EE 660 Homework 7 (Week 14) Posted: 11/22/2021
Jenkins Due: Wed., 12/1/2021, 11:59 PM PST

1. In this problem you will implement transfer learning (TL) based on importance weighting, and compare it with supervised learning (SL). You will work with data from the *TL_data* folder. There are 5 files. They contain source training and test data, labeled and unlabeled target training data, and target test data. The target data had a covariate shift with respect to the source data. In all items below, you should use the *AdaBoostClassifier* from sklearn with default parameters.

   a) Let's start by estimating the classifier's performance on this data with a regular SL problem. Train a classifier on the source training data. Report its accuracy on the source test data.

For parts (b)-(d) below, you will use standard SL techniques but applied to the TL problem (3 different approaches).

   b) Use the classifier trained in item (a) to predict labels of the target test data. Report the accuracy.

   c) Train a classifier only on the labeled target training data. Report its accuracy on the target test data.

   d) Train a classifier on the union of source training and labeled target training data. Report its accuracy on the target test data.

   e) Compare the results of (a)-(d). Explain any differences (and any lack of differences) in accuracy.

For parts (f)-(g), you will use TL techniques on the TL problem.

   f) Let's assume source and target domain features follow multivariate normal distributions with different parameters. Estimate the mean and covariance matrix of each domain. Hint: you can use sklearn's GaussianMixture class. You can proceed in two ways.

      i. Estimate the two means and two covariance matrices simultaneously. This can be done by letting the Gaussian mixture model estimator know there are 2 components in the mixture and providing the entire training data (source + labeled target + unlabeled target) to it.

      ii. Estimate each mean and covariance matrix individually. This can be done by training two separate Gaussian mixture model estimators, each with one component density. One estimator will receive only the source training data and the other estimator will receive all the target training data.

   Which method is likely to yield better results, i.e., means and covariance matrices closer to the true value? Justify your answer.

   Provide the values for the mean and covariance matrix of each domain.

g) Now that you have the parameters of each domain, you can compute the weight of each data samples as $w_i = p_T(x_i)/p_S(x_i)$. Train a classifier on the union of source training and labeled target data using these weights. Report the accuracy on the target test data.

h) Compare results of (g) with results of (b)-(d). Explain any differences and any lack of difference.

2. *EM for semi-supervised learning.* Consider a 2-class semi-supervised learning problem in which there are l labeled samples and $u = 1$ unlabeled sample. (For example, think of being given l labeled samples, and then acquiring unlabeled samples one at a time.) There is 1 feature, and each class is modeled as a Gaussian:

$$p\left(x \middle| y = c, \underline{\theta}\right) = N\left(x \middle| \mu_c, \sigma_c{}^2\right), \quad c = 1,2$$

In the parts below, you will use EM to estimate the means μ_1 and μ_2 . You may assume the priors and variances are given constants. Generally the subscripts *h* and *i* will indicate unlabeled and labeled samples, respectively.

In this problem, parts (a)-(d) are to be done by hand. Part (e) can be done by hand or computer; part (f) is best done by computer.

a) Consider the t^{th} iteration of EM. Derive the E step in terms of given quantities: that is, starting from

$$p\left(\mathcal{H} \middle| \mathcal{D}, \underline{\theta}^{(t)}\right) = p\left(y_h = c_h \middle| x_h, \underline{\theta}^{(t)}\right) = \gamma_{hc_h}^{(t)}, \quad c_h = 1,2$$

show that:

$$\gamma_{hc_h}^{(t)} = \frac{\pi_{c_h}}{\alpha_h^{(t)}\sqrt{2\pi\sigma_{c_h}{}^2}} \exp\left\{-\frac{\left(x_h - \mu_{c_h}^{(t)}\right)^2}{2\sigma_{c_h}{}^2}\right\}$$

in which $\pi_{c_h} \triangleq p\left(y_h = c_h \middle| \underline{\theta}^{(t)}\right) = p\left(y_h = c_h\right)$. Also, find $\alpha_h^{(t)}$.

In parts (b)-(d), you will derive the M step formulas, also for the t^{th} iteration of EM.

b) First, show that

$$p\left(\mathcal{D},\mathcal{H}\middle|\underline{\theta}\right)=p\left(x_h\middle|y_h=c_h,\underline{\theta}\right)\pi_{c_h}\prod_{i=1}^{l}p\left(x_i\middle|y_i=c_i,\underline{\theta}\right)\pi_{c_i}$$

in which $\pi_{c_i}=p\left(y_i=c_i\middle|\underline{\theta}\right)=p\left(y_i=c_i\right)$ and similarly for π_{c_h}.

c) Take $\ln p\left(\mathcal{D},\mathcal{H}\middle|\underline{\theta}\right)$ from your result of (b), plug in for the normal densities, and drop any additive terms that are constants of $\underline{\theta}$. Then plug in to the M equation:

$$\underline{\theta}^{(t+1)}=\arg\max_{\underline{\theta}}\ \mathsf{E}_{\mathcal{H}\middle|\mathcal{D},\underline{\theta}^{(t)}}\left\{\ln p\left(\mathcal{D},\mathcal{H}\middle|\underline{\theta}\right)\right\}$$

$$=\arg\max_{\underline{\theta}}\ \sum_{c_h=1}^{2}\gamma_{hc_h}^{(t)}\ln p\left(\mathcal{D},\mathcal{H}\middle|\underline{\theta}\right)$$

and simplify to get:

$$\underline{\theta}^{(t+1)}=\arg\max_{\underline{\theta}}\ \left\{\sum_{c_h=1}^{2}\gamma_{hc_h}^{(t)}\left[-\frac{\left(x_h-\mu_{c_h}\right)^2}{{\sigma_{c_h}}^2}\right]+\sum_{i=1}^{l}\left[-\frac{\left(x_i-\mu_{c_i}\right)^2}{{\sigma_{c_i}}^2}\right]\right\}$$

in which a constant multiplicative factor of $\frac{1}{2\pi}$ has been dropped. (**Hint:** you may find it useful to use $\gamma_{h1}^{(t)}+\gamma_{h2}^{(t)}=1$.)

d) Re-write your result of part (c) to express it in terms of $\mu_1,\mu_2\ {\sigma_1}^2,{\sigma_2}^2$. (**Hint**: you might find it useful to use the indicator function.) Then solve for $\underline{\theta}^{(t+1)}=\left[\mu_1^{(t+1)},\mu_2^{(t+1)}\right]^T$. (**Hint:** find the argmax by taking $\frac{\partial}{\partial\mu_1}$ and setting equal to 0; similarly for μ_2.) Let $l_1=$ number of labeled samples with label $c_i=1$, and $l_2=$umber of labeled samples with label $c_i=2$. (Note that $\gamma_{hc_h}^{(t)}$ is constant of μ_1 and μ_2 because it used the (constant) estimates $\mu_1^{(t)}$ and $\mu_2^{(t)}$ from the E step.)

e) Given: $\pi_1=\pi_2=0.5,\ {\sigma_1}^2={\sigma_2}^2=1$; data as follows:

labeled data $\left\{\left(x_i,y_i\right)\right\}_{i=1}^{l}=\left\{(1,1),(2,1),(4,2)\right\}$; unlabeled sample $x_h=3$.

Suppose the values for $\underline{\theta}$ at the beginning of the t^{th} iteration of EM are: $\mu_1^{(t)} = 1.5$, $\mu_2^{(t)} = 4.0$.

(i) Calculate the responsibilities $\gamma_{h1}^{(t)}$ and $\gamma_{h2}^{(t)}$ from the E step (using part (a));

(ii) Calculate the new mean estimates $\mu_1^{(t+1)}$ and $\mu_2^{(t+1)}$ from the M step (using part (d) result).

Tip: While not required for part (e), you may find it useful to do the calculations by computer, so that your code can be used for part (f) also.

f) Run more iterations (by computer), until $\mu_1^{(t+1)}$ and $\mu_2^{(t+1)}$ converge (until they change only a small amount from one iteration to the next – choose a suitable threshold). Plot $\mu_1^{(t+1)}$ and $\mu_2^{(t+1)}$ vs. *t*, as well as $\gamma_{h1}^{(t)}$ and $\gamma_{h2}^{(t)}$ vs. *t*. (You are not required to compute $p\left(\mathcal{D}\middle|\underline{\theta}^{(t)}\right)$ in this problem.) Give your final values for $\mu_1^{(t+1)}$, $\mu_2^{(t+1)}$, $\gamma_{h1}^{(t)}$, and $\gamma_{h2}^{(t)}$.

3. In this problem you will explore semi-supervised learning using S3VM, and compare to supervised learning. Throughout this problem, use the qns3vm code available at the course's page under Week 12 with parameters kernel_type='Linear' and 'lam=1.0' (*c.f.* Discussion 12 for more information).

Note: if you get a "PendingDeprecationWarning" that halts the execution of the code, add the line:
"warnings.filterwarnings('ignore', category=PendingDeprecationWarning)"
at the start of your code. The SVM parameters should also be set to kernel='linear' and C=1.0.

Use the data inside the *SSL_data* folder. Load the data files named *ssl_train_data* and *test_data*. On each of them, the first 10 columns are the features, i.e., X_{train} and X_{test}, and the last column represents the true label, i.e., y_{train} and y_{test}. There is a total of 200 training samples and the classes are $y_i = \{0,1\}$. Note that the qns3vm code expects classes $\{-1, 1\}$ and adjust accordingly.

(a) To get an estimate of the best-case scenario, let's start with a dataset that is entirely labeled. Train an SVM classifier on the entire train data and compute its accuracy on the test data.

(b) Now let's assume a scenario where only a few samples of the training data are labeled. Select only the first $2L$ samples of the training set (you can note that the

dataset was built in a way that the first $2L$ samples always contain L samples of each class), train an SVM classifier *only* on those first $2L$ samples, and report the accuracy on the test data for $L = [1, \dots, 10]$. Note that the test set does not change size.

(c) Next, let's repeat the scenario from (b), but let's make use of the unlabeled data. Train an S3VM model on the entire data ($2L$ labeled samples and $200 - 2L$ unlabeled samples), and the report the accuracy on the test data, for $L = [1, \dots, 10]$.

(d) Plot your results of (b) and (c) on a single plot, showing accuracy (percent correct classification on test set) vs. $N_L = 2L$.

(e) Interpret your result of (d).

   a. In what ways, if any, are they what you expected? Explain why you expected them to be so.

   b. In what ways, if any, are they different than what you expected? Explain what you expected that is different, and hypothesize why the difference.