CS6140: Machine Learning

Homework Assignment # 4

Assigned: 11/10/2024 *Due: 11/25/2024, 11:59pm, through Canvas*

Four problems, 160 points in total. Good luck!
Prof. Predrag Radivojac, Northeastern University

Problem 1. (40 points) Consider two classification concepts given in Figure 1, where $x \in \mathcal{X} = [-6, 6] \times [-4, 4]$, $y \in \mathcal{Y} = \{-1, +1\}$ and $p(y|x) \in \{0, 1\}$ is defined in the drawing.

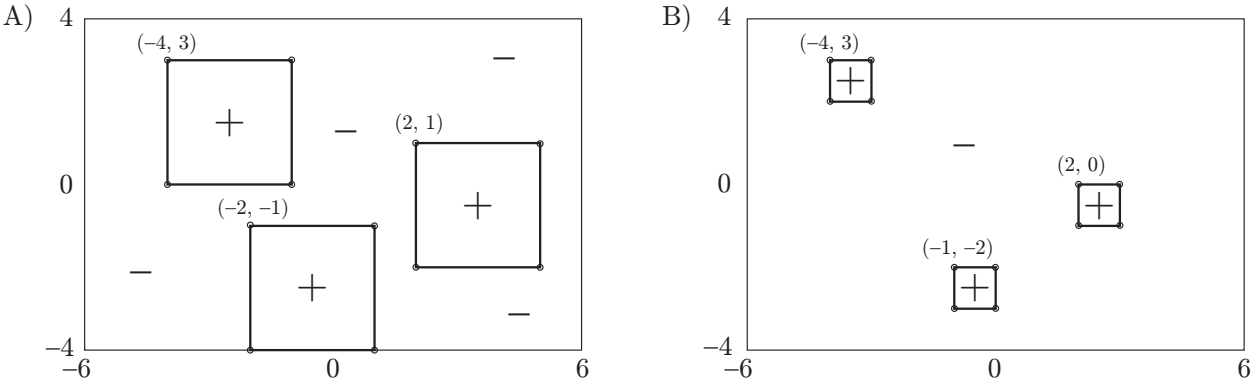

Figure 1: Two concepts where examples that fall within any of the three 3×3 (panel A) or 1×1 (panel B) squares are labeled positive and the remaining examples (outside each of the squares but within $\mathcal{X}$) are labeled negative. The position of the point $x = (x_1, x_2)$ in the upper left-hand corner for each square is shown in the picture. Consider horizontal axis to be x_1 and vertical axis as x_2.

Your experiments in this question will rely on generating a data set of size $n \in \{250, 1000, 10000\}$ drawn from a uniform distribution in $\mathcal{X}$ and labeled according to the rules from Figure 1; e.g., $P(Y = 1|x) = 0.99$ if x that was randomly drawn is inside any of the three squares in either of the two panels, and $P(Y = 1|x) = 0.03$ otherwise (notice that this introduces a certain amount of asymmetric noise in the data). The goal of the following two problems will be to train and evaluate classifiers created from the data generated in this way. You can use any library you want in this assignment and do programming in Python, MATLAB, or R. Your code should be easy to run for each question and sub-question below so that we can replicate your results to the maximum extent possible.

Consider single-output feed-forward neural networks with one or two hidden layers such that the number of hidden neurons in each layer is $h_1 \in \{1, 4, 12\}$ and $h_2 \in \{0, 3\}$, respectively, with $h_2 = 0$ meaning that there is no second hidden layer. Consider one of the standard objective functions as your optimization criterion and use early stopping and regularization as needed. Consider a hyperbolic tangent activation function in each neuron and the output but you are free to experiment with others if you'd like to. For each of the architectures, defined by a parameter combination (h_1, h_2), evaluate the performance of each model using classification accuracy, balanced accuracy, and area under the ROC curve as your performance criteria. To evaluate the performance of your models use cross-validation. However, to evaluate the performance of performance evaluation, generate another very large data set on a fine grid in $\mathcal{X}$. Then use the predictions

from your trained model on all these points to determine the "true" performance. You can threshold your predictions in the middle of your prediction range (i.e., at 0.5 if you are predicting between 0 and 1) to determine binary predictions of your models and to then compare those with true class labels you generated on the fine grid.

Provide meaningful comments about all aspects of this exercise (performance results for different network architectures, accuracy in and of cross-validation, true accuracy when you disregard class-label noise, run time, etc.). The comments should not just re-state the results but rather capture trends and give reasoning as to why certain behavior was observed.

Problem 2. (20 points) Prove representational equivalence of a three-layer neural network with linear activation function in all neurons and a single-layer layer neural network with the same activation function. Assume a single-output network.

Problem 3. (20 points) We saw in class a proof that neural networks can approximate posterior probabilities in binary classification when they are designed to minimize the sum-of-squared-errors loss. The proof was adapted from the following paper: Rojas, R. A short proof of the posterior probability property of classifier neural networks. *Neural Computation* (1996) 8(1):41-43. Derive that the same outcome can be obtained using maximum likelihood estimation.

Problem 4. (80 points) Assess the role of biased data on binary classification performance. This will be accomplished using synthetic data. First, consider the following class-conditional distributions

$$p(\boldsymbol{x}|Y=y) = \sum_{k=1}^{m} w_{ky} \cdot \mathcal{N}(\boldsymbol{\mu}_{ky}, \boldsymbol{\Sigma}_{ky}),$$

where $0 < w_{ky} \leq 1$, $\sum_{k=1}^{m} w_{ky} = 1$, $\boldsymbol{x} \in \mathbb{R}^2$, $y \in \{0, 1\}$ and $m \geq 1$. Then, generate a training set $\mathcal{D}$ of $n_{\mathcal{D}}$ input-output pairs to satisfy

$$p(\boldsymbol{x}) = p(\boldsymbol{x}|Y=0)P(Y=0) + p(\boldsymbol{x}|Y=1)P(Y=1),$$

where $P(Y=1) \in (0, 0.5)$. After the training set is generated, create different biased test sets on which you will evaluate the quality of your training and performance estimation. The test set $\mathcal{T}$ will consist of $n_{\mathcal{T}}$ input-output examples generated as follows:

a) (20 points) Your test data is unbiased. That is, $\mathcal{T}$ is constructed from $\bar{p}(\boldsymbol{x}, y) = p(\boldsymbol{x}, y)$.

b) (20 points) Your test data is non-representative according to the label-shift bias model; i.e.,

$$\bar{p}(\boldsymbol{x}) = p(\boldsymbol{x}|Y=0)\bar{P}(Y=0) + p(\boldsymbol{x}|Y=1)\bar{P}(Y=1),$$

Note that the class-conditionals are unchanged in the test data, but that the class-priors are different; i.e., $\bar{P}(Y=0) \neq P(Y=0)$.

c) (20 points) Your test data is non-representative according to the covariate-shift bias model, where the posteriors are unchanged but the marginal data distributions are distinct; i.e., $\bar{p}(Y=1|\boldsymbol{x}) = p(Y=1|\boldsymbol{x})$ and $\bar{p}(\boldsymbol{x}) \neq p(\boldsymbol{x})$. Make sure that you clearly explain how you ensured that this condition holds.

d) (20 points) Your test data is biased according to a flexible bias model; i.e.,

$$\bar{p}(\boldsymbol{x}) = \bar{p}(\boldsymbol{x}|Y=0)\bar{p}(Y=0) + \bar{p}(\boldsymbol{x}|Y=1)\bar{p}(Y=1),$$

where

$$\bar{p}(\boldsymbol{x}|Y=y) = \sum_{k=1}^{\bar{m}} \bar{w}_{ky} \mathcal{N}(\bar{\boldsymbol{\mu}}_{ky}, \bar{\boldsymbol{\Sigma}}_{ky}),$$

where all model parameters (number of components, weights, mean vectors, covariance matrices) are generally distinct between training data and test data.

The goal of the exercise is to train your models and estimate their accuracy on $\mathcal{D}$ but then use test set $\mathcal{T}$ to check the performance of your trained models "in the wild". You can select $n_{\mathcal{D}}$, $n_{\mathcal{T}}$, and all model parameters manually to allow for stable estimation (e.g., $n_{\mathcal{D}}$ and $n_{\mathcal{T}}$ can be large), but note that the area under the ROC curve (AUC) of your best classifier on the unbiased data in part (a) should not exceed 0.95 or be below 0.65. Then train at least 3 different prediction models to evaluate their sensitivity to data biases. You must use logistic regression, naive Bayes, and at least one non-linear neural network with at least two layers. You can use other models as well.

Use 10-fold cross-validation protocol on $\mathcal{D}$ and AUC to estimate the performance of your model. Then give AUC on the biased test set. Repeat your data generation and evaluation multiple times to verify that the test performance is within confidence intervals of the estimated performance. Discuss your results, visualize data distributions (the data is 2D) and potentially (not mandatory) use data sets with different level of overlap between class-conditional distributions (between positives and negatives) to strengthen your conclusions on the impact of problem difficulty on the impact of bias.

Directions and Policies

Submit a single package containing all answers, results and code. Your submission package should be compressed and named firstnamelastname.zip (e.g., predragradivojac.zip). In your package there should be a single pdf file named main.pdf that will contain answers to all questions, all figures, and all relevant results. Your solutions and answers must be typed[1] and make sure that you type your name and Northeastern username (email) on top of the first page of the main.pdf file. The rest of the package should contain all code that you used. The code should be properly organized in folders and subfolders, one for each question or problem. All code, if applicable, should be turned in when you submit your assignment as it may be necessary to demo your programs to the teaching assistants. Use Matlab, Python or R.

Unless there are legitimate circumstances, late assignments will be accepted up to 5 days after the due date and graded using the following rules:

on time: your score $\times$ 1

1 day late: your score $\times$ 0.9

2 days late: your score $\times$ 0.7

3 days late: your score $\times$ 0.5

4 days late: your score $\times$ 0.3

5 days late: your score $\times$ 0.1

For example, this means that if you submit 3 days late and get 80 points for your answers, your total number of points will be $80 \times 0.5 = 40$ points.

All assignments are individual, except when collaboration is explicitly allowed. **All text must be be your own or, for group assignments (when explicitly permitted), by the members of the group. All sources used for problem solution must be acknowledged**; e.g., web sites, books, research papers, personal communication with people, etc. Academic honesty is taken seriously! For detailed information see Office of Student Conduct and Conflict Resolution.

[1] We recommend Latex; in particular, TexShop-MacTeX combination for a Mac and TeXnicCenter-MiKTex combination on Windows. An easy way to start with Latex is to use the freely available Lyx. You can also use Microsoft Word or other programs that can display formulas professionally.