

<http://goo.gl/Jv7Vj9>

Course website

KYOTO UNIVERSITY

*Statistical Machine Learning Theory*

**Semi-supervised, Active, and Transfer Learning**

Hisashi Kashima  
kashima@i.Kyoto-u.ac.jp

DEPARTMENT OF INTELLIGENCE SCIENCE  
AND TECHNOLOGY

# Topics:

## Semi-supervised, active, and transfer learning

---

- Semi-supervised learning
  - Weighted maximum likelihood estimation
  - Graph-based methods (e.g. label propagation)
  - Self-training
- Active learning
  - Uncertainty sampling
  - Estimated model change
- Transfer learning
  - Covariate shift using with weighted ML estimation
  - Shared parameters and domain specific parameters

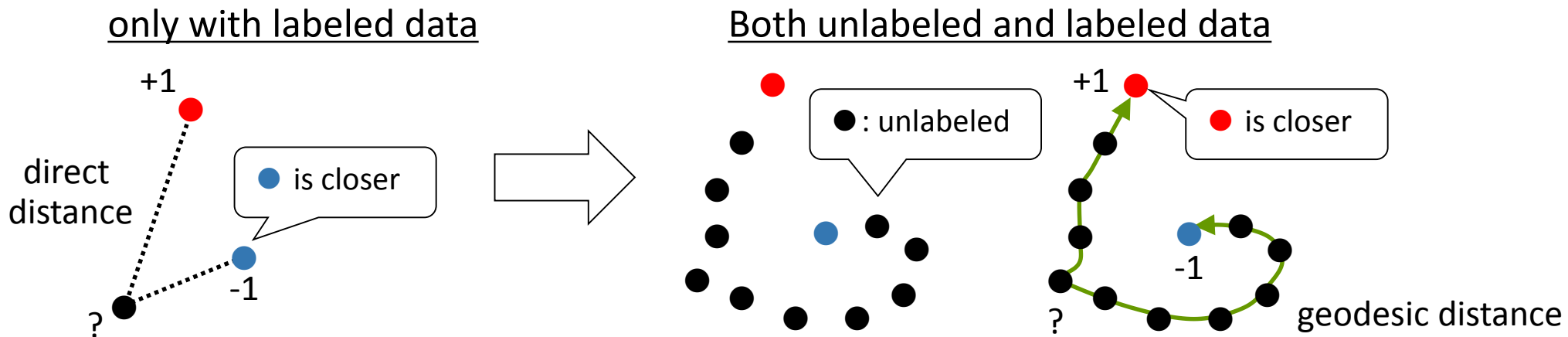
# Semi-supervised learning and active learning: Learning with labeled and unlabeled data

---

- We have both labeled and unlabeled instances
  - Labeled data:  $\{ (\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)}) \}$
  - Unlabeled data:  $\{ \mathbf{x}^{(N+1)}, \dots, \mathbf{x}^{(N+M)} \}$
  - Usually,  $N \ll M$
- Semi-supervised learning uses unlabeled data as well as labeled data
- Active learning
  - has access to an oracle to give labels to unlabeled data
  - has to choose which unlabeled data to query next

# Role of unlabeled data in supervised learning: Information of the input data distribution

- Data generation process
  - Input  $\mathbf{x}$  is generated by input data distribution  $\mathcal{D}_x$
  - Output  $y$  for  $\mathbf{x}$  is generated by conditional distribution  $\mathcal{D}_{y|\mathbf{x}}$
- Unlabeled data can be used for capturing  $\mathcal{D}_x$ 
  - Input data distribution, input space metric, or better representations



# Semi-supervised Learning

# Semi-supervised learning problem:

## Learning with labeled and unlabeled data

---

- We have both labeled and unlabeled instances
  - Labeled data  $L = \{ (\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)}) \}$
  - *Unlabeled data*  $U = \{ (\mathbf{x}^{(N+1)}, y^{(N+1)}), \dots, (\mathbf{x}^{(N+M)}, y^{(N+M)}) \}$
- Estimate a *deterministic mapping*  $f: \mathcal{X} \rightarrow \mathcal{Y}$  (often with a confidence value) or a *conditional probability*  $P(y|\mathbf{x})$

# Typical approaches of semi-supervised learning: Learning with labeled and unlabeled data

---

- Weighted maximum likelihood estimation
- Graph-based learning
- Self-training
- Clustering
- Generative models

## Weighted maximum likelihood:

### Estimate input distribution to weight labeled instances

- The original goal of ML estimation is to maximize

$$E_{\mathbf{x}}[\log P(y|\mathbf{x})] = \int \log p(y|\mathbf{x}) dp(\mathbf{x}) \approx \frac{1}{N} \sum_{i=1}^N \log p(y^{(i)}|\mathbf{x}^{(i)})$$

– Each training data instance is equally weighted

- Weighted maximum likelihood:

Each training data instance is weighted according to  $p(\mathbf{x})$

$$\text{maximize } \sum_{i=1}^N p(\mathbf{x}^{(i)}) \log p(y^{(i)}|\mathbf{x}^{(i)})$$

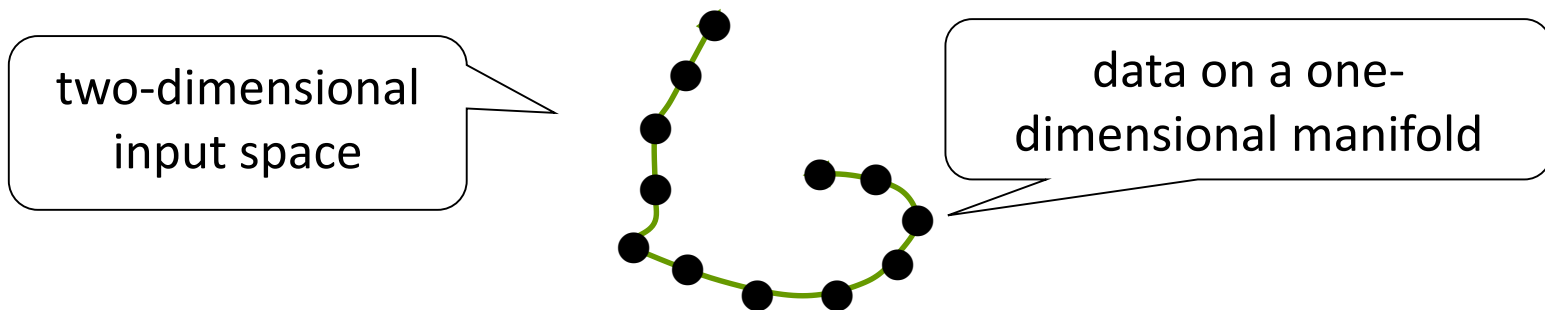
–  $p(\mathbf{x})$  is estimated using unlabeled data



# Graph-based method:

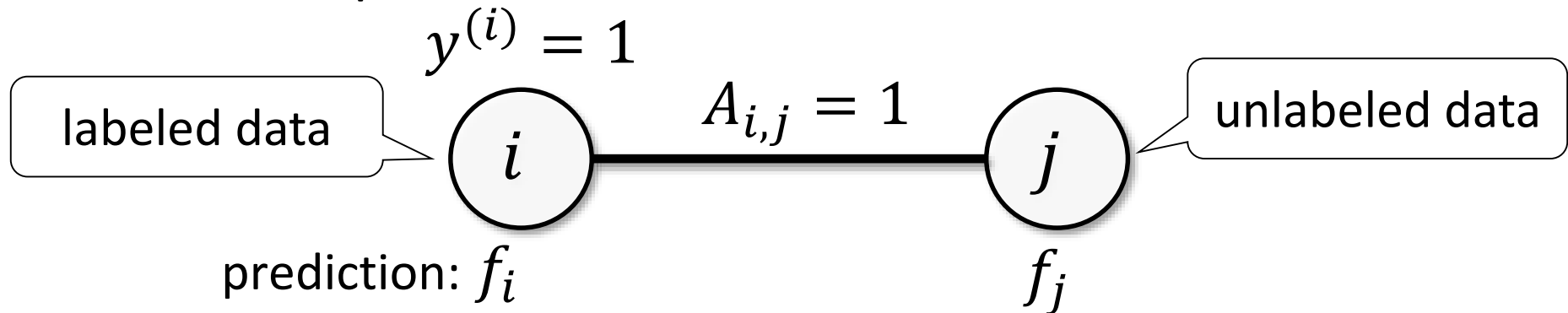
## Capture intrinsic shape of input space

- Basic idea: construct a graph capturing the intrinsic shape of the input space, and make predictions on the graph
- Assumption: Data lie on a manifold in the feature space
- The graph represent adjacency relationships among data
  - $K$ -nearest neighbor graph (e.g.  $A_{i,j} = \{0, 1\}$ )
  - Edge-weighted graph with e.g.  $A_{i,j} = \exp(-\| \mathbf{x}^{(i)} - \mathbf{x}^{(j)} \|_2^2)$



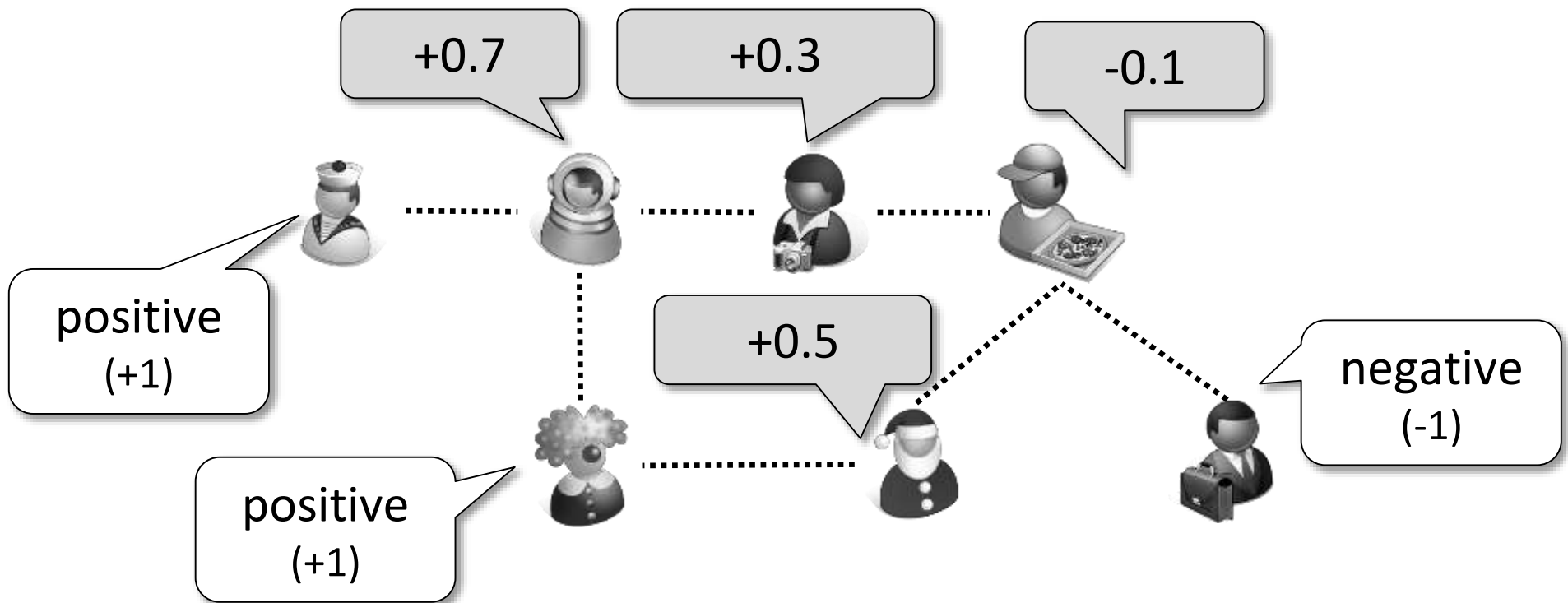
# Label propagation: Simple graph-based method

- Basic idea: Adjacent instances tend to have the same label
  - Note that we have test instances (i.e. transductive setting)
- minimize $_f \sum_{i=1}^N (f_i - y^{(i)})^2 + \gamma \sum_{i,j} A_{i,j} (f_i - f_j)^2$ 
  - 1st term: (squared) loss function to fit to labeled data
  - 2nd term: regularization function to make adjacent nodes to have similar predictions



# Illustrative example of label propagation: Infection prediction on social network

- Predict if people are infected by some disease
  - Test results are known for some people
  - Infections spread over social networks



# Self-training: Believe what you believe

---

- Procedure:

1. Initialization: train a classifier using labeled dataset  $L$
  2. Use the classifier to assign temporary labels to unlabeled dataset  $U$
  3. Train a classifier using  $L$  and  $U$  (with the temporary labels)
  4. Return to Step 2
- For probabilistic classifier, use the weighted ML estimation

# Active Learning

# Active learning:

## Learning with a label oracle

---

- Start with only unlabeled data  $U = \{ \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)} \}$
- At each round, an active learner can query an unlabeled instance to be labeled by an oracle
  - then update the predictor using current labeled (and unlabeled) data
- An active learning algorithm determines the query strategy specifying which unlabeled instance should be queried next

# Active learning query strategies:

## Choose the most “informative” instance

---

- Basic idea: Query the instance whose label is the most informative
- Several basic strategies to choose “informative” instance
  - Query the instance with the most uncertain label
  - Query the instance which will give the largest expected model change
  - ...

## Uncertainty sampling:

### Query the instance with the most uncertain label

---

- In a linear classifier  $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$ ,  $\mathbf{w}^\top \mathbf{x}$  indicates “confidence level” of the prediction
  - For multi-class classification,
    - use  $\max_k \mathbf{w}^{(k)\top} \mathbf{x}$
    - or, margin  $\max_k \mathbf{w}^{(k)\top} \mathbf{x} - \text{secondbest}_k \mathbf{w}^{(k)\top} \mathbf{x}$
  - For probabilistic classifiers, the entropy  $\sum_y -P(y|\mathbf{x}) \log P(y|\mathbf{x})$  is used as an uncertainty measure
- Query  $\mathbf{x}^{(i)}$  with the lowest confidence/highest uncertainty



# Differences among confidence level, margin, and entropy

[Settles, 2010. page 14]

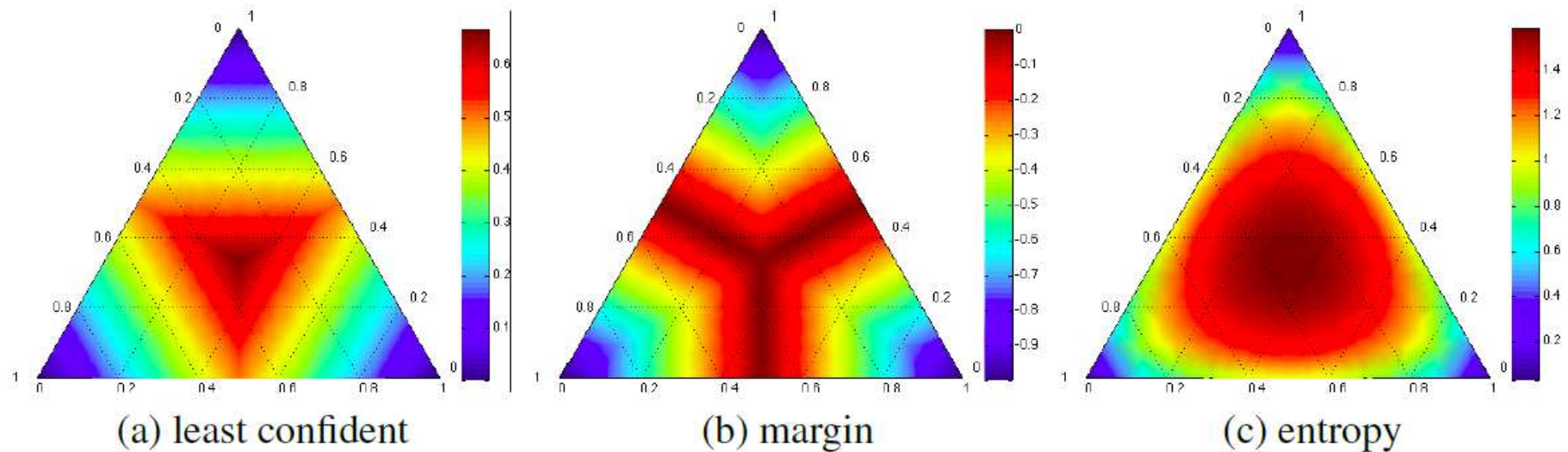


Figure 5: Heatmaps illustrating the query behavior of common uncertainty measures in a three-label classification problem. Simplex corners indicate where one label has very high probability, with the opposite edge showing the probability range for the *other* two classes when that label has very low probability. Simplex centers represent a uniform posterior distribution. The most informative query region for each strategy is shown in dark red, radiating from the centers.

## limitation of uncertainty sampling :

### Uncertainty sampling is based on local information

---

- Querying the least confident instance cares only about the local information
- Obtaining one labeled instance can make an impact on the whole model
- We should take the amount of the “impact” of a label into account

## Expected model change:

### Query the instance which gives the largest model change

- Choose instance  $\mathbf{x}$  which gives the largest (expected) gradient of the objective function:  $\sum_y -P(y|\mathbf{x}) \|\nabla_{\mathbf{w}} J(L \cup (\mathbf{x}, y))\|$

– Assume gradient-based learning methods are used

- e.g. gradient descent  $\mathbf{w}^{\text{new}} \leftarrow \mathbf{w}^{\text{old}} - \gamma \nabla_{\mathbf{w}} J(L \cup (\mathbf{x}, y))$  when new labeled instance  $(\mathbf{x}, y)$  is added to  $L$

- Choose an instance which gives the largest information gain

$$\sum_y -P(y|\mathbf{x}) \sum_{i=N+1}^{N+M} \sum_{y'} P_{\mathbf{w}^{\text{new}}}(y'|\mathbf{x}^{(i)}) \log P_{\mathbf{w}^{\text{new}}}(y'|\mathbf{x}^{(i)})$$

–  $P_{\mathbf{w}^{\text{new}}}$ : model after update with new labeled data  $(\mathbf{x}, y)$

# Transfer Learning

# Transfer learning:

## Training and test data come from different distributions

- In transfer learning, the training dataset and the test dataset are sampled from different distributions
- In transfer learning,
  - Training data come from  $\mathcal{D}_x^{\text{train}}$  and  $\mathcal{D}_{y|x}^{\text{train}}$
  - Test data come from  $\mathcal{D}_x^{\text{test}}$  and  $\mathcal{D}_{y|x}^{\text{test}}$
  - Previously we assumed an input  $x$  sampled from  $\mathcal{D}_x$ , and an output  $y$  from  $\mathcal{D}_{y|x}$
- Example: Domain adaptation
  - Text classification of general text documents and medical texts

## Covariate shift:

### Training and test input distributions are different

---

- Covariate shift: only the input distributions are different
  - $\mathcal{D}_x^{\text{train}} \neq \mathcal{D}_x^{\text{test}}$
  - $\mathcal{D}_{y|x}^{\text{train}} = \mathcal{D}_{y|x}^{\text{test}}$ : conditional distributions are the same
  - Labeled training dataset and unlabeled test dataset are given
- Often occurs when label sampling is limited for some reason
  - Labels are obtained only from the targets to which some actions are taken (e.g. responses to direct mails)
  - Labels can only be taken in controlled environments (eg. in-vitro experiments)
  - Active learning controls the training distribution

# Maximum likelihood learning under covariate shift :

## Maximize likelihood for test input distribution

---

- The distribution on which we want to work well is the test input distribution  $p^{\text{test}}(\mathbf{x})$
- In maximum likelihood estimation, we want to maximize

$$E_{\mathbf{X}}^{\text{test}}[\log P(y|\mathbf{x})] = \int p^{\text{test}}(\mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}$$

- Note that the expectation is taken over  $p^{\text{test}}(\mathbf{x})$
- However we do not have label information for test dataset
  - We can not evaluate the objective function directly

# Covariate shift learning only with training labels: Weighted maximum likelihood with density ratio

- Use of the importance sampling

$$E_{\mathbf{X}}^{\text{test}}[\log P(y|\mathbf{x})] = \int \frac{p^{\text{test}}(\mathbf{x})}{p^{\text{train}}(\mathbf{x})} p^{\text{train}}(\mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}$$

$$\approx \frac{1}{N} \sum_{i=1}^N \frac{p^{\text{test}}(\mathbf{x}^{(i)})}{p^{\text{train}}(\mathbf{x}^{(i)})} \log p(y^{(i)}|\mathbf{x}^{(i)})$$

$$= \frac{1}{N} \sum_{i=1}^N \omega(\mathbf{x}^{(i)}) \log p(y^{(i)}|\mathbf{x}^{(i)})$$

training data  $(\mathbf{x}^{(i)}, y^{(i)})$  is  
weighted with  $\omega(\mathbf{x}^{(i)})$

–Weighted ML estimation with weight  $\omega(\mathbf{x}^{(i)}) = \frac{p^{\text{test}}(\mathbf{x}^{(i)})}{p^{\text{train}}(\mathbf{x}^{(i)})}$



## Practical considerations:

### Density ratio estimation and adaptive importance

---

- Estimation of the density ratio  $\omega(\mathbf{x}) = \frac{p^{\text{test}}(\mathbf{x})}{p^{\text{train}}(\mathbf{x})}$  is required
  - Density estimation of  $p^{\text{test}}$  and  $p^{\text{train}}$
  - Some approaches directly estimate  $\omega$
- Adaptive importance weighted ML estimation:
  - Practically  $\omega^\lambda(\mathbf{x}^{(i)}) = \left( \frac{p^{\text{test}}(\mathbf{x}^{(i)})}{p^{\text{train}}(\mathbf{x}^{(i)})} \right)^\lambda$  ( $0 \leq \lambda \leq 1$ ) works better

# Transfer learning of different conditional distributions: Adaptation to model changes

---

- Transfer learning of different conditional distributions
  - $\mathcal{D}_{y|x}^{\text{train}} \neq \mathcal{D}_{y|x}^{\text{test}}$
  - $\mathcal{D}_x^{\text{train}} = \mathcal{D}_x^{\text{test}}$ : Input distributions are the same
  - Labels are available in both training and test datasets
- Adaptation to changes of predictive models
  - Transfer knowledge from a general task to a specific task (and vice versa)
  - Model changes over time

# A simple approach to model change adaptation:

## Shared parameters and domain specific parameters

---

- Assume linear models (e.g.  $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$ )
  - The source domain model has  $\mathbf{w}^{(s)}$ , while the target domain model has  $\mathbf{w}^{(t)}$
- The models have shared parts and domain specific parts
  - Source domain model  $\mathbf{w}^{(s)} = \mathbf{v}^{(0)} + \mathbf{v}^{(s)}$
  - Target domain model  $\mathbf{w}^{(t)} = \mathbf{v}^{(0)} + \mathbf{v}^{(t)}$
  - Equivalent to setting  $\mathbf{w} = (\mathbf{v}^{(0)}, \mathbf{v}^{(s)}, \mathbf{v}^{(t)})$  and  $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{x}, \mathbf{0})$  for the source domain and  $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{0}, \mathbf{x})$  for the target domain
- A standard classification method is applicable