Homework 2
Naïve Bayes; SVM

CMU 10-701: Machine Learning (Spring 2017)

OUT: Feb 13
DUE: Feb 27, 11:59 PM

Section A : Multiple Choice Questions

- **There might be one or more right answers**. Please explain your choice in one or two sentences.

1. [**This is to remind you that there might be one or more answers.**] Which of the following is(are) *not* one of the TAs of this class?
   A. Yichonnnnnnnng Xu
   B. Ademacia Yu
   C. Danish Schwartz
   D. Pradeep Prakhar Naval
   ABCD. Check the piazza for the real names!

2. [**2 Points**] Which of the following methods is(are) model-based method(s)?

   A. Support Vector Machines
   B. Logistic Regression
   C. Naive Bayes
   D. Linear Regression

   C. Naive Bayes is a generative model.

3. [**2 Points**] Suppose we are learning a Naive Bayes model on a 2-dimensional binary data using MLE estimation. That is, we are learning a naive Bayes classifier from $X \in \{0,1\}^2$ to $Y \in \{0,1\}$. Suppose the training data is $((0,0),1),((0,1),1),((1,1),1),((1,1),0),((0,0),0)$. What is the maximum likelihood estimation(MLE) probability of $P(Y=1|X=(1,0))$ under the Naive Bayes model?
   A. $\frac{2}{5}$. B. $\frac{1}{3}$. C. $\frac{3}{5}$. D. $\frac{11}{18}$.
   A. We compute that $P(Y=1)=3/5, P(Y=0)=2/5$ and $P(X_1=1|Y=1)=P(X_2=0|Y=1)=1/3$, $P(X_1=1|Y=0)=P(X_2=0|Y=0)=1/2$. So the computed probability is

$$\frac{3/5*(1/3)*(1/3)}{3/5*(1/3)*(1/3)+2/5*(1/2)*(1/2)}=0.4.$$

4. [**2 Points**] For each of the following loss functions, which one is(are) non-convex in $f(x)$?

   A. 0-1 loss: $l(y,f(x))=1_{y\neq f(x)}$.
   B. Mean squared error: $l(y,f(x))=(y-f(x))^2$.
   C. The hinge loss: $l(y,f(x))=\max(0,1-yf(x))$.
   D. The logistic loss: $l(y,f(x))=\log(1+\exp(-yf(x)))$.

A. It is easy to verify all others are convex.

5. **[2 points]** Suppose we use the following form of SVM on a dataset $\{(x_i, y_i)\}_{i=1}^n$:

$$\text{minimize } \frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda\|w\|_2^2$$
$$\text{subject to } y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, ..., n.$$

Suppose we have $\xi_1 = 1, \xi_2 = 0, \xi_3 = 2$ at the optimal solution. Which of the following statements is(are) true for the optimal classifier?
A. x_1 lies on the decision boundary.
B. x_2 lies on the decision boundary.
C. x_2 lies on the wrong side of the decision boundary (i.e., y_2 is of a different sign as the classifier prediction.).
D. x_3 lies on the wrong side of the decision boundary (i.e., y_3 is of a different sign as the classifier prediction.).
AD. $\xi_i = 0$ indicates the sample lies correctly beyond the margin; $\xi_i = 1$ indicates the sample lies exactly on the boundary; $\xi_i > 1$ indicates the sample is not classified correctly.

Part B, Problem 1: Naïve Bayes[20 Points] (Adams and Danish)

Part 1.1 :

$$P(Y|X) = P(X|Y)P(Y)$$
$$P(X|Y) = P(X_1X_2X_3..X_n|Y)$$

Using conditional independence assumption

$$P(X|Y) = P(X_1X_2X_3..X_n|Y) = \prod_{i=1}^{n} P(X_i|Y)$$

Part 1.2 : Using the independence assumption, we have the following

$$P(Y|X) = P(X|Y)P(Y)$$
$$P(Y|X) = \prod_{i=1}^{n} P(X_i|Y)P(Y)$$

For this model, we need to estimate all values of the form $P(X_i = k|Y = j)$ where X_i takes values from $\{1, 2, ..., K\}$ and Y ranges from $\{0, 1\}$. Hence, to estimate $P(X_i = k|Y = j)$ will require at most $K * 2$ parameters. In fact, since $\sum_{l=1}^{k} P(X_i = l|Y = j) = 1$, we can use one less parameter, hence total number of parameters would be $2 * (K - 1)$.
Note that we would have to do this for each i, where i ranges from 1 to n, hence total number of parameters are $2(K - 1)n$

Similarly, without the independence assumption, we need to model $P(X_1X_2..X_n|Y = j)$, here the number of possible outcomes for $X_1X_2X_3..X_n$ can be K^n, and here too we can avoid one paramater. Hence, the total parameters are $2(K^n - 1)$. Notice the difference in the number of parameters in the two cases.

Part 1.3: We know the following

$$P(Y = 1) = \pi$$
$$P(Y = 0) = 1 - \pi$$
$$P(Y) = \pi^Y(1 - \pi)^{1-Y}$$
$$P(X_i = k|Y = j) = \mu_{ijk}$$

Now, we write the likelihood term

$$l(\pi,\mu) = \prod_{p=1}^{N} P(X^p, Y^p; \mu, \pi)$$

$$l(\pi,\mu) = \prod_{p=1}^{N} P(Y^p;\pi) \prod_{p=1}^{N} P(X^p|Y^p;\mu)$$

$$l(\pi,\mu) = \prod_{p=1}^{N} P(Y^p;\pi) \prod_{p=1}^{N}\prod_{i=1}^{n} P({X_i}^p|Y^p;\mu)$$

$$L(\pi,\mu) = \log l(\pi,\mu)$$

$$L(\pi,\mu) = \log \prod_{p=1}^{N} P(Y^p;\pi) + \log \prod_{p=1}^{N}\prod_{i=1}^{n} P({X_i}^p|Y^p;\mu)$$

$$L(\pi,\mu) = \log\left((\pi)^{\sum Y^p}(1-\pi)^{N-\sum Y^p}\right) + \sum_{p=1}^{N}\sum_{i=1}^{n} \log P({X_i}^p = {x_i}^p | Y^p = y^p;\mu) \tag{1}$$

$$L(\pi,\mu) = \log\left((\pi)^{\sum Y^p}(1-\pi)^{N-\sum Y^p}\right) + \sum_{p=1}^{N}\sum_{i=1}^{n} \mu_{iy^p x_i^p}$$

$$\hat{\pi} = \underset{\pi}{\operatorname{argmax}}\, L(\pi,\mu)$$

$$\hat{\mu} = \underset{\mu}{\operatorname{argmax}}\, L(\pi,\mu)$$

Taking the derivate, and equating it to zero, we get the following

$$\hat{\pi} = \frac{\sum_{i=1}^{N} \mathbb{1}_{Y=1}}{N}$$

where $\mathbb{1}$ is an indicator function.

Also, the feature distribution conditioned on a label is Bernoulli, and after taking the derivate and equating it to zero we get,

$$\hat{\mu_{ijk}} = \frac{\sum_{i=1}^{N} \mathbb{1}_{Y=j;X_i=k}}{\sum_{i=1}^{N} \mathbb{1}_{Y=j}}$$

Part 1.4: Given N observations $\{(X^p, Y^p)\}_{p=1}^{N}$, derive the MLE estimator of θ_{ij}
Directly continuing from equation 1, we get
Using the result of the MLE estimate for the gaussian distribution, we can directly write

$$L(\pi,\theta) = \log\left((\pi)^{\sum Y^p}(1-\pi)^{N-\sum Y^p}\right) + \sum_{p=1}^{N}\sum_{i=1}^{n} \log P({X_i}^p = {x_i}^p | Y^p = y^p;\theta)$$

$$L(\pi,\theta) = \log\left((\pi)^{\sum Y^p}(1-\pi)^{N-\sum Y^p}\right) + \sum_{p=1}^{N}\sum_{i=1}^{n} (-0.5 * ({x_i}^p - \theta_{iy^p})^2) + C$$

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(\pi,\theta)$$

Taking derivate and equating it to zero we get,

$$\hat{\theta_{ij}} = \frac{\sum_{i=1}^{N}(\mathbb{1}_{Y=j}X_i)}{\sum_{i=1}^{N}\mathbb{1}_{Y=j}}$$

Part 1.5 Proving that the decision boundary is linear
Decision boundary is given by the following equation

$$X_1, X_2, ..., X_n : P(Y = 0|X_1, X_2, ..., X_n) = P(Y = 1|X_1, X_2, ..., X_n)$$
$$P(Y = 1|X_1, X_2, ..., X_n) = \pi \prod_{i=1}^{N}(P(X_i|Y = 1))$$
$$P(Y = 0|X_1, X_2, ..., X_n) = (1 - \pi) \prod_{i=1}^{N}(P(X_i|Y = 0))$$

equating the two forms and using gaussian distribution, we get

$$\pi e^{\sum_{i=1}^{N}(-0.5(X_i-\theta_{i1})^2)} = (1 - \pi)e^{\sum_{i=1}^{N}(-0.5(X_i-\theta_{i0})^2)}$$

Taking log transformation

$$\log \frac{\pi}{1-\pi} + \sum_{i=1}^{N}(-0.5(X_i - \theta_{i1})^2) = \sum_{i=1}^{N}(-0.5(X_i - \theta_{i0})^2)$$

Notice that after expanding the expression, the quadratic terms will cancel with each other, and the decision boundary will be linear in input features. Hence, the decision boundary is a line.

Support Vector Machine (25 pts) (Hao and Prakhar)

Suppose we have the following data

$$\mathcal{D} = (\mathbf{X}, \mathbf{y})$$

where $\mathbf{X} \in \mathbb{R}^{d \times n}$, the i-th column x_i are the features of the i-th training sample and y_i is the label of the i-th training sample. $y \in \{-1, 1\}^n$ if this is a classification problem and $y \in \mathbb{R}^n$ if this is a regression problem.

0.1 SVM for Classification (15 pts)

1. (5 pts) In the linearly separable case if one of the training samples is removed, will the decision boundary shift toward the point removed or shift away from the point removed or remain the same? Justify your answer.

   Now if we consider that the decision boundary is of Logistic Regression, will the decision boundary change or remain the same? Explain you answer. (No need to mention the direction of change)

   Solution:
   In the linearly separable case, if one of the training samples is removed:
   (1) If the point is not a support vector, then the margin remains unchanged.
   (2) If the point is a support vector, then the margin length can become larger and move towards the point which is removed if the point was the only support vector or remain unchanged otherwise.

   Logistic regression focusses on maximizing the probability of the data. The farther the data lies from the separating hyperplane, the happier LR is as opposed to SVM which tries to explicitly find the maximum margin. If a point is not a support vector, it doesn't really matter. Since LR is a density estimation technique, each point will carry some weight and have some effect on the decision boundary.

   Recall from the lecture notes that if we allow some misclassification in the training data, the primal optimization of SVM (soft margin) is given by

$$\begin{aligned} \underset{w,\xi_i}{\text{minimize}} \quad & \frac{1}{2}||w||_2^2 + C\sum_{i=1}^{n}\xi_i & \\ \text{subject to} \quad & y_i(w^T x_i) \geq 1 - \xi_i & \forall i = 1, \ldots, n \\ & \xi_i \geq 0 & \forall i = 1, \ldots, n \end{aligned}$$

   where $\xi_1, \ldots, \xi_n$ are called slack variables.

2. (3 pts) Suppose the optimal $\xi_1, \ldots, \xi_n$ have been computed. Use the ξ_i to obtain an upper bound on the number of misclassified instances.

   Solution:
   The hinge loss is the upper bound on the number of misclassified instances.
   Here, we choose the hinge loss function as $h(z) = max(0, 1 - z)$.
   The primal optimization of SVM is given by

$$\underset{w,\xi_i}{\text{minimize}} \frac{1}{2}||w||_2^2 + C\sum_{i=1}^{n}\xi_i$$

   Now the slack variable appears in 1 constraint and we try to minimize that, i.e., we satisfy one of the two constraints given, i.e.,

$$y_i(w^T x_i) \geq 1 - \xi_i$$

or

$$\xi_i \geq 0$$

The slack variable is the tighter/larger one of the two numbers. So it can either be zero or $1 - y_i(w^T x_i)$
Thus,

$$\xi_i = max(0, 1 - y_i(w^T x_i))$$

which is the hinge loss function.
Hence,

$$\xi_i = max(0, 1 - y_i(w^T x_i)) = h(y_i(w^T x_i))$$

And we know that the hinge loss is the upper bound on the number of misclassified instances. Thus, the upper bound is given by

$$\implies \sum_{i=1}^{n} h(y_i(w^T x_i))$$

or simply

$$\sum_{i=1}^{n} (\xi_i > 1)$$

3. (3 pts) In the primal optimization of SVM, what's the role of the coefficient C? Briefly explain your answer by considering two extreme cases, i.e., $C \to 0$ and $C \to \infty$.

   Solution:
   C is the trade-off parameter which tells us whether we would rather have a small norm of w, meaning a large margin, or rather have no violations of the margin constraints, meaning small sum of hinge loss.
   (1) When $C \to 0$, more emphasis is given to find the largest margin irrespective of several noises, i.e, no misclassification will be penalized.
   (2) When $C \to \infty$, we put a higher and higher weight on violations of margin constraints, so we find a hyperplane where the required slack is minimized even at the expense of the margin.

4. (2 pts) Compare Hard SVM and Logistic Regression when the two classes are linearly separable. Give any significant differences. (*Hint* - think in terms of decision boundary)

   Solution: When two classes are linearly separable, both Hard SVM and Logistic Regression can always find a solution. The major difference is that Logistic Regression finds a decision boundary that maximize it's likelihood function while Hard SVM finds a decision boundary with maximal margin.

5. (2 pts) Compare Soft SVM and Logistic Regression when the two classes are not linearly separable. Give any significant differences.

   Solution: When the two classes are not linearly separable, Logistic Regression will still find a decision boundary that maximize its likelihood function. For Soft SVM, it will find a decision boundary that best balance the margin and errors. (Note: the key difference between SVM and Logistic Regression is that SVM is more of a geometric-motivated model while Logistic Regression is more of a probability-motivated model.)

0.2 SVM for Regression (10 pts)

Let $x \in \mathbb{R}^d$ be the feature vector and $y \in \mathbb{R}$ be the label. In this question, we use a linear predictor for the label, i.e. given the feature vector, we predict the label by

$$\hat{y}(x) = w^\top x$$

where $w \in \mathbb{R}^d$ is the linear coefficient vector. In this question, we consider the **epsilon insensitive loss function**, defined by

$$L_\epsilon(y, \hat{y}) = \begin{cases} 0 & \text{if } |y - \hat{y}| < \epsilon \\ |y - \hat{y}| - \epsilon & \text{otherwise.} \end{cases}$$

where ϵ is a tuning parameter. To obtain a good w, we would like to solve the following optimization:

$$J(w) = \frac{1}{n} \sum_{i=1}^{n} L_\epsilon(y, \hat{y}(x_i)) + \lambda \|w\|_2^2. \tag{2}$$

1. (2 pts) When $\epsilon = 0$, is the loss function the same as the absolute error loss? Show why. What role do you think ϵ plays?
   Solution:
   When $\epsilon = 0$, the nature of the loss function L_ϵ is equivalent to that of $|x|$ or in other words an absolute value norm. So L_ϵ is basically the absolute loss.
   Compare this function to the squared error loss function: We can see that in this function, as long as the error is within ϵ, we do not penalize it. However, in a squared error function, even for a small error, we penalize it by the square of it and so on.

2. (8 pts) Notice that (2) is not a differentiable. Show how to convert (2) into a optimization problem whose objective is differentiable and constraints are linear by introducing slack variables.
   Solution:
   The optimization problem give is

   $$J(w) = \frac{1}{n} \sum_{i=1}^{n} L_\epsilon(y, \hat{y}(x_i)) + \lambda \|w\|_2^2$$

   We know that hinge loss is an upper bound on the number of misclassified examples. Our loss function in this case is similar to the hinge loss. Thus, we can express this function as:

   $$J(w) = \frac{1}{n} \sum_{i=1}^{n} \xi_i + \lambda \|w\|_2^2$$

   where ξ_i are the slack variables.
   We know that the loss function is given by:

   $$L_\epsilon(y, \hat{y}(x_i)) = \begin{cases} 0 & \text{if } |y - \hat{y}(x_i)| < \epsilon \\ |y - \hat{y}(x_i)| - \epsilon & \text{otherwise.} \end{cases}$$

   So considering the nature of our loss function, we can convert this optimization problem into a form having the following constraints:

   $$y - \hat{y}(x_i) \leq \epsilon + \xi_i$$

   and

   $$-(y - \hat{y}(x_i)) \leq \epsilon + \xi_i$$
   $$\implies (y - \hat{y}(x_i)) \geq -\epsilon - \xi_i$$

   These constraints are correct since let us assume that ξ_i is zero. Then in that case, $y - \hat{y}(x_i)$ should lie between $-\epsilon - \xi_i$ and $\epsilon + \xi_i$ which is satisfied by the constraints.
   Thus, the optimization problem of

   $$J(w) = \frac{1}{n} \sum_{i=1}^{n} L_\epsilon(y, \hat{y}(x_i)) + \lambda \|w\|_2^2$$

can be converted to

$$J(w) = \frac{1}{n}\sum_{i=1}^{n} \xi_i + \lambda \|w\|_2^2$$

having constraints

$$y - \hat{y}(x_i) \leq \epsilon + \xi_i$$

$$y - \hat{y}(x_i) \geq -\epsilon - \xi_i$$

and

$$\xi_i \geq 0$$

which is an optimization problem having a differentiable objective and linear constraints.

Part B, Problem 3: Minimum Enclosing Ball [x Points] (Calvin and Dan)

1. **Lagrangian (Hard MEB)**:

$$\mathcal{L}(R, \boldsymbol{c}, \boldsymbol{\alpha}) = R^2 - \sum_{i=1}^{n} \alpha_i \left(R^2 - \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 \right)$$

2. **Dual (Hard MEB):**
   $\frac{\partial \mathcal{L}}{\partial R}$

$$\begin{aligned} 0 = \frac{\partial \mathcal{L}}{\partial R} &= 2R - 2R \sum_{i=1}^{n} \alpha_i \\ \implies \sum_{i=1}^{n} \alpha_i &= 1 \end{aligned}$$

   $\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}}$

$$\begin{aligned} 0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}} &= 2 \sum_{i=1}^{n} \alpha_i \left(\boldsymbol{c} - \boldsymbol{z}_i \right) \\ \boldsymbol{c} \sum_{i=1}^{n} \alpha_i &= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\ \boldsymbol{c} &= \left(\sum_{i=1}^{n} \alpha_i \right)^{-1} \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\ &= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \end{aligned}$$

   $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}}$

$$\begin{aligned} 0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}} &= \sum_{i=1}^{n} \left(R^2 - \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 \right) \\ R^2 &= \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 \end{aligned}$$

Then the dual criterion is:

$$\begin{aligned}
\mathcal{L}(R^*, \boldsymbol{c}^*, \boldsymbol{\alpha}) &= (R^*)^2 - \sum_{i=1}^{n} \alpha_i (R^*)^2 + \sum_{i=1}^{n} \alpha_i \|\boldsymbol{c}^* - \boldsymbol{z}_i\|_2^2 \\
&= \sum_{i=1}^{n} \alpha_i \|\boldsymbol{c}^* - \boldsymbol{z}_i\|_2^2 \\
&= \sum_{i=1}^{n} \alpha_i \left[(\boldsymbol{c}^*)^\intercal (\boldsymbol{c}^*) - 2 (\boldsymbol{c}^*)^\intercal \boldsymbol{z}_i + \boldsymbol{z}_i^\intercal \boldsymbol{z}_i \right] \\
&= (\boldsymbol{c}^*)^\intercal (\boldsymbol{c}^*) \sum_{i=1}^{n} \alpha_i - 2 \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \alpha_j \boldsymbol{z}_j \right)^\intercal \boldsymbol{z}_i + \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_i \\
&= \left(\sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \right)^\intercal \left(\sum_{j=1}^{n} \alpha_j \boldsymbol{z}_j \right) - 2 \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j + \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_1 \\
&= \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j - 2 \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j + \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_1 \\
&= - \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j + \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_i
\end{aligned}$$

The dual problem can be written as:

$$\begin{aligned}
\underset{\alpha}{\text{argmax}} \quad & - \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j + \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_i \\
\text{subject to} \quad & \alpha_i \geq 0, \quad \forall i \\
& \sum_{i=1}^{n} \alpha_i = 1
\end{aligned}$$

Or equivalently:

$$\begin{aligned}
\underset{\alpha}{\text{argmin}} \quad & \sum_{i,j=1}^{n} \alpha_i \alpha_j \boldsymbol{z}_i \boldsymbol{z}_j - \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_i \\
\text{subject to} \quad & \alpha_i \geq 0, \quad \forall i \\
& \sum_{i=1}^{n} \alpha_i = 1
\end{aligned}$$

3. **Lagrangian (Soft MEB)**:

$$\mathcal{L}(R, \boldsymbol{c}, \boldsymbol{\zeta}, \boldsymbol{\alpha}, \boldsymbol{\gamma}) = R^2 + \eta \sum_{i=1}^{n} \zeta_i - \sum_{i=1}^{n} \alpha_i \left(R^2 + \zeta_i - \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 \right) - \sum_{i=1}^{n} \gamma_i \zeta_i$$

4. **Dual Problem (Soft MEB)**:

$\frac{\partial \mathcal{L}}{\partial R}$

$$
\begin{aligned}
0 = \frac{\partial \mathcal{L}}{\partial R} &= 2R - 2R \sum_{i=1}^{n} \alpha_i \\
\implies \sum_{i=1}^{n} \alpha_i &= 1
\end{aligned}
$$

$\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}}$

$$
\begin{aligned}
0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}} &= 2 \sum_{i=1}^{n} \alpha_i \left(\boldsymbol{c} - \boldsymbol{z}_i \right) \\
\boldsymbol{c} \sum_{i=1}^{n} \alpha_i &= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\
\boldsymbol{c} &= \left(\sum_{i=1}^{n} \alpha_i \right)^{-1} \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\
&= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i
\end{aligned}
$$

$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}}$

$$
\begin{aligned}
0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}} &= \sum_{i=1}^{n} \left(R^2 + \zeta_i - \| \boldsymbol{c} - \boldsymbol{z}_i \|_2^2 \right) \\
R^2 &= \frac{1}{n} \sum_{i=1}^{n} \| \boldsymbol{c} - \boldsymbol{z}_i \|_2^2 + \zeta_i
\end{aligned}
$$

$\frac{\partial \mathcal{L}}{\partial \zeta_i}$

$$
\begin{aligned}
0 = \frac{\partial \mathcal{L}}{\partial \zeta_i} &= \eta - \alpha_i - \gamma_i \\
\alpha_i &= \eta - \gamma_i \\
\gamma_i &= \eta - \alpha_i
\end{aligned}
$$

Satisfying $\alpha_i \geq 0, \gamma_i \geq 0$ can equivalently be done by constraining $0 \leq \alpha_i \leq \eta$ in the dual problem. We can then substitute for γ_i in the Lagrangian.

$$
\begin{aligned}
g(\boldsymbol{\alpha}) = \min_{R,\boldsymbol{c},\boldsymbol{\zeta}} \mathcal{L}(R,\boldsymbol{c},\boldsymbol{\zeta},\boldsymbol{\alpha}) &= R^2 + \eta\sum_{i=1}^{n}\zeta_i - \sum_{i=1}^{n}\alpha_i\left(R^2+\zeta_i - \|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2\right) - \sum_{i=1}^{n}\gamma_i\zeta_i \\
&= \min_{R,\boldsymbol{c},\boldsymbol{\zeta}} R^2 + \eta\sum_{i=1}^{n}\zeta_i - \sum_{i=1}^{n}\alpha_i\left(R^2+\zeta_i - \|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2\right) - \sum_{i=1}^{n}(\eta-\alpha_i)\zeta_i \\
&= \min_{R,\boldsymbol{c}} R^2 - \sum_{i=1}^{n}\alpha_i\left(R^2 - \|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2\right) \\
&= \min_{R,\boldsymbol{c}} R^2 - R^2\sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}\alpha_i\|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2 \\
&= \min_{\boldsymbol{c}} \sum_{i=1}^{n}\alpha_i\|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2 \\
&= \min_{\boldsymbol{c}} \sum_{i=1}^{n}\alpha_i\left(\boldsymbol{c}^\intercal\boldsymbol{c} - 2\boldsymbol{z}_i^\intercal\boldsymbol{c} + \boldsymbol{z}_i^\intercal\boldsymbol{z}_i\right) \\
&= \min_{\boldsymbol{c}} \sum_{i=1}^{n}\alpha_i\boldsymbol{c}^\intercal\boldsymbol{c} - 2\sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{c} + \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i \\
&= \min_{\boldsymbol{c}} \boldsymbol{c}^\intercal\boldsymbol{c} - 2\sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{c} + \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{z}_i^\intercal\boldsymbol{z}_j - 2\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{z}_i^\intercal\boldsymbol{z}_j + \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{z}_i^\intercal\boldsymbol{z}_j + \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i
\end{aligned}
$$

Then the dual problem is:

$$
\begin{aligned}
\underset{\boldsymbol{\alpha}}{\text{argmax}} \quad & -\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{z}_i^\intercal\boldsymbol{z}_j + \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i \\
\text{subject to} \quad & 0 \le \boldsymbol{\alpha} \le \eta \\
& \sum_{i=1}^{n}\alpha_i = 1
\end{aligned}
$$

Or equivalently:

$$
\begin{aligned}
\underset{\boldsymbol{\alpha}}{\text{argmin}} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{z}_i^\intercal\boldsymbol{z}_j - \sum_{i=1}^{n}\alpha_i\boldsymbol{z}_i^\intercal\boldsymbol{z}_i \\
\text{subject to} \quad & 0 \le \boldsymbol{\alpha} \le \eta \\
& \sum_{i=1}^{n}\alpha_i = 1
\end{aligned}
$$

5. **Boundary conditions on α_i:**
From complementary slackness, we have:

$$
\begin{aligned}
\alpha_i > 0 &\implies h_i(R,\boldsymbol{c},\boldsymbol{\zeta}) = 0 = \|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2 - R^2 - \zeta_i \\
&\implies \|\boldsymbol{c}-\boldsymbol{z}_i\|_2^2 = R^2 + \zeta_i
\end{aligned}
$$

Recalling that we also have $\gamma_i = \eta - \alpha_i$, we have:

$$\begin{aligned}
\eta - \alpha_i > 0 &\implies \zeta_i = 0 \\
\eta > \alpha_i > 0 &\implies \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 = R^2
\end{aligned}$$

Therefore we have:

$$\begin{aligned}
\|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 < R^2 &\implies \alpha_i = 0 \\
\|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 = R^2 &\implies 0 < \alpha_i < \eta \\
\|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2 > R^2 &\implies \alpha_i = \eta
\end{aligned}$$

6. **Primal from dual**:

$$\begin{aligned}
\boldsymbol{c} &= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\
R^2 &= \|\boldsymbol{c} - \boldsymbol{z}_i\|_2^2, \quad \text{where } 0 < \alpha_i < \eta
\end{aligned}$$

7. **L2-SVM Dual**:

$$\mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}) = \frac{1}{2}\|w\|_2^2 + \frac{1}{2}b^2 - \rho + \frac{\lambda}{2}\sum_{i=1}^{n}\xi_i^2 - \sum_{i=1}^{n}\alpha_i\left(y_i\left(\boldsymbol{w}^\intercal x_i + b\right) - \rho + \xi_i\right)$$

$$\begin{aligned}
0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i &\implies \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i \\
0 = \frac{\partial \mathcal{L}}{\partial b} = b - \sum_{i=1}^{n}\alpha_i y_i &\implies b = \sum_{i=1}^{n}\alpha_i y_i \\
0 = \frac{\partial \mathcal{L}}{\partial \xi_i} = \lambda\xi_i - \alpha_i &\implies \xi_i = \frac{\alpha_i}{\lambda_i} \\
0 = \frac{\partial \mathcal{L}}{\partial \rho} = -1 + \sum_{i=1}^{n}\alpha_i &\implies \sum_{i=1}^{n}\alpha_i = 1
\end{aligned}$$

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{w}^*, b^*, \boldsymbol{\xi}^*, \rho^*, \boldsymbol{\alpha}) =& \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\right)^\intercal \left(\sum_{j=1}^{m}\alpha_j y_j \boldsymbol{x}_j\right) + \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i\right)^2 - \rho^* + \frac{\lambda}{2}\sum_{i=1}^{n}\frac{\alpha_i^2}{\lambda^2} \\
& - \sum_{i=1}^{n}\left[\alpha_i y_i \left(\sum_{j=1}^{n}\alpha_j y_j \boldsymbol{x}_j\right)^\intercal \boldsymbol{x}_i + \alpha_i y_i \left(\sum_{j=1}^{n}\alpha_j y_j\right) - \alpha_i \rho^* + \frac{\alpha_i^2}{\lambda}\right] \\
=& \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j + \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i\right)^2 - \rho^* + \frac{1}{2\lambda}\sum_{i=1}^{n}\alpha_i^2 \\
& - \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j - \sum_{i=1}^{n}\alpha_i y_i \left(\sum_{j=1}^{n}\alpha_j y_j\right) + \rho^*\sum_{i=1}^{n}\alpha_i - \frac{1}{\lambda}\sum_{i=1}^{n}\alpha_i^2 \\
=& -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j + \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i\right)^2 - \frac{1}{2\lambda}\sum_{i=1}^{n}\alpha_i^2 - \left(\sum_{i=1}^{n}\alpha_i y_i\right)^2 \\
=& -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j - \frac{1}{2\lambda}\sum_{i=1}^{n}\alpha_i^2 \\
=& -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \left(\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda}\right), \quad \delta_{ij} = \{1 : i = j, 0 : i \neq j\}
\end{aligned}
$$

Then the dual problem is:

$$
\begin{aligned}
\underset{\boldsymbol{\alpha}}{\text{argmax}} \quad & -\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \left(\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda}\right) \\
\text{subject to} \quad & \boldsymbol{\alpha} \geq 0 \\
& \sum_{i=1}^{n}\alpha_i = 1
\end{aligned}
$$

Or equivalently:

$$
\begin{aligned}
\underset{\boldsymbol{\alpha}}{\text{argmin}} \quad & \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \left(\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda}\right) \\
\text{subject to} \quad & \boldsymbol{\alpha} \geq 0 \\
& \sum_{i=1}^{n}\alpha_i = 1
\end{aligned}
$$

8. **Equivalence of MEB and L2-SVM**:
   The constraints in the dual problems are the same, so we just plug the augmented data into the dual

criterion of the MEB problem:

$$
\begin{aligned}
&\underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j \boldsymbol{z}_i^\intercal \boldsymbol{z}_j - \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i^\intercal \boldsymbol{z}_i \\
&= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j \left[y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j + y_i y_j + \frac{\delta_{ij}}{\lambda} \right] - \sum_{i=1}^{n} \alpha_i \left[\boldsymbol{x}_i^\intercal \boldsymbol{x}_i + 1 + \frac{1}{\lambda} \right] \\
&= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \left[\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda} \right] - \sum_{i=1}^{n} \left[1 + 1 + \frac{1}{\lambda} \right] \alpha_i \\
&= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \left[\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda} \right] - \left[1 + 1 + \frac{1}{\lambda} \right] \sum_{i=1}^{n} \alpha_i \\
&= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \left[\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda} \right] - \left[1 + 1 + \frac{1}{\lambda} \right] 1 \\
&= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \left[\boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1 + \frac{\delta_{ij}}{\lambda} \right]
\end{aligned}
$$

9. **Solutions of L2-SVM from MEB**:

$$
\begin{aligned}
\boldsymbol{c} &= \sum_{i=1}^{n} \alpha_i \boldsymbol{z}_i \\
&= \sum_{i=1}^{n} \alpha_i \left[y_i \boldsymbol{x}_i, y_i, \frac{1}{\lambda} \boldsymbol{e}_i \right]^\intercal \\
&= \left[\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i, \sum_{i=1}^{n} \alpha_i y_i, \frac{1}{\lambda} \sum_{i=1}^{n} \boldsymbol{e}_i \right]^\intercal \\
&= \left[\boldsymbol{w}, b, \frac{1}{\lambda} \sum_{i=1}^{n} \boldsymbol{e}_i \right]^\intercal
\end{aligned}
$$

So $\boldsymbol{w}$ is the first p components of c and b is the $(p+1)th$ component of c.

Part C: Programming Exercise [25 Points] (Weixiang and Yiting)

Note: Your code for all of the programming exercises should be submitted to Gradescope. There is a separate 'programming assignment' that should allow you to upload your code easily. Code should be uploaded to this separate programming assignment, while visualizations and written answers should still be submitted within the primary Gradescope assignment. In your code, **Please make it clear in the comments which are the primary functions to compute the answers to each question.**

Note: The data files for this subproblem, and the following subproblems can be found here.

Problem 1: Is Naive Bayes Really "Naive"? [15 Points] (Weixiang)

In this problem, you will hopefully gain more insights about Naive Bayes, and how it performs in the real world. For our task, you still use bag of words representation but apply the Naive Bayes classification on the dataset we provide to make classifications. More specifically, you will explore Naive Bayes to classify movie

reviews into two classes - positive & negative, the same as homework 1! **You are free to use any Naive Bayes third-party libraries (e.g., sklearn). But you must write your own code to evaluate accuracy and answer any other questions below.**

Dataset

Note: The dataset for this subsection is the same as the dateset of homework 1 Logistic Regression Problem. As a reminder, The dataset comprises of two folders : 'train' and 'test', and each of these in turn contain two subfolders - pos & neg. Each file in these subfolders is a unique review. In total, we have 25K training reviews (12.5K positive, and remaining 12.5K negative). The test folder too has 12.5K positive and 12.5K negative reviews.

Exercises

You first need to build the Naive Bayes model with the word counts for both classes (pos and neg) with the training data. Then, you classify each test sample with your Naive Bayes model by calculating $P(Y|W_1, W_2, ...W_m)$ while applying the Naive Bayes assumption.

Hints: You might need to smooth (e.g., use a MAP with pseudo-count of 1 for each word) to avoid zero probability.
After finishing all these steps, answer the following questions

THERE ARE REFERENCE CODES POSTED ON PIAZZA IN A ZIP FILE.

1. **[3 Points]** List the top five most frequent words of both pos and neg classes of the training set. Now exclude all the stop words in "stopwords.txt" and list the top five moset frequent words of both pos and neg classes of the training set. What do you find out? Explain why not apply those "stopwords" as our features?

   This is only a reference, since there are different ways for pre-processing.

   positive:

   With all words: ('the', 172635), ('and', 89427), ('a', 83507), ('of', 76641), ('to', 66478)

   Without stop words: ('time', 6208), ('story', 6610), ("it's", 8536), ('movie', 18647), ('film', 20064)

   negative:

   With all words: ('the', 162710), ('a', 79144), ('and', 74086), ('of', 68789), ('to', 68727)

   Without stop words: ('time', 5990), ('bad', 7254), ("it's", 8537), ('film', 18365), ('movie', 24341)]

   Stop words can't be as features, since they have similar distribution in both classes. They are not representative for different classes.

2. **[5 Points]** Report the overall test accuracy, the confusion matrix and the F_1 score for both pos and neg classes. Confusion matrix is a matrix that each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class (or vice versa).

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{3}$$

$$precision = \frac{\sum true\ positive}{\sum(true\ positive\ + false\ positive)} \tag{4}$$

$$recall = \frac{\sum true\ positive}{\sum(true\ positive\ + false\ negative)} \tag{5}$$

accuracy: 0.81648

precision: 0.860029122679

recall: 0.756

f1: 0.804666212534

confusion matrix:

[10962 1538]

[3050 9450]

3. **[4 Points]** Does the assumption of Naive Bayes hold for our dataset? Why? Does the classification results confirm your answer? Why?

   No, it doesn't hold. For each class, some words are likely to appear together, such as "good" and "excellent" for positive class. The assumption of Naive Bayes is that each word is conditionally independent for each class, which is actually not true in the real world.

   Pick up word "funny" and "love" for positive class.

   Prove: Get $P(funny|positive) = 0.11624$, $P(love|positive) = 0.21344$, $P(love, funny|positive) = 0.02984$. Then we can find that $P(love, funny|positive) \neq P(love|positive) \times P(funny|positive)$

   Interesting thing is that although the assumption doesn't hold, the performance on text classification is actually pretty descent

4. **[3 Points]** Compared to the results of homework 1 with logistic regression model, which method is better? How can we improve the accuracy of our Naive Bayes model?

   Here is a reference answer. This is an open-end question.

   For two methods, they both get similar accuracy. Naive Bayes makes strong assumption, with less computation. For efficiency, Naive Bayes is faster.

   Tuning smoothing parameter α, get more training data etc.

Problem 2: Can we make dumb learners smart? [10 Points] (Weixiang and Yiting)

In this problem, you will hopefully gain more insights into adaboost, and how it performs in the real world. More specifically, You will apply adaboost to make binary classification with the Johns Hopkins University Ionosphere database from UCI Machine Learning Repository. **You are free to use any adaboost third-party libraries (e.g., sklearn). But you must write your own code to evaluate accuracy and error.**

Dataset

The dataset comprises of one file "ionosphere.txt".This radar data was collected by a system in Goose Bay, Labrador. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere. There are total 351 samples. For each sample, there are 34 elements as features and the 35th element is the label, where "b" means "Bad" and "g" means "Good".

Exercises

In this problem, you are no longer provided with perfect balanced train and test sets (That's what the real world problem usually looks like!). You are supposed to split the dataset such that 80% of the samples (both "Good" and "Bad" classes) are for training and the remaining 20% are for testing. You are welcome to use any split strategies and report what you find in your writeup. Then you need to train your adaboost model (you can use any weak learners) and test it with your test data.

After finishing all these steps, answer the following questions.

1. **[3 Points]** Report your weak learner, best train accuracy and best test accuracy.

   This is just a reference answer. You can choose different weak learner.

   Weak learner: Decision Stump

   Best Train Accuracy: 100%

   Best Test Accuracy: $\approx 95.5\%$

2. **[4 Points]** Plot train error and test error for varying values of t from 0 to 100, where t is the number of iteration cycles.

   This is just a reference answer. Based on different weak learners, you may have different results.

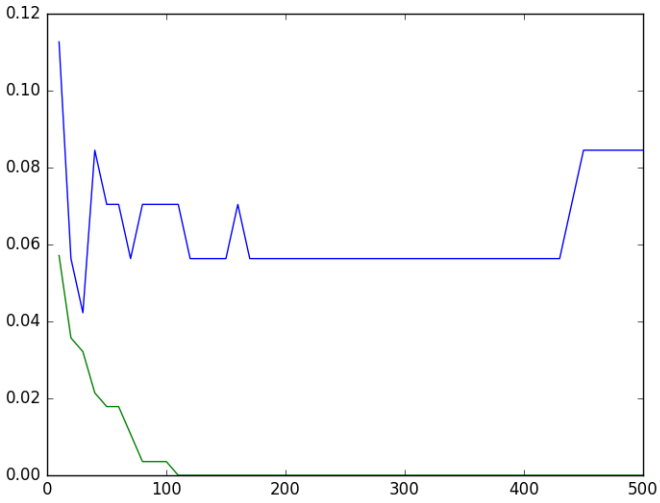

Figure 1: Test Accuracy vs. Train Accuracy

3. **[3 Points]** Intuitively, is adaboost robust to overfitting? Does your results match your intuition? Why (explain in 1 - 2 sentences)?

   The adaboost is robust to overfitting. According to my result, the overfitting occurs only when iteration number is too high. Within 100 iterations, overfitting doesn't occur.

Submission

For the entire programming exercise, please turn in your codes in a **single zipped folder** that contains two subfolders and each folder contains corresponding source code files for each problem.