beta_i^-$

$\therefore$ The Lagrangian function: $\mathcal{L}(w, b, \xi_i^+, \xi_i^-, \alpha_i, \beta_i^+, \beta_i^-) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{n}\alpha_i(y_i - w^T x_i - \epsilon - \xi_i^+) - \sum_{i=1}^{n}\beta_i^+\xi_i^+ - \sum_{i=1}^{n}\beta_i^-\xi_i^-$

where $y_i - w^T x_i \le \epsilon + \xi_i^+$, $\xi_i^+ \ge 0$, $\xi_i^- \ge 0$.

(3) The KKT conditions for the SVR primal problem are as follows:

① Stationarity: $\nabla_w \mathcal{L} = w - \sum_{i=1}^{n}\alpha_i x_i + \sum_{i=1}^{n}\beta_i^+ x_i - \sum_{i=1}^{n}\beta_i^- x_i = 0$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}\beta_i^+ - \sum_{i=1}^{n}\beta_i^- = 0$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i^+} = C - \alpha_i - \beta_i^+ = 0, \quad \frac{\partial \mathcal{L}}{\partial \xi_i^-} = C - \alpha_i - \beta_i^- = 0$$

② Primal feasibility:

$y_i - w^T x_i - \epsilon - \xi_i^+ \le 0$, $w^T x_i - y_i - \epsilon - \xi_i^- \le 0$, $\xi_i^+ \ge 0$, $\xi_i^- \ge 0$

③ Dual feasibility: $\alpha_i \ge 0$, $\beta_i^+ \ge 0$, $\beta_i^- \ge 0$

② complementary slackness: $\alpha_i (y_i - w^T x_i - \epsilon - \xi_i^+) = 0$, $\beta_i^+ \xi_i^+ = 0$, $\beta_i^- \xi_i^- = 0$

$\therefore C = \alpha_i + \beta_i^+ = \alpha_i + \beta_i^-$

$\therefore$ The dual Lagrangian:

$$L_d(\alpha_i, \beta_i^+, \beta_i^-) = \sum_{i=1}^{n} (\alpha_i + \beta_i^+ + \beta_i^-) - \frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \beta_i^+ + \beta_i^-)(\alpha_j - \beta_j^+ + \beta_j^-) x_i^T x_j - \epsilon \sum_{i=1}^{n} (\alpha_i - \beta_i^+ + \beta_i^-)$$

(4) Yes, quadratic optimization solvers can be used to solve the dual problem in Support Vector Regression (SVR). The dual problem in SVR typically involves quadratic optimization, which makes it suitable for quadratic optimization solvers.

(5) The support vectors in this problem should satisfy the following conditions:

① They are the data points for which the corresponding Lagrange multipliers α_i are nonzero.

② They lie on or within the margin boundary, so the slack variables ξ_i^+ and ξ_i^- associated with these data points are equal to zero.

(6) $f(X) = \sum_{i=1}^{n} \alpha_i (X \cdot x_i) + b$. where

- α_i are the Lagrange multipliers associated with the support vectors x_i.
- x_i are the support vectors.
- b is the bias term.
- X is the unseen sample.

(7) Yes. In kernelized SVR, the prediction function becomes:
$f(X) = \sum_{i=1}^{n} \alpha_i k(X, x_i) + b$. where $k(X, x_i)$ is the kernel function.

(8) Increasing ϵ:

① Larger margin of tolerance for errors

② The model will be less sensitive to individual data points and might lead to a smoother regression line.

③ May lead to a large number of support vectors and potentially

overfitting if the margin becomes too wide.

Decreasing ϵ:

① Smaller margin of tolerance for errors.

② The model will be more sensitive to individual data points and might lead to a more complex regression line that closely fit the training data.

③ It will lead to a smaller number of support vectors and potentially underfitting if the margin becomes too narrow.

In general: A larger ϵ leads to a simpler model with a larger margin but may sacrifice precision, while a smaller ϵ leads to a more complex model with a smaller margin but may capture more intricate patterns in the data.

(g) Adjusting C controls the trade-off between model complexity and generalization performance. A larger C leads to a more complex model with potentially better performance on the training data but a higher risk of overfitting, while a smaller C leads to a simpler model with potentially better generalization performance but a higher risk of underfitting.

1.5 Problem: C4.5 Decision Tree (5 pts.)

In this problem, we are set to construct a decision tree using the C4.5 algorithm. This particular algorithm leverages the Information Gain Ratio as a construction principle, guiding the formation of the decision tree.

Let's denote our dataset as D and its size as $|D|$. Suppose the dataset contains K distinct classes, labeled as C_k, where k ranges from 1 to K. The size of the individual data points allocated within each class C_k is expressed as $|C_k|$. As such, the sum of the sizes of all classes, i.e. $\sum_{k=1}^{K} |C_k| = |D|$. Given that the attribute A comprises n unique values, denoted as $\{a_1, a_2, \cdots, a_n\}$, it can bifurcate the dataset D into n subsets $D_1, D_2, \cdots, D_n$. Each subset size is in turn represented as $|D_i|$. It follows that the aggregation of all these subset sizes, i.e. $\sum_{i=1}^{n} |D_i|$, equates to the overall size of the original dataset, that is, $|D|$. Let the set of samples in subset D_i that belong to class C_k be represented by D_{ik}, that is, $D_{ik} = D_i \cap C_k$. The count of samples in D_{ik} is represented by $|D_{ik}|$. The empirical entropy of dataset is:

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}.$$

The empirical conditional entropy of attribute A respect to dataset D is:

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}.$$

The information gain is:

$$g(D, A) = H(D) - H(D|A).$$

We can calculate the Information Gain Ratio of attribute A for the dataset D is:

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}.$$

We can utilize the Information Gain Ratio to build a decision tree through the C4.5 algorithm, as described in Algorithm 1. For more information about the C4.5 algorithm, you can refer to the example `https://sefiks.com/2018/05/13/a-step-by-step-c4-5-decision-tree-example/`. According to the training dataset provided Table 1, generate a decision tree using the Information Gain Ratio (C4.5 algorithm).

Algorithm 1 C4.5 Algorithm

Require: Training dataset D, attribute set A, threshold ϵ
Ensure: Decision tree T

1: If D belongs to the same class C_k, T is a single-node tree, and class C_k is assigned as the label of the node. Return T
2: If A is an empty set, set T as a single-node tree, and assign the label of the node as the class with the highest number of instances. Return T
3: Calculate g_R for each attribute A, select the attribute A_g with the maximum **information gain ratio**
4: If the **information gain ratio** of A_g is less than ϵ, T is a single-node tree, and the class C_k with the maximum number of instances in D is assigned as the label. Return T
5: Split A_g into several non-empty subsets D_i
6: For each subset D_i, use it as the training dataset, $A - \{A_g\}$ as the attribute set, and recursively call the previous steps to obtain T_i. Return T_i

.

ID	Age	Work	House	Credit	Class
1	young	No	No	Normal	No
2	young	No	No	Good	No
3	young	Yes	No	Good	Yes
4	young	Yes	Yes	Normal	Yes
5	young	No	No	Normal	No
6	middle	No	No	Normal	No
7	middle	No	No	Good	No
8	middle	Yes	Yes	Good	Yes
9	middle	No	Yes	Excellent	Yes
10	middle	No	Yes	Excellent	Yes
11	old	No	Yes	Excellent	Yes
12	old	No	Yes	Good	Yes
13	old	Yes	No	Good	Yes
14	old	Yes	No	Excellent	Yes
15	old	No	No	Normal	No

Table 1: Dataset of Loan Application

Solution: Total samples $(|D|)=15$, Number of classes $(K)=2$ (Yes and No)

The class distribution: $|C_{Yes}|=9$, $|C_{No}|=6$

Empirical entropy: $H(D)=-\left(\frac{|C_{Yes}|}{|D|}\cdot\log_2\left(\frac{|C_{Yes}|}{|D|}\right)+\frac{|C_{No}|}{|D|}\cdot\log_2\left(\frac{|C_{No}|}{|D|}\right)\right)$

$=-\left(\frac{9}{15}\cdot\log_2\frac{9}{15}+\frac{6}{15}\cdot\log_2\frac{6}{15}\right)\approx 0.97$

Attribute: Age

(1) Class Distribution:

For Age = Young: $|C_{Yes}|=2$, $|C_{No}|=3$

For Age = Middle: $|C_{Yes}|=3$, $|C_{No}|=2$

For Age = Old: $|C_{Yes}|=4$, $|C_{No}|=1$

(2) Entropy:

For Age = Young: $H(D|Age=Young)=-\left(\frac{2}{5}\cdot\log_2\frac{2}{5}+\frac{3}{5}\cdot\log_2\frac{3}{5}\right)\approx 0.971$

For Age = Middle: $H(D|Age=Middle)=-\left(\frac{3}{5}\cdot\log_2\frac{3}{5}+\frac{2}{5}\cdot\log_2\frac{2}{5}\right)\approx 0.971$

For Age = Old: $H(D|Age=old)=-\left(\frac{4}{5}\cdot\log_2\frac{4}{5}+\frac{1}{5}\cdot\log_2\frac{1}{5}\right)\approx 0.722$

(3) Weighted average of entropies:

$H(D|Age)=\frac{5}{15}\times H(D|Age=Young)+\frac{5}{15}\times H(D|Age=Middle)+\frac{5}{15}\times H(D|Age=old)=\frac{5}{15}\times 0.971+\frac{5}{15}\times 0.971+\frac{5}{15}\times 0.722=0.888$

(4) Information gain:

$IG(Age)=H(D)-H(D|Age)\approx 0.97-0.888=0.082$

(5) Split information:

$SI(Age)=-\left(\frac{5}{15}\cdot\log_2\frac{5}{15}+\frac{5}{15}\log_2\frac{5}{15}+\frac{5}{15}\cdot\log_2\frac{5}{15}\right)\approx 1.585$

(6) Information gain ratio:

$IGR(Age)=\frac{IG(Age)}{SI(Age)}\approx\frac{0.082}{1.585}=0.052$

Attribute: Work

(1) Class Distribution:

For Work = Yes: $|C_{Yes}|=5$, $|C_{No}|=0$

For Work = No: $|C_{Yes}|=4$, $|C_{No}|=6$

(2) Entropy:

For Work = Yes: $H(D|Work=Yes)=-\frac{5}{5}\cdot\log_2\frac{5}{5}=0$

For Work = No: $H(D|Work=No) = -(\frac{4}{10}\cdot\log_2\frac{4}{10} + \frac{6}{10}\cdot\log_2\frac{6}{10}) = 0.971$

(3) Weighted Average of Entropies:

$H(D|Work) = \frac{5}{15} \times H(D|Work=Yes) + \frac{10}{15} \times H(D|Work=No) \approx 0.647$

(4) Information gain:

$IG(Work) = H(D) - H(D|Work) \approx 0.97 - 0.647 = 0.323$

(5) Split information:

$SI(Work) = -(\frac{5}{15}\cdot\log_2\frac{5}{15} + \frac{10}{15}\cdot\log_2\frac{10}{15}) \approx 0.918$

(6) Information gain ratio:

$IGR(Work) = \frac{IG(Work)}{SI(Work)} \approx \frac{0.323}{0.918} \approx 0.352$

Attribute: House

(1) Class distribution:

For House = Yes: $|C_{Yes}| = 6$, $|C_{No}| = 0$

For House = No: $|C_{Yes}| = 3$, $|C_{No}| = 6$

(2) Entropy: For House = Yes. $H(D|House=Yes) = -\frac{6}{6}\log_2\frac{6}{6} = 0$

For House = No. $H(D|House=No) = -(\frac{3}{9}\log_2\frac{3}{9} + \frac{6}{9}\log_2\frac{6}{9}) = 0.918$

(3) Weighted average of entropies:

$H(D|House) = \frac{6}{15} \times H(D|House=Yes) + \frac{9}{15} \times H(D|House=No) = 0.551$

(4) Information gain:

$IG(House) = H(D) - H(D|House) = 0.97 - 0.551 = 0.419$

(5) Split information:

$SI(House) = -(\frac{6}{15}\log_2\frac{6}{15} + \frac{9}{15}\log_2\frac{9}{15}) = 0.971$

(6) Information gain ratio:

$IGR(House) = \frac{IG(House)}{SI(House)} = \frac{0.419}{0.971} = 0.432$

Attribute: Credit

(1) Class Distribution:

For Credit = Normal. $|C_{Yes}| = 1$, $|C_{No}| = 4$

For Credit = Good. $|C_{Yes}| = 4$, $|C_{No}| = 2$

For Credit = Excellent. $|C_{Yes}| = 4$, $|C_{No}| = 0$

(2) Entropy:

For Credit = Normal, $HCD|Credit=Normal) = -(\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}) = 0.722$

For Credit = Good, $HCD|Credit=Good) = -(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}) = 0.918$

For Credit = Excellent, $HCD|Credit=Excellent) = -\frac{4}{4}\log_2\frac{4}{4} = 0$

(3) Weighted average of entropies:

$HCD|Credit) = \frac{5}{15} \times HCD|Credit=Normal) + \frac{6}{15} \times HCD|Credit=Good) + \frac{4}{15} \times HCD|Credit=Excellent) = 0.608$

(4) Information Gain:

$IG(Credit) = HCD) - HCD|Credit) = 0.97 - 0.608 = 0.362$

(5) Split information:

$SI(Credit) = -(\frac{5}{15}\log_2\frac{5}{15} + \frac{6}{15}\log_2\frac{6}{15} + \frac{4}{15}\log_2\frac{4}{15}) = 1.566$

(6) Information Gain Ratio:

$IGR(Credit) = \frac{IG(Credit)}{SI(Credit)} = \frac{0.362}{1.566} = 0.231$

Summarize calculated gain and gain ratios:

Attribute	Gain	Gain Ratio
Age	0.082	0.052
Work	0.323	0.352
House	0.419	0.432
Credit	0.362	0.231

House Attribute comes with both maximized gain and gain ratio. This means that we need to put House attribute in root of decision tree.

House = Yes

ID	House	Age	Work	Credit	Class
4	Yes	Young	Yes	Normal	Yes
8	Yes	Middle	Yes	Good	Yes
9	Yes	Middle	No	Excellent	Yes
10	Yes	Middle	No	Excellent	Yes
11	Yes	Old	No	Excellent	Yes
12	Yes	Old	No	Good	Yes

When House = Yes.
The class is always Yes.
So we don't need to split further here.

House = No

ID	House	Age	Work	Credit	Class
1	No	Young	No	Normal	No
2	No	Young	No	Good	No
3	No	Young	Yes	Good	Yes
5	No	Young	No	Normal	No
6	No	Middle	No	Normal	No
7	No	Middle	No	Good	No
13	No	Old	Yes	Good	Yes
14	No	Old	Yes	Excellent	Yes
15	No	Old	No	Normal	No

Since when House = No, the class can be No or Yes. We need to split the decision tree further.

As seen, class would be yes when Work is yes, and it would be no if work is no.

Therefore, final form of decision tree is demonstrated below.