**D.2.1 Selecting a subset of the target population to apply the PTR from the source population:**

First, we need to estimate how many samples in the target population are similar (from) the source population. This problem is closely related to the “Mixture Proportion Estimation”.62 We adapt the methods proposed by Ramaswamy et. al via Kernel Embedding of Distributions.62 Mathematically,

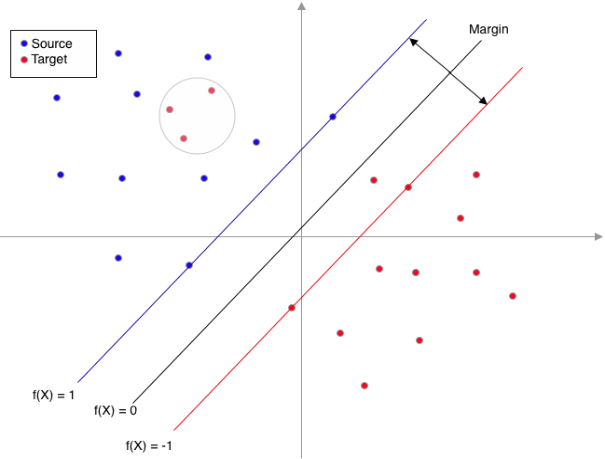


Figure 3. A subset of target population (the red dots in the circle) is more similar to the source population, then we should apply the PTR learnt from source population to these samples.

suppose *,* where *G* is the distribution of the covariates in the target population (related to the treatment decision) which is a mixture of component *F* (source population) and component *E* (unknown/unobserved) . Then, the task is estimating the proportion parameter which represents of proportion of target population that is from the source population. The authors proposed kernel mean based gradient thresholder to estimate , which involves quadratic programming and kernel embedding.62

Given the estimated proportion, we need to rank the samples in the target population based on how similar are these samples compared to the source population. To do so, we would separate the hyperplane to discriminate the source and target population. Then, the distances of samples in the target population to the separating hyperplane can be regarded as a measure of how similar these samples are to samples in the source population. Hence we could prioritize the target population to apply the PTR from the source population.

Suppose we have the pooled data set available. The study/population label , with +1 for source and -1 for target population. The predictor ***X*** is a p-dimensional vector, which are patients’ characteristics. We will build a classifier to separate the two populations based on ***X***. This is a classification problem and there are many techniques available in the literature. Meanwhile, as we use this classifier to select the samples in the target population that are similar to the source population, we propose to use the large margin classifiers.35 Given the pooled data set, a large margin classifier is trained to obtain a map , such that the predicted class label is assigned using the sign of . Note that when is positive, then the prediction is correct. The term is known as the functional margin, and is a separating hyperplane – one side of the hyperplane is the +1 class and the other side is the -1 class. Hence, the is the distance of a sample (with label and covariate ) to the separating hyperplane. Furthermore, this indicates that if the value of a sample from target population is positive and large, then that sample is more similar to the source population (Figure 3). Compared to other types of classifiers such as the nearest neighbor or random forest, the large margin interpretation of large margin classifier provides a direct measure of how similar a sample from the target population will be to samples in the source population. In this proposal, we will use the large-margin unified machine (LUM),63 a flexible and data-adaptive large margin that leverage the strength of multiple existing classifiers.

**Large-margin unified machine** **(LUM):** In general, the objective function of a large margin classifier can be written in the regularization framework of a loss plus a penalty.35 The loss is a measure of the goodness of fit between the model and data, and the penalty controls the complexity of the model to avoid overfitting. Specifically, the optimization problem of a large margin classifier can be expressed as follows: , where is the function class that all candidate solution functions belong to, is a regularization term penalizing the complexity of , is the loss function, and *λ* is a tuning parameter balancing the two terms. A natural loss function is the so called 0 − 1 loss with value 1 if , and otherwise, i.e. . However, the 0 − 1 loss is difficult for optimization due to its nonconvexity. Consequently, various convex surrogate loss functions have been proposed in the literature to alleviate the computational problem.64 For example, SVM uses hinge loss, penalized logistic regression uses logistic loss, and AdaBoost uses exponential loss.35 Recently, Liu et al. proposed LUM which is differentiable everywhere, hence it has some computational advantage, and our previous work has shown it is very powerful for other applications including classifying cancer subtypes.65 The LUM loss is indexed by two parameters a and c with the following explicit form:

The left piece of with is the same as the hinge loss used in the SVM. The right piece is a convex curve whose shape is controlled by with rate of decay controlled by . With and , LUM is equivalent to the standard SVM. With and fixed , LUM loss is a hybrid of the SVM and AdaBoost. For these reasons, LUM can be useful for both easy and difficult classification problems, which makes it suitable for prioritizing the selection of subsets in the target population to apply the PTR from the source population.