MTTTS16 Learning from Multiple Sources

5 ECTS credits

Autumn 2019, University of Tampere
Lecturer: Jaakko Peltonen

Lecture 9:
Domain adaptation and covariate shift

On this lecture:

- Setups where most of the training data comes from a different distribution than the test data

Part 1: Covariate Shift

Covariate shift

- Basic assumption in statistical learning: training data and test data are drawn from the same underlying distribution.
- Consider the setting: a set of real-valued training data pairs (x, y) is provided to train a model for a supervised learning problem (classification/regression/prediction).
- Additionally data of the form x is provided from one (or more) **test environments** where the model will be used.
- How should we predict a value of y given a value x from within a particular test environment?
- One simple example of covariate shift is sample selection bias: proportions of classes in training data are different than in future test data. (can be a big problem e.g. for political polls etc.)

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

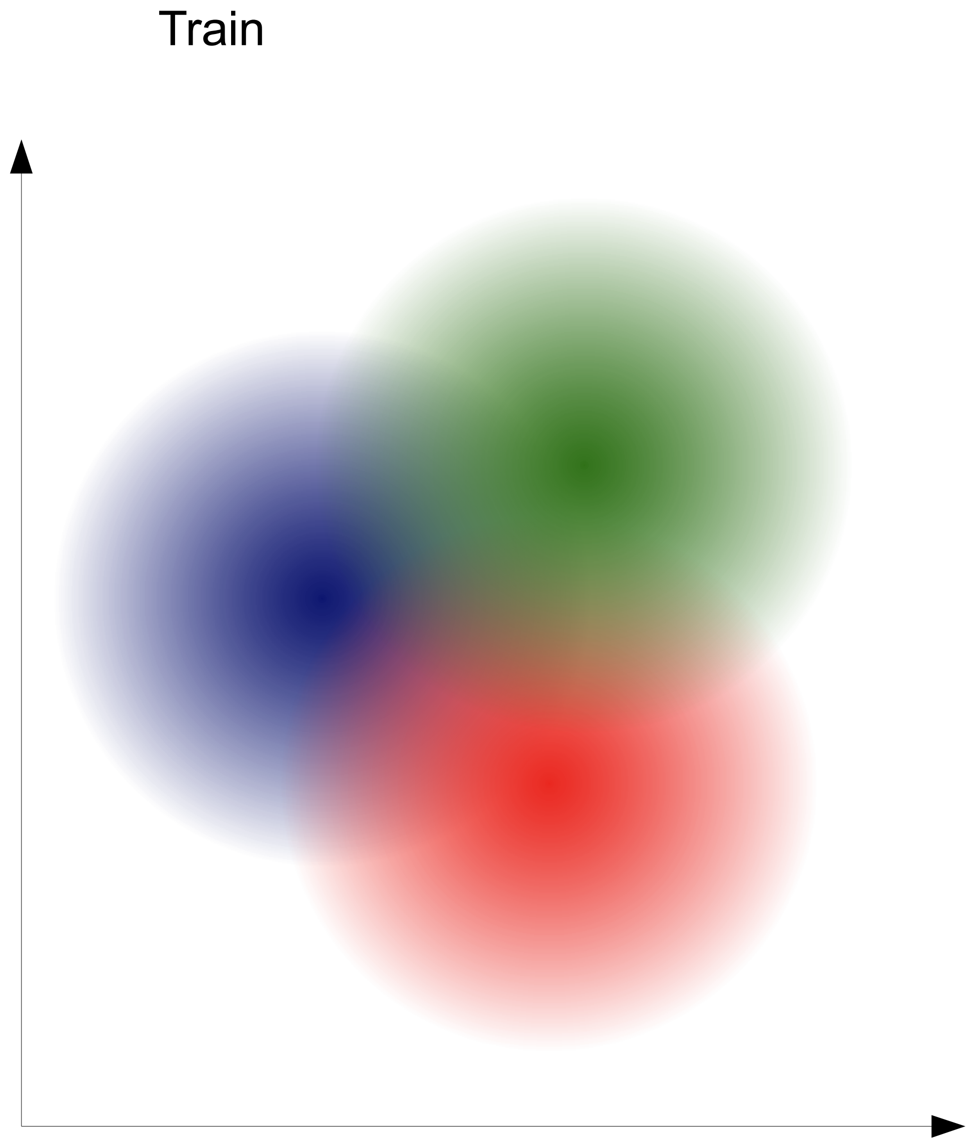
Covariate shift

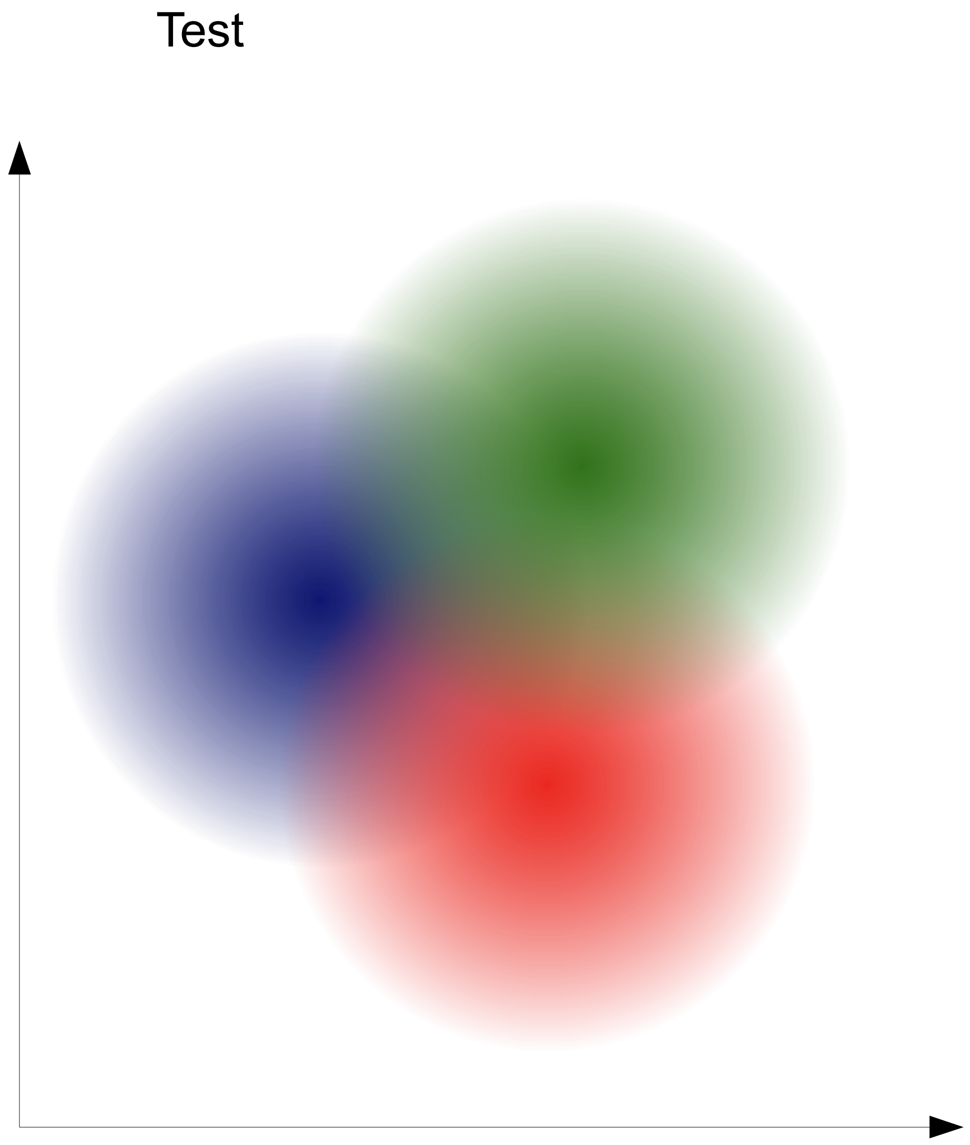
- Three different types of covariate shift:

1. Independent covariate shift: $P_{train}(\mathbf{y}|\mathbf{x}) = P_{test}(\mathbf{y}|\mathbf{x})$, but $P_{train}(\mathbf{x}) \neq P_{test}(\mathbf{x})$.
2. Dependent prior probability change: $P_{train}(\mathbf{x}|\mathbf{y}) = P_{test}(\mathbf{x}|\mathbf{y})$, but $P_{train}(\mathbf{y}) \neq P_{test}(\mathbf{y})$.
3. Latent prior probability change: $P_{train}(\mathbf{x}, \mathbf{y}|\mathbf{r}) = P_{test}(\mathbf{x}, \mathbf{y}|\mathbf{r})$ for all values of some latent variable $\mathbf{r}$, but $P_{train}(\mathbf{r}) \neq P_{test}(\mathbf{r})$.

- Suppose we are only interested in $P_{test}(\mathbf{y}|\mathbf{x})$, then covariate shift in case 1 has no effect on modeling
- Case 2 means class prior probabilities change (e.g. due to imbalanced sampling of training data)
- Case 3 involves a more general assumption

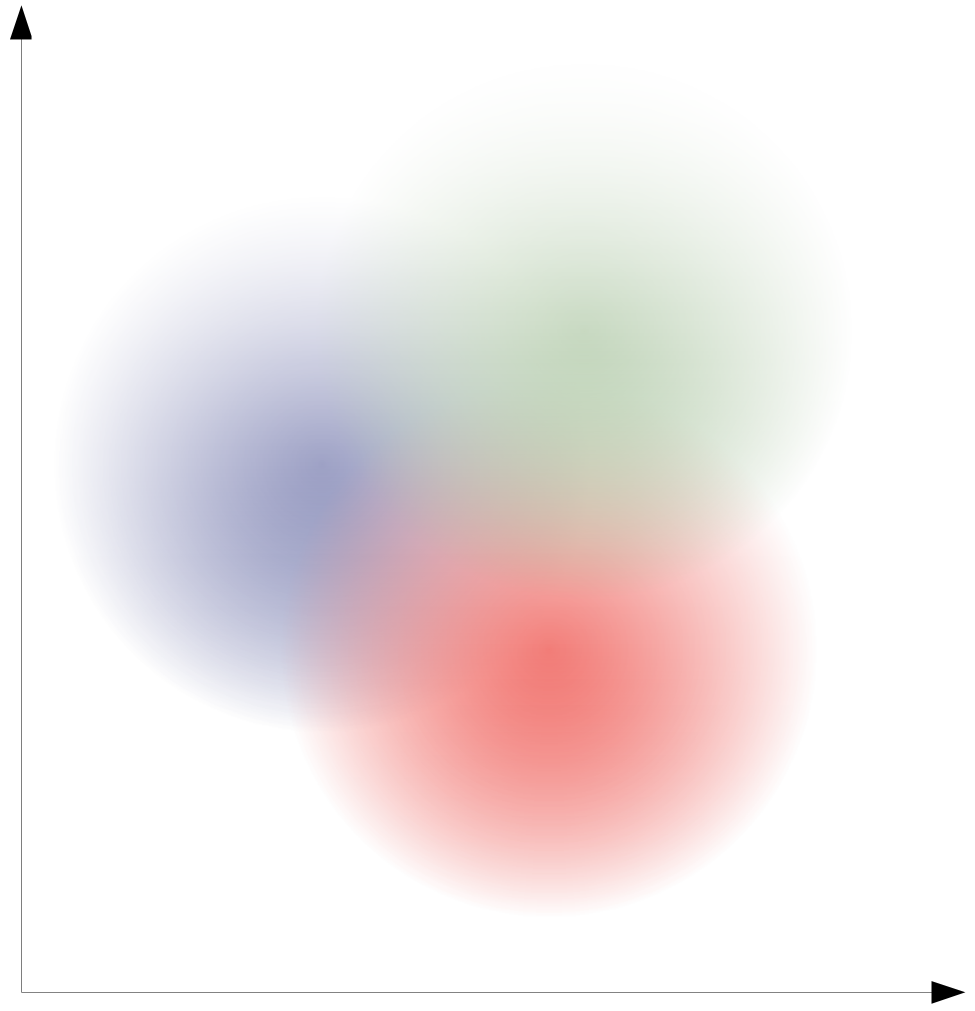
Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.


Train

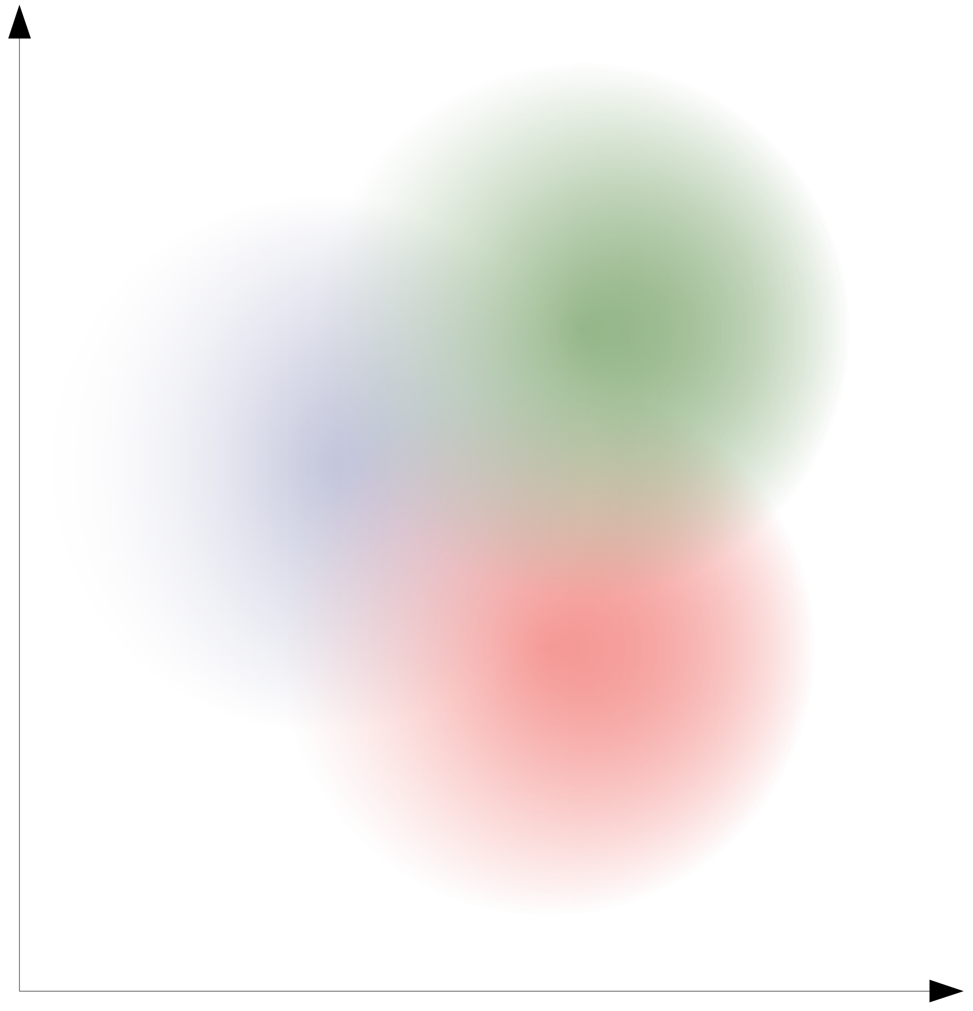

Test

INDEPENDENT COVARIATE SHIFT

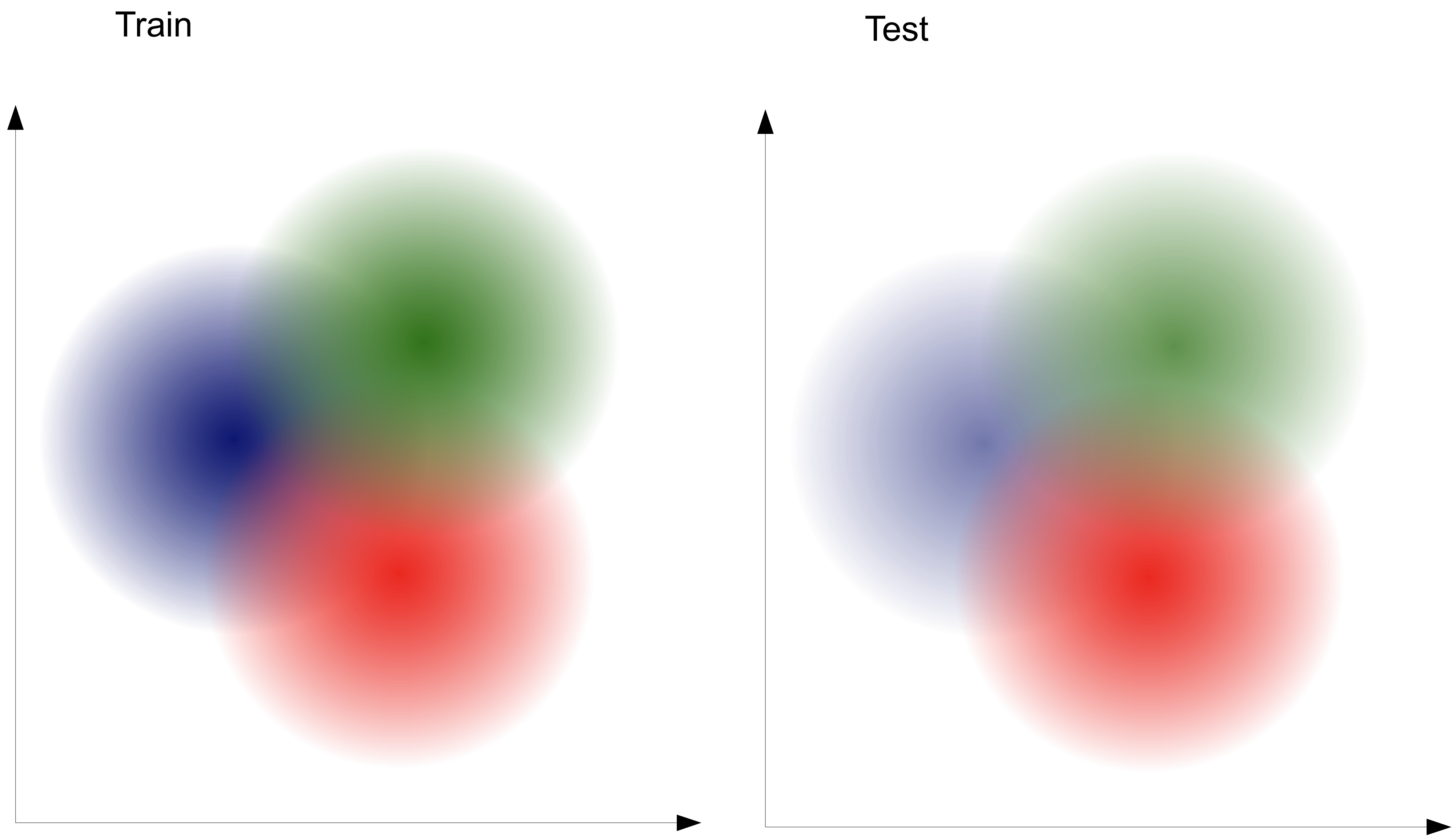
Train



Test



DEPENDENT PRIOR PROBABILITY CHANGE



Covariate shift

- Assume we have single training and single test set. Each sample x is assumed to come from one of several **data sources** using a **mixture distribution** corresponding to the source.
- The proportions of each of the sources varies across the training and test datasets.
- Here covariate shift affects the contribution of different sources to the data.
- There is a latent feature set (properties within each source) upon which each dataset is dependent. Variations between the two datasets depend on variation of the proportions of those latent features (sources) but the form of each source stays the same

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

- Sources in set 1 may occur in the test data, and potentially also in the training data, and are associated with regression model $P_1(\mathbf{y}|\mathbf{x})$
- Sources in set 2 occur only in the training data, and are associated with regression model $P_2(\mathbf{y}|\mathbf{x})$
- Set 1 has M distributions $P_{1t}(\mathbf{x})$, t=1,...,M, associated with $P_1(\mathbf{y}|\mathbf{x})$
- Set 2 has M$_2$ distributions $P_{2t}(\mathbf{x})$, t=1,...,M$_2$, associated with $P_2(\mathbf{y}|\mathbf{x})$
- Training data distribution:

relative proportions in source set 1

$$P_D(\mathbf{x}) = \sum_t \beta_1 \gamma_{1t}^D P_{1t}(\mathbf{x}) + \sum_t \beta_2 \gamma_{2t}^D P_{2t}(\mathbf{x})$$

proportions of train vs. test distributions in training data

relative proportions in source set 2

- Test data distribution:

$$P_T(\mathbf{x}) = \sum_t \gamma_{1t}^T P_{1t}(\mathbf{x})$$

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

- Full parametric model of observed data and latent values:

$$P(\{i^\mu, \mathbf{y}^\mu, \mathbf{x}^\mu | \mu \in D\}, \{i^\nu, \mathbf{x}^\nu | \nu \in T\} | \boldsymbol{\beta}, \boldsymbol{\Theta}, \boldsymbol{\Omega}) =$$
$$\prod_{\mu \in D} P(s^\mu|\boldsymbol{\beta}) P(t^\mu|\boldsymbol{\gamma}, s^\mu) P_{s^\mu t^\mu}(\mathbf{x}^\mu|\Omega_{t^\mu}) P_{s^\mu}(\mathbf{y}^\mu|\mathbf{x}^\mu, \boldsymbol{\Theta}) \prod_{\nu \in T} P(t^\nu|\boldsymbol{\gamma}) P_{1t^\mu}(\mathbf{x}^\nu|\boldsymbol{\Omega})$$

- Here μ are indices of training data points, $i^\mu = (s^\mu, t^\mu)$ are the indices of the source-set (1 or 2) and the component within the source-set used to generate a training data point, y^μ, x^μ are the output and input values for the data point
- Here ν are indices of test data points, $i^\nu = (t^\mu)$ are the indices of the component within the test-source used to generate a test data point, x^ν are input values for the test data point
- The parameters are the regression parameters $\boldsymbol{\Theta}$, mixture parameters $\boldsymbol{\Omega}$, and mixing proportions $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$.

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

- Full parametric model of observed data and latent values:

$$P(\{i^\mu, \mathbf{y}^\mu, \mathbf{x}^\mu | \mu \in D\}, \{i^\nu, \mathbf{x}^\nu | \nu \in T\} | \boldsymbol{\beta}, \boldsymbol{\Theta}, \boldsymbol{\Omega}) =$$
$$\prod_{\mu \in D} P(s^\mu | \boldsymbol{\beta}) P(t^\mu | \boldsymbol{\gamma}, s^\mu) P_{s^\mu t^\mu}(\mathbf{x}^\mu | \Omega_{t^\mu}) P_{s^\mu}(\mathbf{y}^\mu | \mathbf{x}^\mu, \boldsymbol{\Theta}) \prod_{\nu \in T} P(t^\nu | \gamma) P_{1t^\mu}(\mathbf{x}^\nu | \boldsymbol{\Omega})$$

- A parametric model like this can be optimized by the expectation-maximization algorithm to maximize the marginal likelihood of the observed data with respect to parameters $(\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}, \boldsymbol{\Omega})$
- This yields E-step:

$$\alpha_{st}^{\mu} = \frac{\beta_s \gamma_{st}^{D} P_{st}(\mathbf{x}^\mu | \boldsymbol{\Omega}) P_s(\mathbf{y}^\mu | \mathbf{x}^\mu, \boldsymbol{\Theta})}{\sum_{s,t} \beta_s \gamma_{st}^{D} P_{st}(\mathbf{x}^\mu | \boldsymbol{\Omega}) P_s(\mathbf{y}^\mu | \mathbf{x}^\mu, \boldsymbol{\Theta})}$$

proportional responsibility of source s, component t for a training data point

$$\alpha_{1t}^{\nu} = \frac{\gamma_{1t}^{T} P_{1t}(\mathbf{x}^\mu | \boldsymbol{\Omega})}{\sum_t \gamma_{1t}^{T} P_{1t}(\mathbf{x}^\mu | \boldsymbol{\Omega})}$$

proportional responsibility of component t for a test data point

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

- M-step depends on the form of the mixture components. If they are Gaussian of the form $N(\mathbf{x}; \mathbf{m}_{st}, \mathbf{K}_{st})$

- then the M-step becomes

component mean

$$\mathbf{m}_{st} = \frac{\sum_{\mu\in(D,T)} \alpha^{\mu}_{st}\mathbf{x}^{\mu}}{\sum_{\mu\in D,T} \alpha^{\mu}_{st}}, \mathbf{K}_{st} = \frac{\sum_{\mu\in(D,T)} \alpha^{\mu}_{st}(\mathbf{x}^{\mu}-\mathbf{m}_{st})(\mathbf{x}^{\mu}-\mathbf{m}_{st})^{T}}{\sum_{\mu\in D,T} \alpha^{\mu}_{st}}$$

source proportion

$$\beta_s = \frac{1}{|D|}\sum_{\mu\in D,t} \alpha^{\mu}_{st}, \gamma^{D}_{st} = \frac{1}{|D|}\sum_{\mu\in D}\frac{\alpha^{\mu}_{st}}{\beta_s}, \gamma^{T}_{1t} = \frac{1}{|T|}\sum_{\nu\in T}\alpha^{\nu}_{1t}$$

comp. proportion in training

comp. proportion in testing

$$\mathbf{f}_s = \arg\min_{\mathbf{f}_s}\left[\sum_{\mu,t}\alpha^{\mu}_{st}(\mathbf{f}_s(\mathbf{x}^{\mu})-\mathbf{y}^{\mu})^{T}(\mathbf{f}_s(\mathbf{x}^{\mu})-\mathbf{y}^{\mu})\right]$$

regression functions, precise parameterization depends on form of regressor (e.g. linear regression)

- Test data is associated with one regression model $P_1(\mathbf{y}|\mathbf{x})$.. . Predictive distribution for the test set is the learnt predictor $P_1(\mathbf{y}|\mathbf{x}_i)$ for each test sample $\mathbf{x}_i$

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

- Special case: **importance weighted least squares**
- Suppose P_D and P_T are known, and the source set1 contains just the one component which must be P_T by definition.
- Suppose the two regressors have a large and identical variance Σ. In this simple case, we do not need to know the actual test points (they are only used to infer the test distribution, which is assumed given here).
- The M step update only involves an update to the regressor
- With an assumption $P(\mathbf{y}^\mu|\mathbf{x}^\mu,\Theta_1) \approx P(\mathbf{y}^\mu|\mathbf{x}^\mu,\Theta_2)$ the equations become

- E-step: M-step:

$$\alpha^\mu \approx \frac{P_T(\mathbf{x}^\mu)\beta_1}{P_D(\mathbf{x}^\mu)}$$

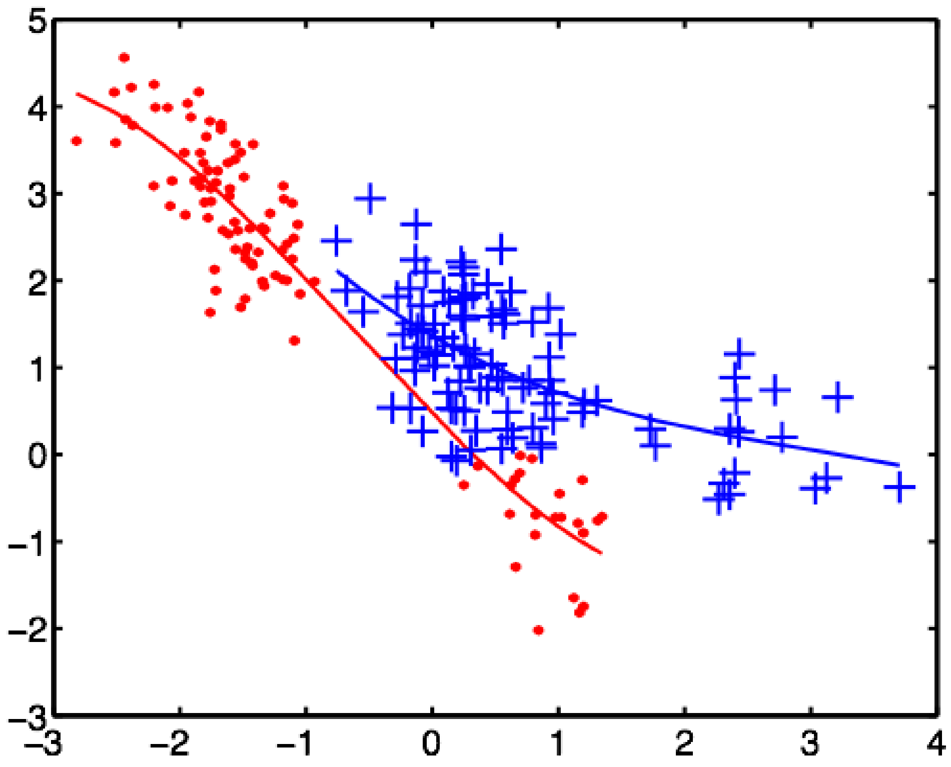
$$\mathbf{f}_1 = \arg\min_{\mathbf{f}_1}\left[\sum_\mu \alpha^\mu(\mathbf{f}_1(\mathbf{x}^\mu)-\mathbf{y}^\mu)^T(\mathbf{f}_1(\mathbf{x}^\mu)-\mathbf{y}^\mu)\right]$$

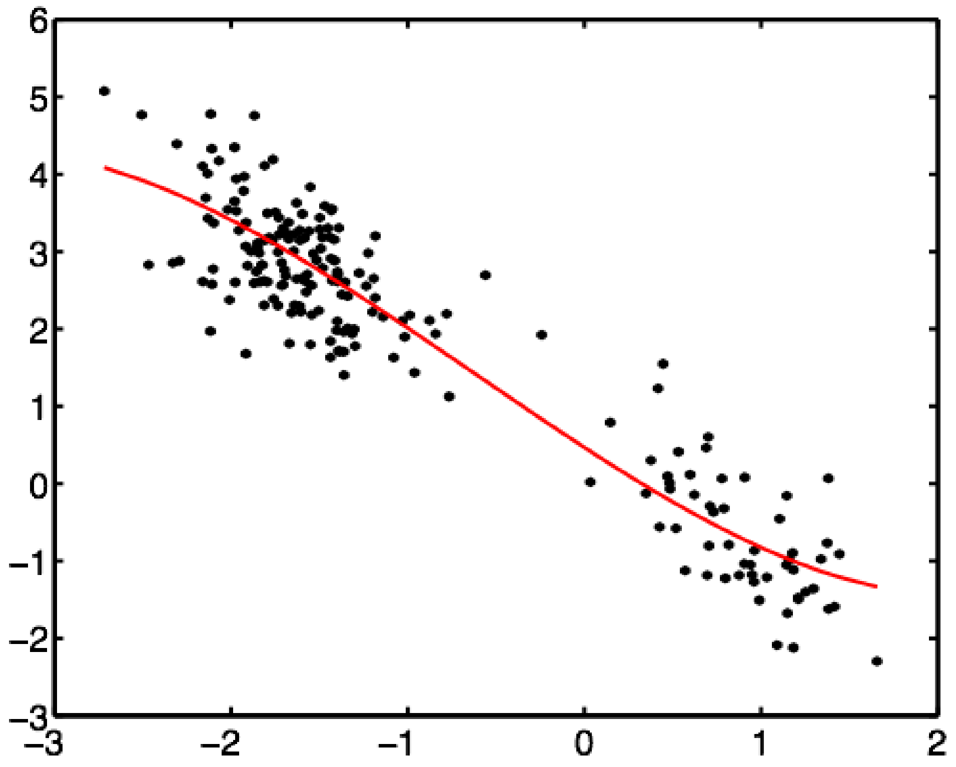
- Here β_1 is a shared constant and can be dropped. No need to learn it or any parameters of mixture 2 at all!

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Covariate shift

• Example outcome on simple data: predict vertical from horizontal. Pluses are points that become associated more to to mixture components of the “training-only” source set 2, dots are points that become associated more to to mixture components of the “both training and test” source set 1





• Original functions are mixtures of two linear regressors each. The learned functions are here cubic regressors, details not on this course; one can just as well use some other regressor.

Following the approach from Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems. Images from that paper.

Part 2: Importance weighted cross-validation for covariate shift

Importance weighted cross-validation for cov. shift

- For learning a supervised predictor, empirical risk minimization is a standard approach:

$$\widehat{\theta}_{ERM} \equiv \underset{\theta\in\Theta}{\operatorname{argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i, \widehat{f}(x_i;\theta)) \right]$$

loss function for input x, output y, and prediction f

- If the ratio of train and test densities is somehow known, we can do importance sampling:

if $\frac{p_{test}(x_i)}{p_{train}(x_i)}$ is known (or can be estimated from unlabeled data) we can weight losses by it to approximate loss on test data:

$$\widehat{\theta}_{IWERM} \equiv \underset{\theta\in\Theta}{\operatorname{argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} \frac{p_{test}(x_i)}{p_{train}(x_i)} \ell(x_i, y_i, \widehat{f}(x_i;\theta)) \right]$$

Following the approach from Sugiyama, M., Krauledat, M., and Muller, K-R. 2007. Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research, (8): 985-1005. Images from that paper.

Importance weighted cross-validation for cov. shift

• One can use an adaptive estimate (where 0<= lambda<=1 controls risk):

$$\widehat{\theta}_{AIWERM} \equiv \underset{\theta \in \Theta}{\operatorname{argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} \left(\frac{p_{test}(x_i)}{p_{train}(x_i)} \right)^{\lambda} \ell(x_i, y_i, \widehat{f}(x_i;\theta)) \right]$$

• Cross-validation can be used to choose lambda, but covariate shift must be taken into account in the procedure:

$$\widehat{R}^{(n)}_{LOOIWCV} \equiv \frac{1}{n} \sum_{j=1}^{n} \frac{p_{test}(x_j)}{p_{train}(x_j)} \ell(x_j, y_j, \widehat{f}_j(x_j))$$

this simply weights validation errors by importances.

• It turns out this has good theoretical properties: gives unbiased estimate of risk even under covariate shift

$$\mathbb{E}_{\{x_i, y_i\}_{i=1}^{n}} \left[\widehat{R}^{(n)}_{LOOIWCV} \right] = R^{(n-1)}$$

Following the approach from Sugiyama, M., Krauledat, M., and Muller, K-R. 2007. Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research, (8): 985-1005. Images from that paper.

References for parts 1-2

- Daume, H., and Marcu, D. 2006. Domain adaptation for statistical classifiers. Journal of Artificial Intelligence Research, (26):101-126.
- S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of Representations for Domain Adaptation. In proceedings of NIPS 2006.
- Storkey, A. J., and Sugiyama, M. 2007. Mixture Regression for Covariate Shift. Advances in Neural Information Processing Systems.
- Sugiyama, M., Krauledat, M., and Muller, K-R. 2007. Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research, (8): 985-1005.

Part 3:
Covariate Shift in Multi-task Settings

Asymmetric multi-task learning

- In **covariate shift** scenarios (lecture 11) the conditional probability of labels is assumed to stay the same between training and test data, but the marginal probability of inputs changes between training and test data.

- A typical solution was to change the weight of training samples **x** during learning, by assigning importance weights proportional to the density ratio $\frac{p_{\text{test}}(\mathbf{x})}{p_{\text{train}}(\mathbf{x})}$ inside the cost function of the task.

- To get the weights, instead of separately estimating the densities, one can directly try to estimate the ratio by various techniques (e.g. based on some labeled data, or by trying to match some statistics like mean, variance etc. between the weighted training distribution and available samples from the test distribution)

- Can we extend this idea to the multi-task learning scenario?

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- In a multi-task setting not only the joint distributions (of samples and labels) may differ between tasks, but also the training and test distribution may differ within each task.
- Usually some tasks have similar joint distributions
- Consider the following scenario: you have several tasks (with inputs x and outputs y) where the task is indexed by z, so that the conditional probability of future test inputs and outputs from task $p_{\text{test}}(\mathbf{x}, y|z) = p_{\text{test}}(\mathbf{x}|z)p(y|\mathbf{x}, z)$
- An unlabeled test sample $T = \langle(\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_m, z_m)\rangle$ is available where inputs are known and which task they come from, but not the labels.
- In each task z the training data comes from a different distribution $p_{\text{train}}(\mathbf{x}, y|z) = p_{\text{train}}(\mathbf{x}|z)p(y|\mathbf{x}, z)$ than its test data: the marginal probability of inputs is different.
- We assume $p_{\text{train}}(\mathbf{x}|z)$>0 wher $p_{\text{test}}(\mathbf{x}|z)$ >0 (all test inputs can occur in training)

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. **Transfer Learning by Distribution Matching for Targeted Advertising.** In *Proc. NIPS 2008.*. Images from that paper.

Asymmetric multi-task learning

• For each task we wish to learn an input-output function minimizing the expected loss (e.g. a squared error, negative log-probability of labels, or some other loss) over the distribution of test data:

$$\mathbf{E}_{(\mathbf{x},y)\sim p_{\text{test}}(\mathbf{x},y|z)}\left[\ell(f_z(\mathbf{x}),y)\right]$$

• In targeted advertising, z might indicate one of several large web sites (e.g. a news or shopping portal), **x** might be a browsing profile of a user, and the task might be to predict the gender y of the user from the browsing profile of the user on the portal.

• For a small number of users of each portal, you might be able to collect the gender from users who take surveys.

• People who answer surveys are not the same as all users ---> the marginal distribution of labeled training data differs rom the distribution of all users (= test data distribution).

• We could use a covariate shift method in each task separately, but it would not make use of relationships between tasks.

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- Define an indicator variable s=1 for training data, s=-1 for test data.
- Consider simply pooling all training data of all tasks: the data in the resulting pool has the distribution $\sum_z p(z|s=1)p(\mathbf{x},y|z,s=1)$
- We will create a task-specific resampling weight $r_t(\mathbf{x},y)$ for each sample in the pool. The resampling weights match the pool distribution to the target distribution
- We can then rewrite

$$\mathbf{E}_{(\mathbf{x},y)\sim p(\mathbf{x},y|t,s=-1)}[\ell(f(\mathbf{x},t),y)]$$

expected loss over test data from task t

$$= \quad \mathbf{E}_{(\mathbf{x},y)\sim\sum_z p(z|s=1)p(\mathbf{x},y|z,s=1)}[r_t(\mathbf{x},y)\ell(f(\mathbf{x},t),y)]$$

expected weighted loss over pooled training data from all tasks

- We need to find weights $r_t(\mathbf{x},y)$ that satisfy the above

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- It can be shown that to satisfy the equation, we get

$$r_t(\mathbf{x}, y) \quad = \quad \frac{p(\mathbf{x}, y|t, s=1)}{\sum_z p(z|s=1)p(\mathbf{x}, y|z, s=1)} \frac{p(\mathbf{x}|t, s=-1)}{p(\mathbf{x}|t, s=1)}$$

- To show this,

$$\mathbf{E}_{(\mathbf{x},y)\sim p(\mathbf{x},y|t,s=-1)}[\ell(f(\mathbf{x},t),y)]$$

$$= \int \frac{\sum_z p(z|s=1)p(\mathbf{x}, y|z, s=1)}{\sum_{z'} p(z'|s=1)p(\mathbf{x}, y|z', s=1)} \frac{p(\mathbf{x}|t, s=1)}{p(\mathbf{x}|t, s=1)} p(\mathbf{x}, y|t, s=-1)\ell(f(\mathbf{x}, t), y) d\mathbf{x}dy$$

$$= \int \sum_z \left(\frac{p(z|s=1)p(\mathbf{x}, y|z, s=1)}{\sum_{z'} p(z'|s=1)p(\mathbf{x}, y|z', s=1)} \frac{p(\mathbf{x}|t, s=1)}{p(\mathbf{x}|t, s=1)} p(\mathbf{x}|t, s=-1)p(y|\mathbf{x}, t)\ell(f(\mathbf{x}, t), y) \right) d\mathbf{x}dy$$

$$= \int \sum_z \left(p(z|s=1)p(\mathbf{x}, y|z, s=1) \frac{p(\mathbf{x}|t, s=1)p(y|\mathbf{x}, t)}{\sum_{z'} p(z'|s=1)p(\mathbf{x}, y|z', s=1)} \frac{p(\mathbf{x}|t, s=-1)}{p(\mathbf{x}|t, s=1)} \right.$$

$$\left. \ell(f(\mathbf{x}, t), y) \right) d\mathbf{x}dy$$

$$= \mathbf{E}_{(\mathbf{x},y)\sim \sum_z p(z|s=1)p(\mathbf{x},y|z,s=1)} \left[\frac{p(\mathbf{x}, y|t, s=1)}{\sum_{z'} p(z'|s=1)p(\mathbf{x}, y|z', s=1)} \frac{p(\mathbf{x}|t, s=-1)}{p(\mathbf{x}|t, s=1)} \ell(f(\mathbf{x}, t), y) \right]$$

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- The equation

$$r_t(\mathbf{x}, y) = \frac{p(\mathbf{x}, y|t, s=1)}{\sum_z p(z|s=1)p(\mathbf{x}, y|z, s=1)} \frac{p(\mathbf{x}|t, s=-1)}{p(\mathbf{x}|t, s=1)}$$

has two factors: (left) proportion of joint probability in the correct task t compared to the overall pool, (right) ratio of marginal probability in test samples of task t versus training samples from the task.

- The remaining question is how to estimate the above weights: estimating all densities involved in the equation is overly difficult since all we need is the final weight.
- We will transform the estimation into two classification problems!

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- The equation can be rewritten as

$$r_t(\mathbf{x},y) = r_t^1(\mathbf{x},y)\,r_t^2(\mathbf{x})$$

where $r_t^1(\mathbf{x},y) = \frac{p(\mathbf{x},y|t,s=1)}{\sum_z p(z|s=1)p(\mathbf{x},y|z,s=1)} = p(t|\mathbf{x},y,s=1)$

is the probability that a labeled training sample (x,y,s=1) comes from a particular task t, and

the ratio $r_t^2(\mathbf{x}) = \frac{p(\mathbf{x}|t,s=-1)}{p(\mathbf{x}|t,s=1)}$ is proportional to another probability:

$$\begin{aligned} r_t^2(\mathbf{x}) &= \frac{p(\mathbf{x}|t,s=-1)}{p(\mathbf{x}|t,s=1)} = \frac{p(s=-1|\mathbf{x},t)p(\mathbf{x}|t)}{p(s=-1|t)} \frac{p(s=1|t)}{p(s=1|\mathbf{x},t)p(\mathbf{x}|t)} \\ &\propto \frac{1}{p(s=1|\mathbf{x},t)} - 1 \end{aligned}$$

probability that an input x from task t is a training sample

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- The two probabilities $p(t|\mathbf{x}, y, s=1)$ and $p(s=1|\mathbf{x}, t)$ can each be **estimated** from the set of labeled training samples of all tasks and unlabeled test samples of all tasks.

- The probabilities are probabilities of labels, so estimating them can be done as **probabilistic classification.**

- If there are two tasks, then both probabilities are probabilities of binary labels -----> we can use for example logistic regression to estimate the probabilities.

- For example we can set

$$p(t|\mathbf{x}, y, s=1, \mathbf{u}_t) = \frac{1}{1+\exp(-\mathbf{u}_t^\top \Phi(\mathbf{x}, y))}$$

where $\Phi(\mathbf{x}, y)$ is a mapping from (**x**,y) to features, $\mathbf{u}_t$ is a parameter.

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

Asymmetric multi-task learning

- For binary labels for example

$$\Phi(\mathbf{x}, y) = \begin{bmatrix} \delta(y, +1)\Phi(\mathbf{x}) \\ \delta(y, -1)\Phi(\mathbf{x}) \end{bmatrix}$$

- The rest is standard logistic regression!
- Once it is done, then the rest is just weighting samples in standard learning as in the previous covariate shift.

Following the approach from S. Bickel, C. Sawade, and T. Scheffer. Transfer Learning by Distribution Matching for Targeted Advertising. In *Proc. NIPS 2008*. Images from that paper.

References for part 3

- Steffen Bickel, Christoph Sawade, and Tobias Scheffer. **Transfer Learning by Distribution Matching for Targeted Advertising.** In *Proceedings of NIPS 2008, International Conference on Neural Information Processing Systems*, 2008.