Introduction Transfer Learning

A. de Mathelin, M. Atiq

Michelin - Centre Borelli, ENS Paris-Saclay

December 2022

Outline

Motivation

Example 1 : Object recognition

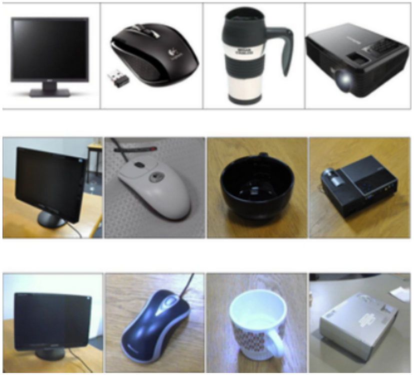
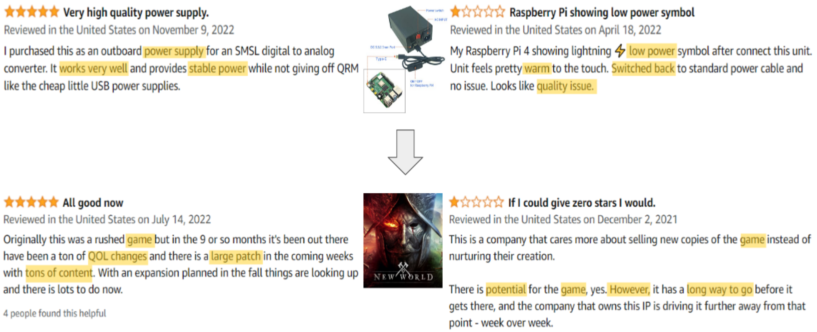


Figure – Transfer object recognition from one camera environment to a different one

Motivation

Example 2 : Sentiment analysis



★★★★★ **Very high quality power supply.**

Reviewed in the United States on November 9, 2022

I purchased this as an outboard power supply for an SMSL digital to analog converter. It works very well and provides stable power while not giving off QRM like the cheap little USB power supplies.

★☆☆☆☆ **Raspberry Pi showing low power symbol**

Reviewed in the United States on April 18, 2022

My Raspberry Pi 4 showing lightning ⚡ low power symbol after connect this unit. Unit feels pretty warm to the touch. Switched back to standard power cable and no issue. Looks like quality issue.

★★★★★ **All good now**

Reviewed in the United States on July 14, 2022

Originally this was a rushed game but in the 9 or so months it's been out there have been a ton of QOL changes and there is a large patch in the coming weeks with tons of content. With an expansion planned in the fall things are looking up and there is lots to do now.

4 people found this helpful

★☆☆☆☆ **If I could give zero stars I would.**

Reviewed in the United States on December 2, 2021

This is a company that cares more about selling new copies of the game instead of nurturing their creation.

There is potential for the game, yes. However, it has a long way to go before it gets there, and the company that owns this IP is driving it further away from that point - week over week.

Figure – Transfer customer feedback reviews from one product to another

Motivation

Example 3 : Fall detection

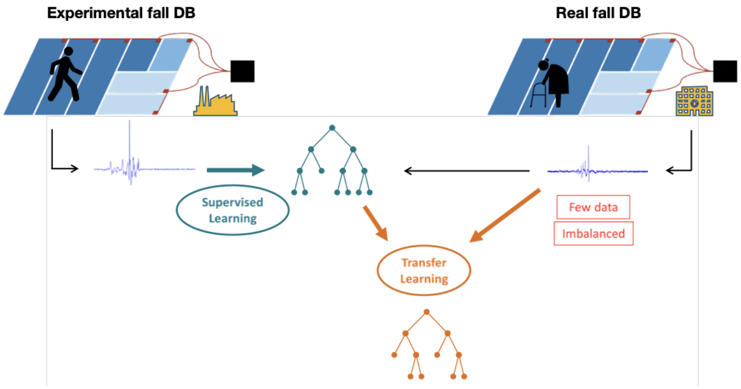


Figure – Transfer industrial detection task from a simulated environment to a real one

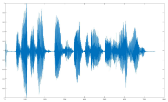
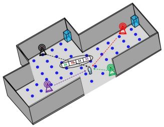
Applications

Various kinds of data :

- Time series
- Image and videos
- Audio and speech
- Natural language processing

Various fields of application :

- Anomaly detection
- Biology
- Medicine
- Video games
- Indoor localization
- Computer vision

Transfer learning definition

Transfer learning implies :

- Starting from a classical ML problem with $\mathcal{X}$ a feature space and $\mathcal{Y}$ a label space, (X, Y) random variables over these spaces.
- Directional aspect : to move from a **Source** problem to a different but related **Target** problem
- Two different distributions : $(X_S, Y_S) \sim P_S$ and $(X_T, Y_T) \sim P_T$

Transfer learning approaches usually rely on :

- Both **Source** and **Target** data
- Some **similarity measures** between distributions
- Some assumptions about relations between **Source** and **Target** distributions

Transfer learning definition

Domain/Task

$\mathcal{X}$ a feature space and $\mathcal{Y}$ a label space, a couple of random variables $(X, Y) \sim P$.

- **Domain** : $\mathcal{D} = \{X \in \mathcal{X}, P(X)\}$
- **Task** : $\mathcal{T} = \{Y \in \mathcal{Y}, P(Y|X = x)\}$

Transfer learning

Exploit knowledge acquired from previous learning on **source** domains/tasks ($\mathcal{D}_S$,$\mathcal{T}_S$) to benefit to new learning on *different* but *related* **target** domains/tasks ($\mathcal{D}_T$,$\mathcal{T}_T$).

- Domain shift : $\mathcal{X}_S \neq \mathcal{X}_T$ or $P_S(X_S) \neq P_T(X_T)$
- Task shift : $\mathcal{Y}_S \neq \mathcal{Y}_T$ or $P_S(Y_S|X_S = x) \neq P_T(Y_T|X_T = x)$

Transfer learning definition

How is it different from classic ML ?

Risk of a classifier

Risk $R(h)$ of a *classifier* $h : \mathcal{X} \longrightarrow \mathcal{Y}$, according to a *loss function* ℓ, under a *probability distribution* P :

$$R(h) = \mathop{\mathbb{E}}_{(X,Y)\sim P}[\ell(h(X), Y)]$$

Empirical risk minimization

With $\{(x_i, y_i)\}_{i=1}^{N} \in \mathcal{X} \times \mathcal{Y}$ being N i.i.d samples drawn according to P distribution,

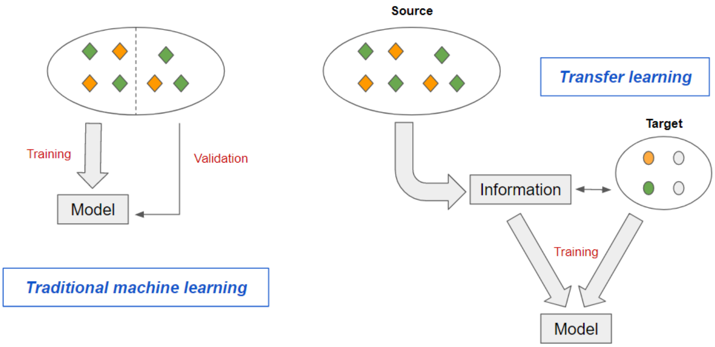
Objective : Minimize $\hat{R}(h) = \frac{1}{N}\sum_{i=1}^{N} \ell(h(x_i), y_i)$

In transfer learning, risks on **Source** and **Target** can differ :

$$P_S \neq P_T \longrightarrow R_S(h) \neq R_T(h)$$

Transfer learning definition

How is it different from classic ML ?



Why is re-training only on target domain/task not sufficient ?

- Few target data
- Unlabeled target data
- Partially representative data
- Missing features

Taxonomy of transfer learning (problem oriented)

Heterogeneous transfer

While feature space or label space differs from **source** to **target** : $\mathcal{X}_S \neq \mathcal{X}_T$ or $\mathcal{Y}_S \neq \mathcal{Y}_T$.

Example 1 : Transfer an object recognition task from depth-camera images to normal images ($\mathcal{X}_S \neq \mathcal{X}_T$).
Example 2 : Transfer from music instrument detection task to music style recognition task ($\mathcal{Y}_S \neq \mathcal{Y}_T$).

Different categories of transfer depending on labels availability :

- **Inductive** transfer : Labels are available both in **source** and **target**.
- **Transductive** transfer : Labels are available in **source** but not in **target**.
- Fully **unsupervised** transfer : No labels are available both on **source** and **target**.

Taxonomy of transfer learning (solution oriented)

Different categories of transfer learning depending on their type of approach :

- **Instance-based** : Re-weight **source** and **target** data instances to correct distributions shift and use both for training.
- **Feature-based** : Find a common feature representation that compensates differences between **source** and **target** domains.
- **Parameter-based** : Modify parameters of a model pre-trained on **source** to get a new model suited for **target** domain/task.

Taxonomy of transfer learning

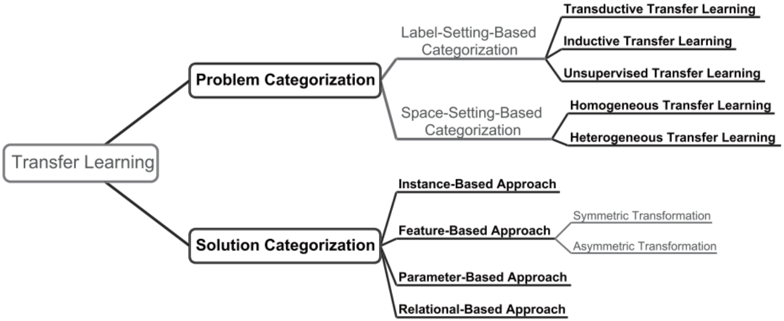

Transfer Learning
Problem Categorization
Label-Setting-Based Categorization
Transductive Transfer Learning
Inductive Transfer Learning
Unsupervised Transfer Learning
Space-Setting-Based Categorization
Homogeneous Transfer Learning
Heterogeneous Transfer Learning
Solution Categorization
Instance-Based Approach
Feature-Based Approach
Symmetric Transformation
Asymmetric Transformation
Parameter-Based Approach
Relational-Based Approach

Figure – Taxonomy of transfer learning situations and methods

"A Comprehensive Survey on Transfer Learning" [F. Zhuang et al., IEE 2021]

Transfer learning borders/limits

Negative transfer

Negative transfer occurs while source knowledge deteriorates performance of learning target task using target domain only. It can happen for several reasons :

- Dissimilarity between source and target domains is too important
- Source and target tasks are not enough related
- Transfer learning strategy is not well-suited for the problem

Related fields

- Multi-source learning
- Multi-task learning
- Reinforcement learning
- Optimal transport

Common assumptions on source/target relation

Covariate shift

Marginal distribution changes between source and target but predictive dependency stays the same.

- $\mathcal{D}_T \neq \mathcal{D}_S, \quad (P_T(X) \neq P_S(X))$
- $\mathcal{T}_T = \mathcal{T}_S, \quad (P_T(Y|X) = P_S(Y|X))$

Target shift

Conditional distribution for a given label y stays the same but the overall proportion of each label changes between source and target. $\forall y \in \mathcal{Y}$,

- $P_T(Y = y) \neq P_S(Y = y)$
- $P_T(X|Y = y) = P_S(X|Y = y)$

Common assumptions on source/target relation

Sample-selection bias

Same underlying distribution P both on source and target. But hidden selection variable ξ that tends to exclude some observations from source data.

- $P_S(X, Y) = P(\xi = 1|X, Y)P(Y|X)P(X)$
- $P_T(X, Y) = P(X, Y) = P(Y|X)P(X)$

$\implies P_S(X, Y) = P(\xi = 1|X, Y)P_T(X, Y)$

Concept shift

Both conditional distribution change between source and target.

- $P_T(Y|X) \neq P_S(Y|X)$
- $P_T(X|Y) \neq P_S(X|Y)$

Outline

Unsupervised Domain Adaptation

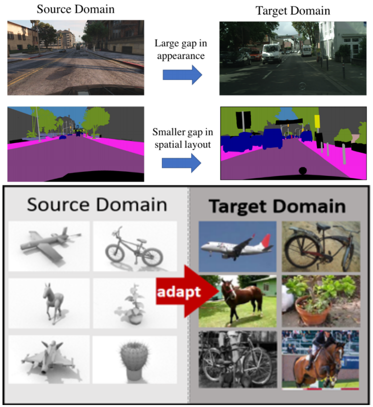


Figure – **Adaptation of Synthetic data** : GTA to CityScape segmentation (top) ; generated 3D objects to real pictures classification (bottom)

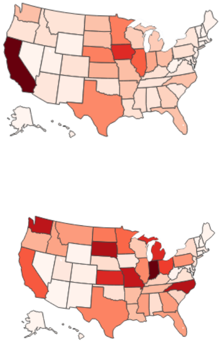


Figure – **Sample Bias** : The distribution of the surveyed population differs than the targeted population

UDA : Definition

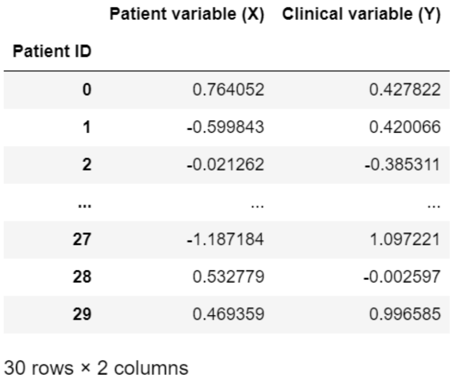
Definition 1 (Unsupervised Domain Adaptation (UDA))

Let's consider a feature space $\mathcal{X}$ and a label space $\mathcal{Y}$. We define $P_S(X, Y)$ and $P_T(X, Y)$ the respective source and target joint distributions over $\mathcal{X} \times \mathcal{Y}$.

We call **Unsupervised Domain Adaption** the learning setting composed of :

- A source **labeled** sample $\mathcal{S} = \{(x_1, y_1), ..., (x_m, y_m)\} \in \mathcal{X} \times \mathcal{Y}$ where $x_i \sim P_S(X)$ and $y_i \sim P_S(Y|X = x_i)$
- A target **unlabeled** sample $\mathcal{T}_{\mathcal{X}} = \{x'_1, ..., x'_m\} \in \mathcal{X}$ where $x'_j \sim P_T(X)$

UDA : Example

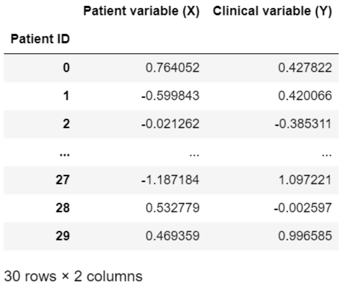


	Patient variable (X)	Clinical variable (Y)
Patient ID		
0	0.764052	0.427822
1	-0.599843	0.420066
2	-0.021262	-0.385311
...	...	...
27	-1.187184	1.097221
28	0.532779	-0.002597
29	0.469359	0.996585

30 rows × 2 columns

Figure – We consider a simple 1D regression task, where the learner is interested in the correlation between a specific patient variable X (eg : age, weight...) and a clinical variable Y (eg : disease evolution...). The learner has collected 30 samples and want to fit a linear model $Y = aX + b$.

UDA : Example



Patient ID	Patient variable (X)	Clinical variable (Y)
0	0.764052	0.427822
1	-0.599843	0.420066
2	-0.021262	-0.385311
...	...	...
27	-1.187184	1.097221
28	0.532779	-0.002597
29	0.469359	0.996585

30 rows × 2 columns

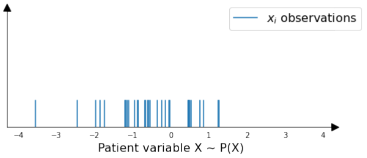


Figure – The observations of the patient variable can be drawn on the x-axis

UDA : Example

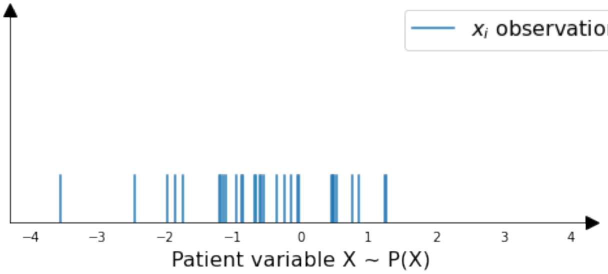


Figure – In practice, the learner assume that the collected sample is independently and identically distributed from an unknown source input distribution $P_S(X)$.

UDA : Example

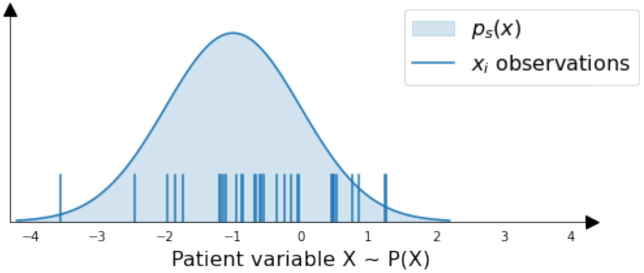


Figure – In this specific case, the sample is drawn according to a Gaussian distribution centered in -1

UDA : Example

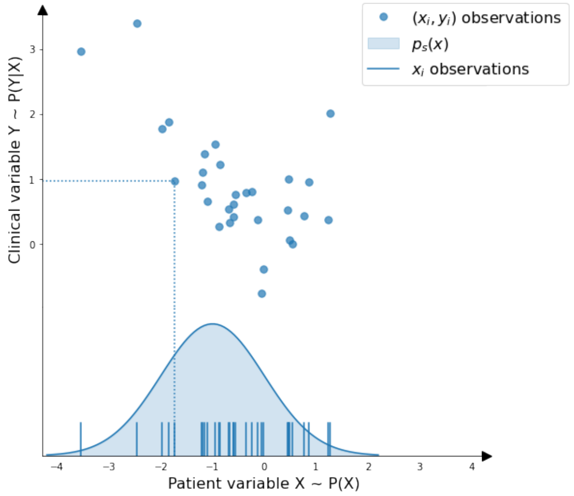


Figure – The learner has also access, in the source sample, to observations of the clinical variable Y. These observations can be drawn on the y-axis.

UDA : Example

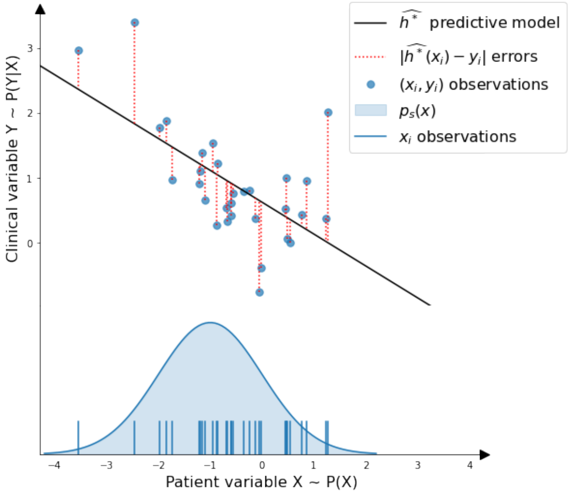


Figure – Based on the (x_i, y_i) observations, the learner can fit a linear model to estimate the correlation between the two variables.

UDA : Example

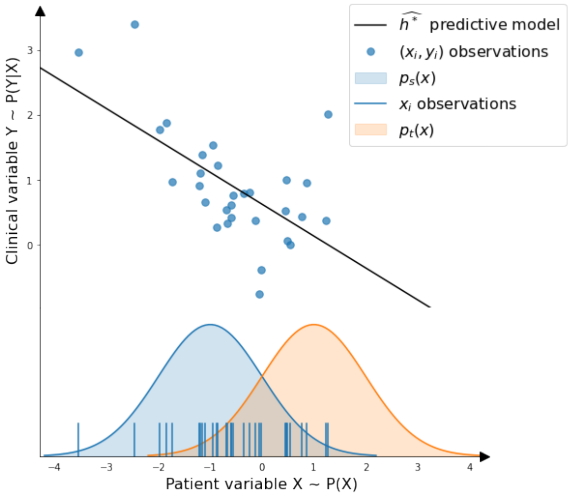


Figure – In the transfer learning scenario, the source collected sample is not representative of the targeted population (here in orange)

UDA : Example

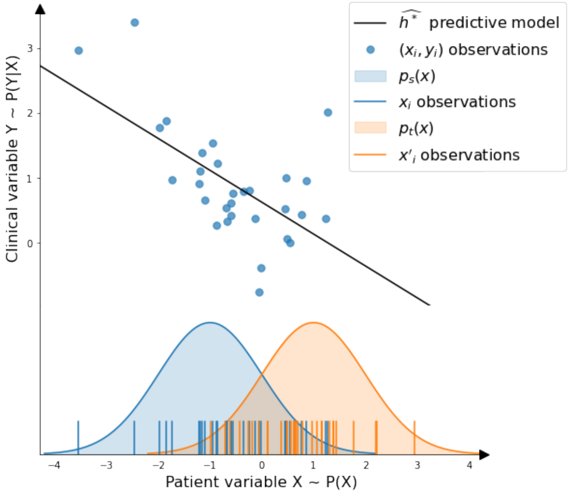


Figure – The learner has access to observations of the patient variable in the target domain.

UDA : Example

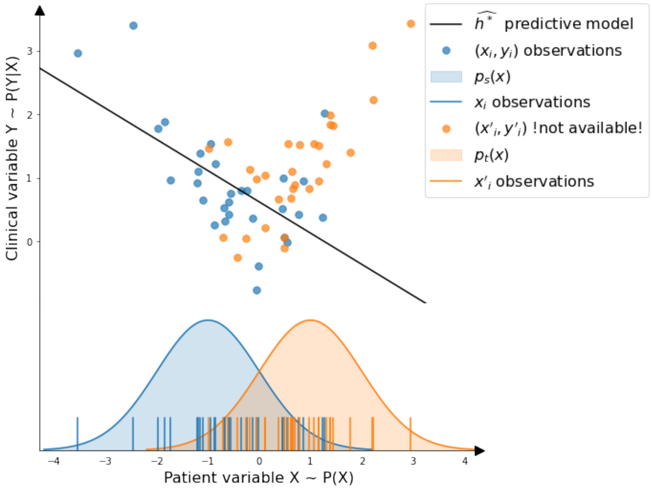


Figure – However, at training time, the learner has not access to target observations of the clinical variable. In this case, we plot them to see how the domain shift can mislead the learning of the model.

UDA : Example

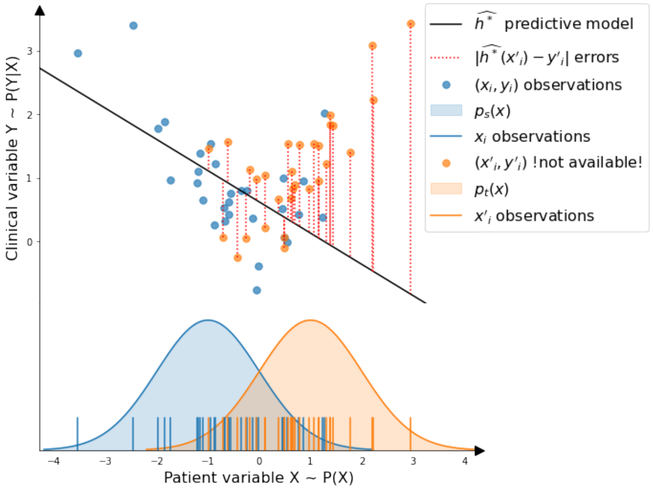


Figure – We can see that our model has been biased by the non representative source data and thus produce large errors on the target domain.

UDA : Problematic

Let's introduce the following notations :

- **loss function** : $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.
  Example : $\ell(y, y') = ||y - y||_2^2$
- **Hypothesis space** : $\mathcal{H}$, a functional space such that $h : \mathcal{X} \to \mathcal{Y}$ for any $h \in \mathcal{H}$.
  Example : $\mathcal{H} = \{h : x \to x\beta^T, \beta \in \mathbb{R}^p\}$ (linear hypotheses)

For any $h \in \mathcal{H}$, the target risk is defined as follows :

$$R_T(h) = \mathop{\mathbb{E}}_{(x,y)\sim P_T(X,Y)} [\ell(h(x), y)] \tag{1}$$

If target labels $y'_j \in \mathcal{Y}$ were available, a straightforward estimation of $R_T(h)$ would be :

$$\widehat{R_T}(h) = \frac{1}{n} \sum_{j=1}^{n} \ell(h(x'_j, y'_j)) \tag{2}$$

UDA : Problematic

UDA Problematic

Let ℓ be a loss function and $\mathcal{H}$ a hypothesis space. We define the optimal target hypothesis $h^* \in \mathcal{H}$ as :

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \underset{(x,y) \sim P_T(X,Y)}{\mathbb{E}} [\ell(h(x), y)] \tag{3}$$

The goal of UDA is to find the best possible estimation of h^* based on the source labeled sample $\mathcal{S}$ and the target unlabeled sample $\mathcal{T}_{\mathcal{X}}$

Instance-based approach

Assuming $P_S(X,Y), P_T(X,Y)$ have density functions $p_s(x,y), p_t(x,y)$ and $w(x,y) = p_t(x,y)/p_s(x,y)$:

$$\begin{aligned}
\underset{(x,y)\sim P_T(X,Y)}{\mathbb{E}}[\ell(h(x),y)] &= \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_t(x,y)\,\ell(h(x),y)dxdy \\
&= \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \frac{p_t(x,y)}{p_s(x,y)} p_s(x,y)\,\ell(h(x),y)dxdy \\
&= \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} w(x,y)p_s(x,y)\,\ell(h(x),y)dxdy \\
&= \underset{(x,y)\sim P_S(X,Y)}{\mathbb{E}}[w(x,y)\ell(h(x),y)]
\end{aligned}$$

The target risk can be written as a "reweighted" source risk :

$$\underset{(x,y)\sim P_T(X,Y)}{\mathbb{E}}[\ell(h(x),y)] = \underset{(x,y)\sim P_S(X,Y)}{\mathbb{E}}[w(x,y)\ell(h(x),y)]$$

Instance-based approach : Density ratio

Definition 2 (Density Ratio)

Let be $p_s(x, y)$ and $p_t(x, y)$ the respective source and target density functions such that $\text{supp}(p_T(x, y)) \subset \text{supp}(p_S(x, y))$. The **density ratio** or **importance weight** is defined, for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$, as :

$$w(x, y) = \frac{p_t(x, y)}{p_s(x, y)}$$

Remark : the support of the target distribution is required to be included in the support of the source distribution, i.e :

$$p_s(x, y) = 0 \implies p_t(x, y) = 0, \;\; \forall\, (x, y) \in \mathcal{X} \times \mathcal{Y}$$

Problem : $w(x, y)$ involves $p_t(x, y) = p_t(y|x)p_t(x)$ which requires target labels to be estimated.

Instance-based approach : Covariate-Shift Assumption

Definition 3 (Covariate Shift)

The source and target joint distributions $P_S(X, Y)$ and $P_T(X, Y)$ follows the **covariate-shift** assumption if, for any $x \in \mathcal{X}$:

$$P_T(Y|X = x) = P_S(Y|X = x) \tag{4}$$

Proposition 1

*Let $p_s(x, y)$ and $p_t(x, y)$ be the source and target density functions. If the **covariate-shift** assumption holds then, for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$, the **density ratio** can be written :*

$$w(x, y) = w(x) = \frac{p_t(x)}{p_s(x)} \tag{5}$$

Proof : $w(x, y) = \frac{p_t(x,y)}{p_s(x,y)} = \frac{p_t(y|x)p_t(x)}{p_s(y|x)p_s(x)} = \frac{p_t(x)}{p_s(x)}$

Examples Covariate-Shift

Definition 4 (Covariate Shift)

The source and target joint distributions $P_S(X, Y)$ and $P_T(X, Y)$ follows the **covariate-shift** assumption if, for any $x \in \mathcal{X}$:

$$P_T(Y|X = x) = P_S(Y|X = x) \qquad (6)$$

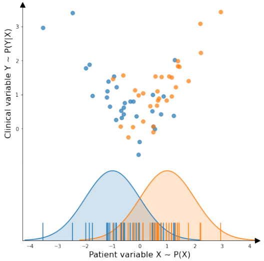


Figure – (A)

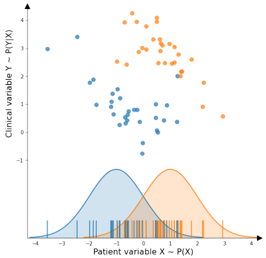


Figure – (B)

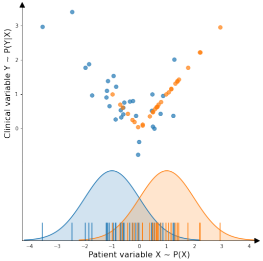


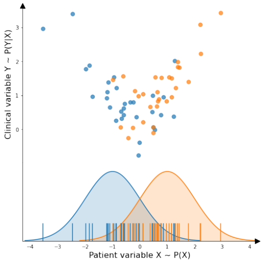
Figure – (C)

Examples Covariate-Shift

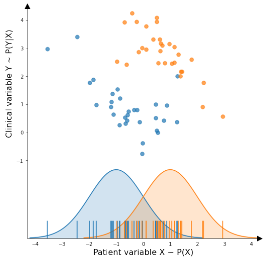
Definition 5 (Covariate Shift)

The source and target joint distributions $P_S(X,Y)$ and $P_T(X,Y)$ follows the **covariate-shift** assumption if, for any $x \in \mathcal{X}$:

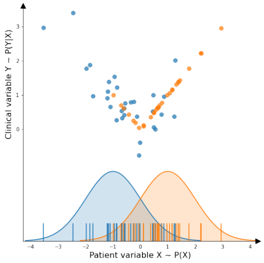
$$P_T(Y|X=x) = P_S(Y|X=x) \tag{7}$$



$P_S(Y|x) \sim \mathcal{N}(|x|,1)$
$P_T(Y|x) \sim \mathcal{N}(|x|,1)$



$P_S(Y|x) \sim \mathcal{N}(|x|,1)$
$P_T(Y|x) \sim \mathcal{N}(4-|x|,1)$



$P_S(Y|x) \sim \mathcal{N}(|x|,1)$
$P_T(Y|x) \sim \mathcal{N}(|x|,0.01)$

Covariate-Shift in real life

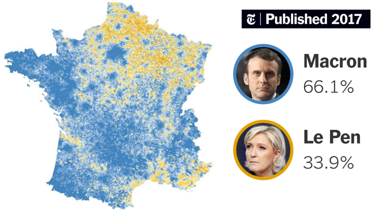


Figure – **Election Polls** (source : nytimes.com). It is generally assumed that people give the same vote for the survey and the election.

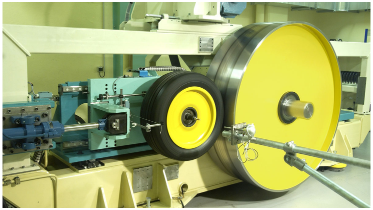


Figure – **Industrial Predictive Models** (source : michelin.com). The measuring machine used to build the training set is supposed to be the same than the one used for the target set (ex : they have the same measurement noise)

Instance-based approach

We have shown that, for any $h \in \mathcal{H}$:

$$\mathop{\mathbb{E}}_{(x,y)\sim P_T(X,Y)}[\ell(h(x),y)] = \mathop{\mathbb{E}}_{(x,y)\sim P_S(X,Y)}[w(x,y)\ell(h(x),y)]$$

Under the **covariate-shift** assumption, we have :

$$\mathop{\mathbb{E}}_{(x,y)\sim P_T(X,Y)}[\ell(h(x),y)] = \mathop{\mathbb{E}}_{(x,y)\sim P_S(X,Y)}[w(x)\ell(h(x),y)]$$

$\Longrightarrow$ Assuming that $w(x)$ is known for any $x \in \mathcal{X}$, the target risk of $h \in \mathcal{H}$ can be estimated thanks to the source sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$:

$$\mathop{\mathbb{E}}_{(x,y)\sim P_T(X,Y)}[\ell(h(x),y)] \approx \frac{1}{m}\sum_{i=1}^{m} w(x_i)\ell(h(x_i),y_i)$$

Outline

Instance-based approach

We have shown that, for any $h \in \mathcal{H}$:

$$\underset{(x,y)\sim P_T(X,Y)}{\mathbb{E}}[\ell(h(x),y)] \approx \frac{1}{m}\sum_{i=1}^{m} w(x_i)\ell(h(x_i),y_i)$$

Moreover, the quantity $w(x_i) = p_t(x_i)/p_t(x_i)$ does not involve labels and can be estimated using only the source and target input samples $S_{\mathcal{X}} = \{x_1, ..., x_m\} \sim P_S(X)$, $T = \{x'_1, ..., x'_m\} \sim P_T(X)$

UDA Instance-based

A UDA instance-based approach consist in solving the following optimization problem :

$$\text{Estimating : } \ \widehat{w}(x_i) \ \ \forall x_i \in S_{\mathcal{X}}$$

$$\widehat{h^*} = \underset{h\in\mathcal{H}}{\operatorname{argmin}} \frac{1}{m}\sum_{i=1}^{m} \widehat{w}(x_i)\ell(h(x_i),y_i)$$

Exercise : Weighted Regression

Setup

Let's consider the regression problem with $X \in \mathbb{R}^{m \times p}$ the matrix of input data $\{x_1, ..., x_m\}$ such that $x_i \in \mathbb{R}^p$ is the i$^{\text{th}}$ row of X and $Y \in \mathbb{R}^{m \times 1}$ the vector of output data $\{y_1, ..., y_m\}$ with $y_i \in \mathbb{R}$ for any $i \in [|1, m|]$.
We consider the UDA instance-based problem where $w_i = p_t(x_i)/p_s(x_i)$ is assumed to be known for any x_i. The loss function is the squared error : $\ell(y, y') = (y - y')^2$ and $\mathcal{H}$ the hypothesis space of linear hypotheses $\mathcal{H} = \{h : x \to x\beta^T, \beta \in \mathbb{R}^p\}$

Question :

Find a close-form solution β^* of the UDA instance-based problem involving X, Y and $\Delta = \text{diag}(\sqrt{w_1}, ..., \sqrt{w_m})$.

Exercise : Weighted Regression

Solution :

The problem can be formulated as follows :

$$\beta^* = \underset{\beta \in \mathbb{R}^p}{\text{argmin}} \sum_{i=1}^{m} w_i (x_i \beta^T - y_i)^2$$

By denoting $\Delta = \text{diag}(\sqrt{w_1}, ..., \sqrt{w_m})$, we have :

$$\begin{aligned}\beta^* &= \underset{\beta \in \mathbb{R}^p}{\text{argmin}} ||\Delta(X\beta^T - Y)||_2^2 \\ &= \underset{\beta \in \mathbb{R}^p}{\text{argmin}} ||\Delta X\beta^T - \Delta Y||_2^2\end{aligned}$$

Assuming $X^T \Delta^2 X$ invertible, we have :

$$\begin{aligned}\beta^* &= [(\Delta X)^T \Delta X]^{-1} (\Delta X)^T \Delta Y \\ &= [X^T \Delta^2 X]^{-1} X^T \Delta^2 Y\end{aligned}$$

UDA : Example

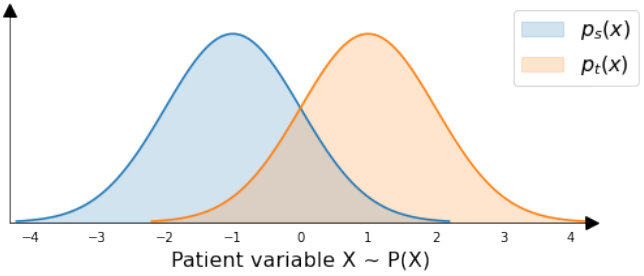


Figure – Let's consider the two shifted Gaussians problem introduced previously.

UDA : Example

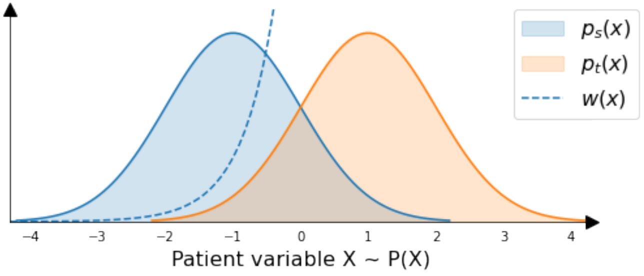


Figure – If the density functions of both domains are known, we can derive the density ratio $w(x)$ (here in dashed line).

UDA : Example

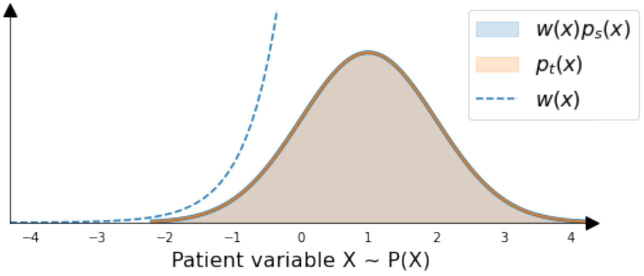


Figure – We can see that the target density to the source reweighted density match perfectly.

UDA : Example

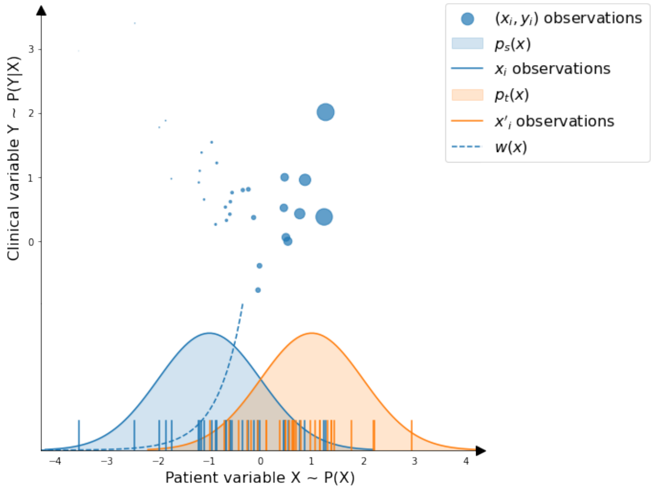


Figure – We present here the source obsrevations (x_i, y_i) sized with their corresponding weights $w(x_i)$ (the larger $w(x_i$, the larger the size of the blue points).

UDA : Example

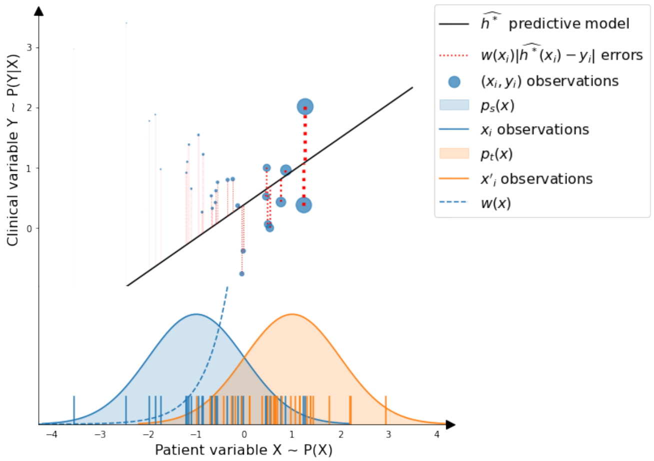


Figure – We now fit a linear model with the reweighted objective.

UDA : Example

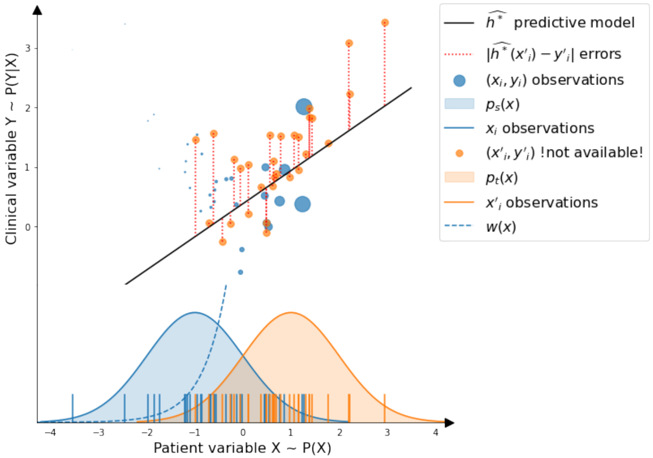


Figure – We observe that the learned linear model is closer to the best target model than previously.

UDA : Example

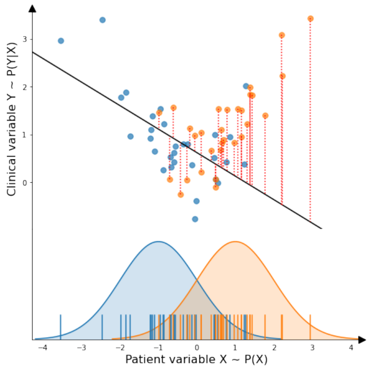


Figure – Without Importance Weighting

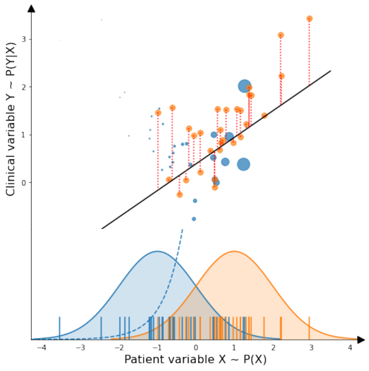


Figure – With Importance Weighting

Instance-based approach

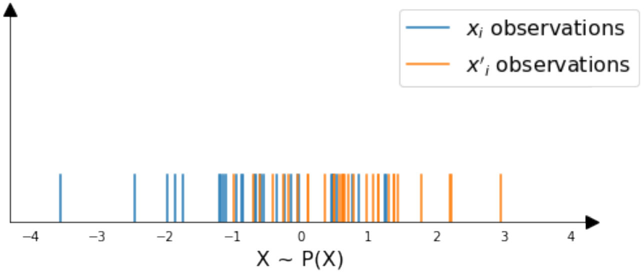


Figure – Generally, $p_s(x)$ and/or $p_t(x)$ are not known. The learner has only access to samples $\mathcal{S}_{\mathcal{X}}$ and $\mathcal{T}_{\mathcal{X}}$ respectively drawn according to $P_S(X)$ and $P_T(X)$.

Problem : How can we estimate $w(x)$ based on $\mathcal{S}_{\mathcal{X}}$ and $\mathcal{T}_{\mathcal{X}}$?

Instance-based approach : Summary

Multiple approaches exist to estimate $w(x)$ and can be summarized as follows :

Density Estimation :

- Parametric Density Estimation
- Kernel Density Estimation

Heuristic :

- Nearest Neighbors Weighting [Loog, 2012]

Distribution Matching :

- Kullback–Leibler Importance Estimation Procedure (KLIEP) [Sugiyama et al., 2007]
- Kernel Mean Matching [Huang et al., 2007]

Domain Classifier

- Linear/Kernel Classifier [Bickel et al., 2007]
- Neural Network Classifier [Zhang et al., 2018]

Instance-based approach : Density Estimation

Density Estimation

Density Estimation approaches consist in estimating $p_s(x)$, $p_t(x)$ based on the respective samples $\mathcal{S}_\mathcal{X}$ and $\mathcal{T}_\mathcal{X}$. Then, the density ratio is computed for any $x_i \in \mathcal{S}_\mathcal{X}$ as follows :

$$\widehat{w}(x_i) = \frac{\widehat{p_t}(x_i)}{\widehat{p_s}(x_i)}$$

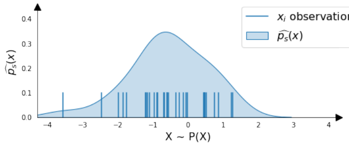


Figure – 1D Density Estimation

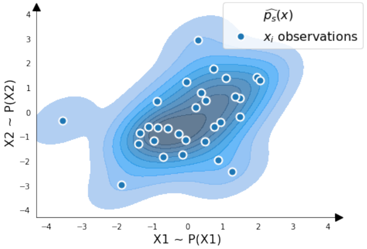


Figure – 2D Density Estimation

Instance-based approach : Parametric Density Estimation

Parametric Density Estimation

The learner assumes a prior input distribution with densities $p_s(\theta_s, x)$, $p_t(\theta_t, x)$ for the source and target domains respectively. The parameters θ_s and θ_t of the distributions are then estimated through Maximum-Likelihood estimation.

$$\widehat{\theta_s^*} = \underset{\theta_s \in \Theta_s}{\operatorname{argmin}} \sum_{x_i \in \mathcal{S}_{\mathcal{X}}} \log(p_s(\theta_s, x_i))$$

$$\widehat{\theta_t^*} = \underset{\theta_t \in \Theta_t}{\operatorname{argmin}} \sum_{x_i' \in \mathcal{T}_{\mathcal{X}}} \log(p_t(\theta_t, x_i'))$$

$$\widehat{w}(x_i) = \frac{p_t(\widehat{\theta_t^*}, x_i)}{p_s(\widehat{\theta_s^*}, x_i)}$$

Instance-based approach : Parametric Density Estimation

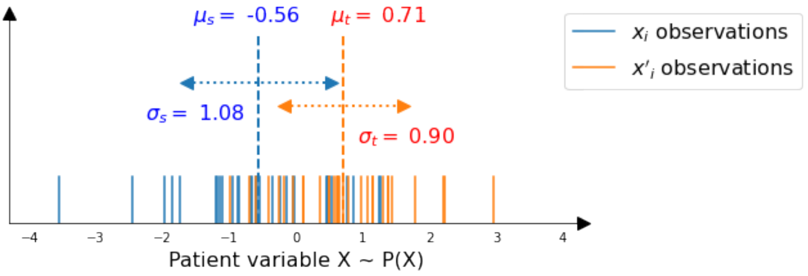


Figure – To perform density estimation with a Gaussian prior, the learner estimates the mean and standard deviations of both distributions based on $\mathcal{S}_\mathcal{X}$ and $\mathcal{T}_\mathcal{X}$.

Instance-based approach : Parametric Density Estimation

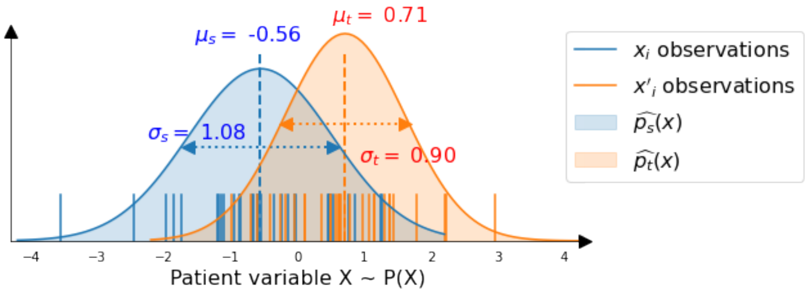


Figure – To perform density estimation with a Gaussian prior, the learner estimates the mean and standard deviations of both distributions based on $\mathcal{S}_\mathcal{X}$ and $\mathcal{T}_\mathcal{X}$.

Instance-based approach : KDE (agnostic)

Kernel Density Estimation

Let's consider $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$ a kernel function. The learner computes estimations of the densities $p_s(x)$, $p_t(x)$ as sums of kernels centered in each observation $x_i \in \mathcal{S}_\mathcal{X}$ (resp $x'_i \in \mathcal{T}_\mathcal{X}$).

$$\widehat{p_s}(x) = C_s \sum_{x_i \in \mathcal{S}_\mathcal{X}} k(x, x_i)$$

$$\widehat{p}_t(x) = C_t \sum_{x'_i \in \mathcal{T}_\mathcal{X}} k(x, x'_i)$$

$$\widehat{w}(x_i) = \frac{\widehat{p}_t(x_i)}{\widehat{p_s}(x_i)}$$

Remark : A typical kernel function is the Gaussian kernel :
$k : (x, x') \rightarrow \exp(-\frac{||x-x'||_2^2}{2\sigma^2})$. $\sigma \in \mathbb{R}_+^*$ is the kernel bandwidth and can be chosen through Maximum Likelihood estimation.

Instance-based approach : Kernel Density Estimation

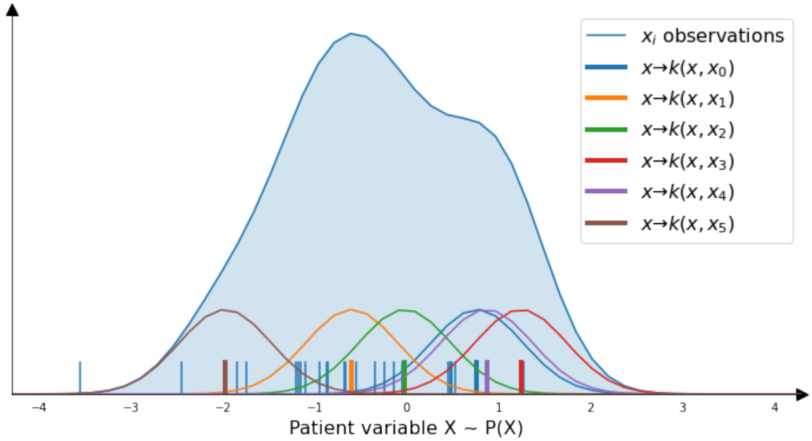


Figure – Kernel Density Estimation consist in aggregating multiple densities functions centered in each observation.

Instance-based approach : Nearest Neighbors Weighting

Nearest Neighbors Weighting

Let's consider $K \in \mathbb{N}^*$ and define $\mathcal{V}_K(x'_j)$ as the set of K-Nearest-Neighbors of $x'_j \in \mathcal{T}_{\mathcal{X}}$ in $\mathcal{S}_{\mathcal{X}}$.

The density ratio $\widehat{w}(x_i)$ is directly estimated through the following heuristic :

$$\widehat{w}(x_i) = \text{Card}\left(\left\{x'_i \in \mathcal{T}_{\mathcal{X}} \mid x_i \in \mathcal{V}_K(x'_j)\right\}\right)$$

Remark : if $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ is a distance over $\mathcal{X}$, a formal definition of $\mathcal{V}_K(x'_j)$ can be written as follows :

$$\mathcal{V}_K(x'_j) = \left\{x \in \mathcal{S}_{\mathcal{X}};\ \sum_{x_i \in \mathcal{S}_{\mathcal{X}}} \mathbb{1}\left(d(x_i, x'_j) < d(x, x'_j)\right) < K\right\}$$

Instance-based approach : Nearest Neighbors Weighting

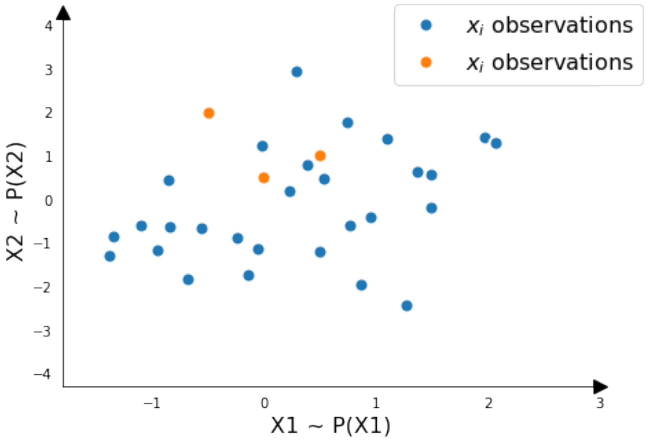


Figure – Let's consider a source and target input samples drawn according to different distributions in $\mathbb{R}^2$

Instance-based approach : Nearest Neighbors Weighting

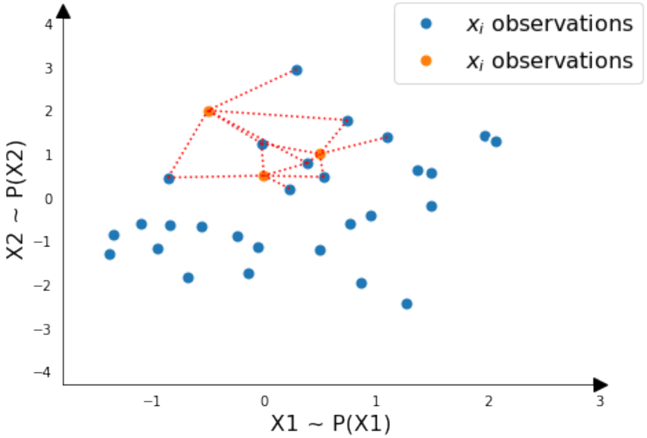


Figure – The Nearest Neighbors Weighting first computes the distance from each target data (orange) to every source data (blue). Then, it selects the K closest source data for each target (here $K = 5$).

Instance-based approach : Nearest Neighbors Weighting

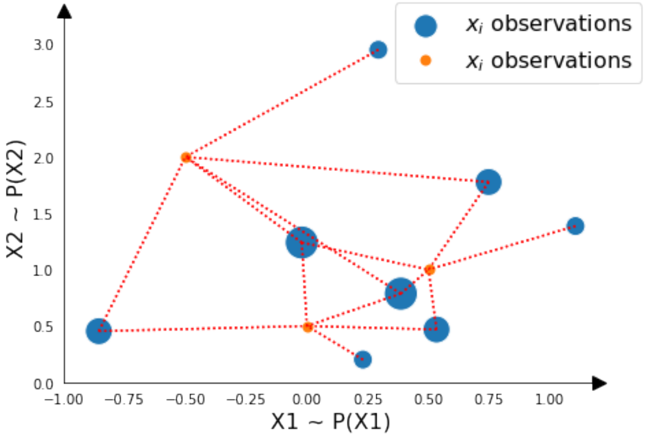


Figure – Finally, each source data is reweighted according to the number of times it has been selected as nearest neighbor by a target data.

Instance-based approach : Distribution Matching

Density Estimation

Distribution Matching approaches consist in directly estimating the density ratio by looking for the weighting $w(x)$ that minimizes a "discrepancy metric" between $w(x)p_s(x)$ and $p_t(x)$:

$$w^* = \underset{w}{\text{argmin}}\, D(p_t(x), w(x)p_s(x))$$

With D a metric measuring the "closeness" of two distributions.

Problematic : How to measure distribution "closeness"?

Instance-based approach : Distribution Matching

Problematic : How to measure distribution "closeness" ?

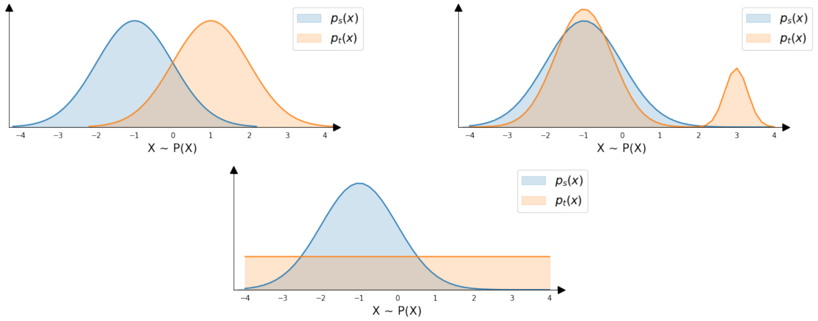


Figure – Which pair of distributions are the closest ?

Instance-based approach : Distribution Matching

Problematic : How to measure distribution "closeness" ?

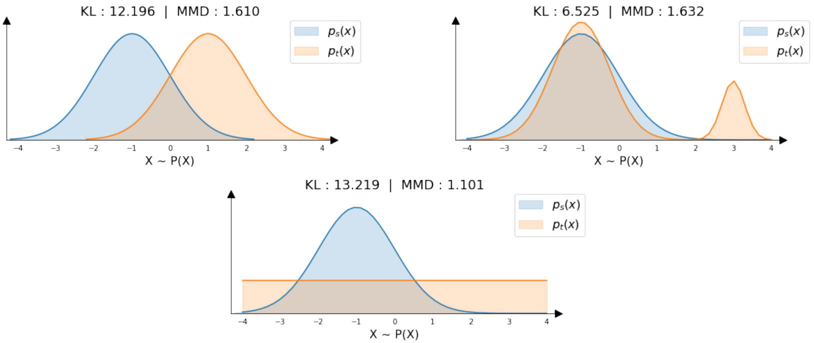


Figure – The notion of "distribution closeness" depends on the considered metric.

Instance-based approach : Distribution Matching

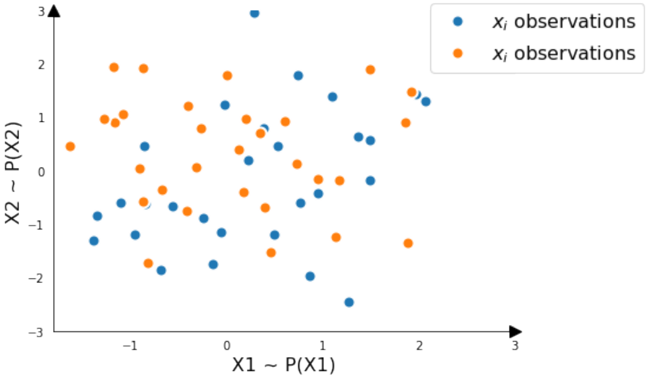


Figure – Let's again consider a source and target input samples drawn according to different distributions in $\mathbb{R}^2$

Instance-based approach : Distribution Matching

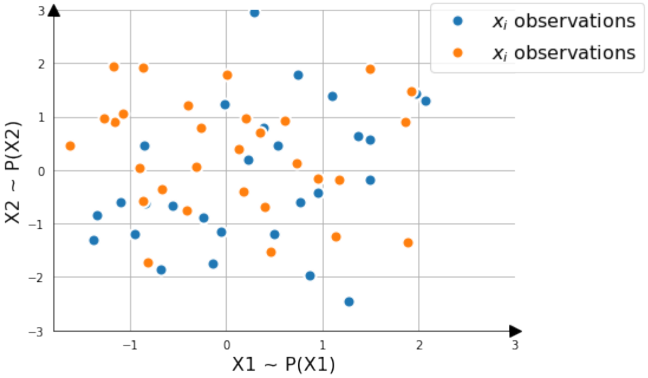


Figure – Computing the Kullback-Leibler divergence between these two samples can be done by splitting the space in several regions, then compare, in each region, the proportion of source and target samples.

Instance-based approach : Distribution Matching

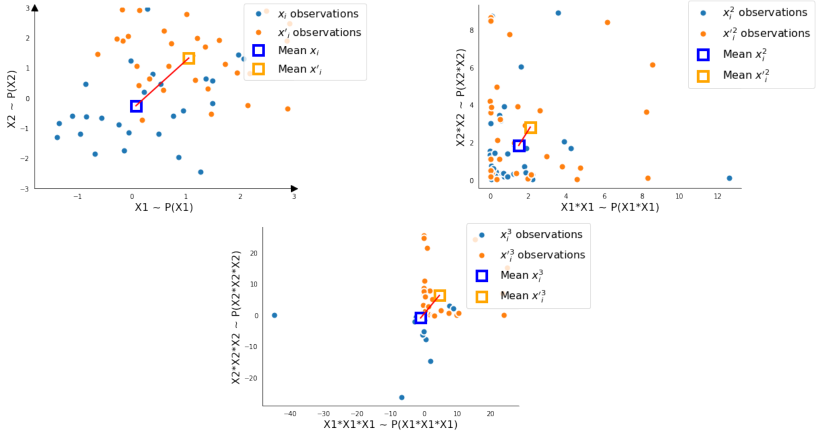


Figure – The Maximum Mean Discrepancy can be seen as a "moment" comparison metric : for several $k \in \mathbb{N}$, we compare $\mathbb{E}_{P_S(X)}[X^k]$ to $\mathbb{E}_{P_T(X)}[X^k]$. If, for any $k \in \mathbb{N}$, $\mathbb{E}_{P_S(X)}[X^k] = \mathbb{E}_{P_T(X)}[X^k]$ then $P_S(X) = P_T(X)$

Instance-based approach : KLIEP

Kullback–Leibler Importance Estimation Procedure (KLIEP)

The KLIEP algorithm looks for the density ratio $w(x)$ which minimizes the Kullback-Leibler divergence between the reweighted source distribution $w(x)p_s(x)$ and the target distribution $p_t(x)$:

$$w^* = \underset{w}{\operatorname{argmin}}\, \mathrm{KL}\left(p_t(x), w(x)p_s(x)\right)$$

Optimization : Writing w as a linear combination of kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$: $w(x) = \sum_{j=1}^{b} \alpha_j k(x, x'_j)$, the optimization algorithm solved by KLIEP is written :

$$\alpha^* = \underset{\alpha \in \mathbb{R}^b}{\operatorname{argmin}} \sum_{x'_i \in \mathcal{T}_\mathcal{X}} \log \left(\sum_{j=1}^{b} \alpha_j k(x'_i, x'_j) \right)$$

$$\text{sc} \sum_{x_i \in \mathcal{S}_\mathcal{X}} \sum_{j=1}^{b} \alpha_j k(x_i, x'_j) = 1 \text{ and } \alpha \geq 0$$

Instance-based approach : Kernel Mean Matching

Kernel Mean Matching (KMM)

The KMM algorithm looks for the density ratio $w(x)$ which minimizes the Maximum Mean Discrepancy (MMD) between the reweighted source distribution $w(x)p_s(x)$ and the target distribution $p_t(x)$:

$$w^* = \underset{w}{\operatorname{argmin}} \operatorname{MMD}\left(p_t(x), w(x)p_s(x)\right)$$

Optimization : Using a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, the optimization algorithm solved by KMM is written :

$$w^* = \underset{w \in \mathbb{R}^{m\times 1}; w \geq 0, w^T \mathbf{1} = m}{\operatorname{argmin}} \quad w^T K w - 2\kappa^T w$$

Where $K \in \mathbb{R}^{m\times m}$ and $\kappa \in \mathbb{R}^{m\times 1}$ such that : $K_{ij} = k(x_i, x_j)$ and $\kappa_i = \frac{m}{n} \sum_{x_j' \in \mathcal{T}_\mathcal{X}} k(x_i, x_j')$ with $x_i, x_j \in \mathcal{S}_\mathcal{X}$

Instance-based approach : Importance Weighting Classifier

Importance Weighting Classifier

The underlying idea of the Importance Weighting Classifier algorithm is to relate the density ratio $w(x)$ to the probability that x belongs to the source domain rather than the target domain. This probability is computed with a classifier $\phi : \mathcal{X} \to [0,1]$ trained to discriminate between $\mathcal{S}_{\mathcal{X}}$ and $\mathcal{T}_{\mathcal{X}}$:

$$\phi^* = \underset{\phi \in \Phi}{\text{argmin}} \sum_{x \in \mathcal{S}_{\mathcal{X}}} \mathbb{1}\left(\phi(x) > \frac{1}{2}\right) + \sum_{x \in \mathcal{T}_{\mathcal{X}}} \mathbb{1}\left(\phi(x) < \frac{1}{2}\right)$$

$$w(x) = \frac{1}{\phi(x)} - 1$$

With Φ a set of classifiers.

Remark : The choice of the classifier set Φ drives the complexity of the weighting w (ex : linear models, neural networks, random forest...)

Instance-based approach : Importance Weighting Classifier

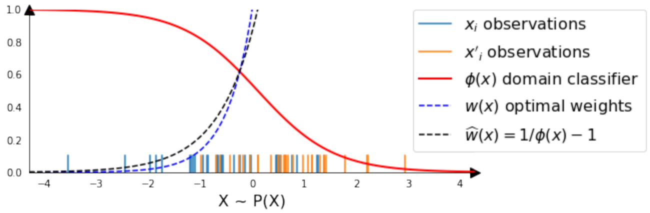


Figure – Here, a linear classifier is used to discriminate between the source (blue) and target (orange) samples.

Instance-based approach : Importance Weighting Classifier

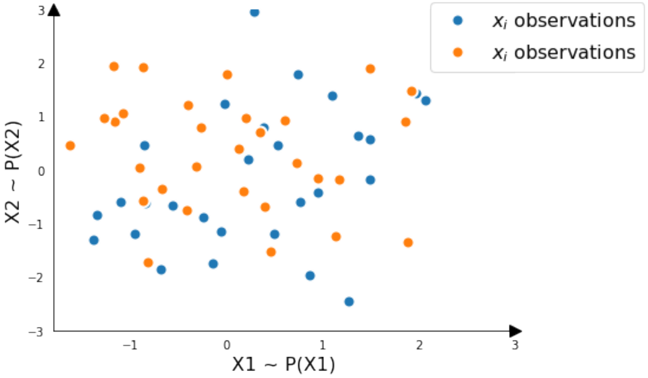


Figure – Here, a neural network classifier is used to discriminate between the source (blue) and target (orange) samples.

Instance-based approach : Importance Weighting Classifier

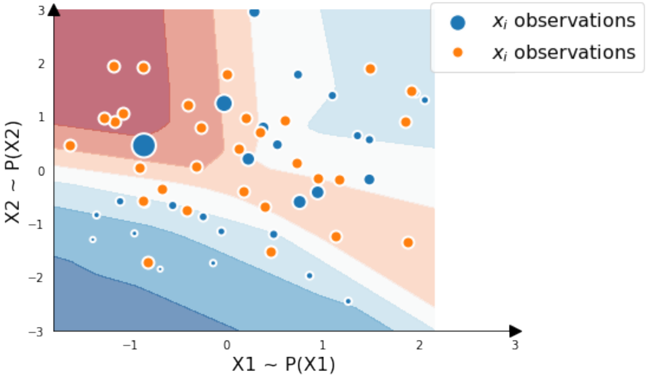


Figure – Here, a neural network classifier is used to discriminate between the source (blue) and target (orange) samples.

Exercise : Importance Weighting Classifier

Setup

Let's consider the source and target densities $p_s(x)$ and $p_t(x)$ such that $\mathrm{supp}(p_t(x)) \subset \mathrm{supp}(p_s(x))$. Let's define $\phi^*(x)$ as the best possible domain classifier with respect to the cross-entropy loss function :

$$\phi^* = \underset{\phi}{\mathrm{argmax}}\, \mathbb{E}_{P_S(X)}[\log(\phi(X))] + \mathbb{E}_{P_T(X)}[\log(1 - \phi(X))]$$

Question : Show that, for any $x \in \mathcal{X}$:

$$w(x) = \frac{1}{\phi^*(x)} - 1$$

With $w(x) = \frac{p_t(x)}{p_s(x)}$

Exercise : Importance Weighting Classifier

Solution :

$$
\begin{aligned}
\phi^* &= \operatorname*{argmax}_{\phi} \mathbb{E}_{P_S(X)}[\log(\phi(X))] + \mathbb{E}_{P_T(X)}[\log(1-\phi(X))] \\
&= \operatorname*{argmax}_{\phi} \int_{x\in\mathcal{X}} p_s(x)\log(\phi(x))dx + \int_{x\in\mathcal{X}} p_t(x)\log(1-\phi(x))dx \\
&= \operatorname*{argmax}_{\phi} \int_{x\in\mathcal{X}} \left(p_s(x)\log(\phi(x)) + p_t(x)\log(1-\phi(x))\right) dx
\end{aligned}
$$

For any $x \in \mathcal{X}$, we can show that $\log(\phi(x)) + p_t(x)\log(1-\phi(x))$ is maximized for $\phi^*(x) = \frac{p_s(x)}{p_s(x)+p_t(x)}$. Then, we conclude that :

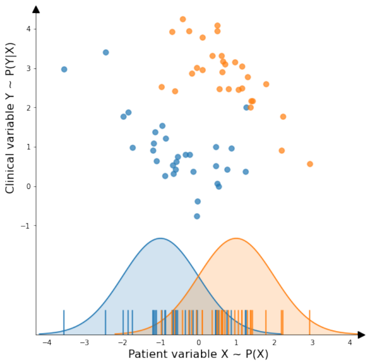
$$w(x) = \frac{1}{\phi^*(x)} - 1$$

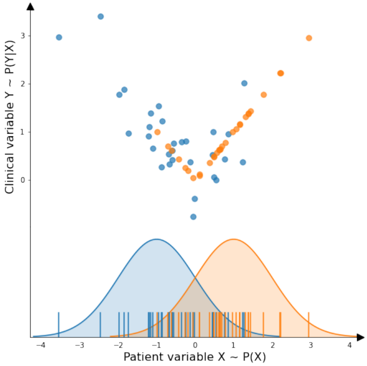
Instance-based approach : What if no covariate-shift ?

Definition 6 (Covariate Shift)

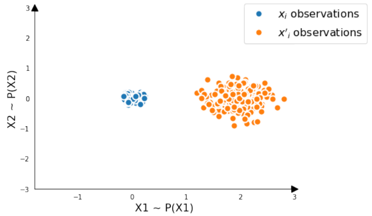
The source and target joint distributions $P_S(X, Y)$ and $P_T(X, Y)$ follows the **covariate-shift** assumption if, for any $x \in \mathcal{X}$:

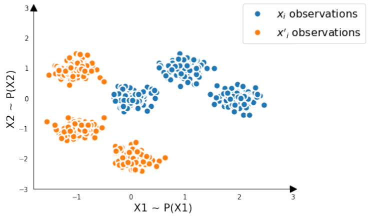
$$P_T(Y|X = x) = P_S(Y|X = x) \quad (8)$$

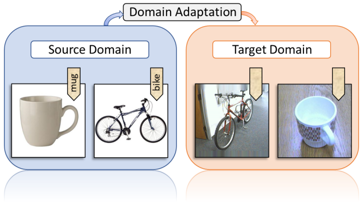


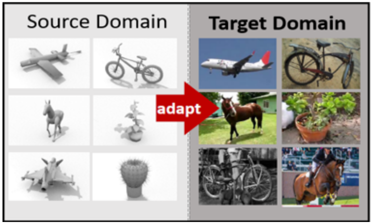


Instance-based approach : What if no common support ?

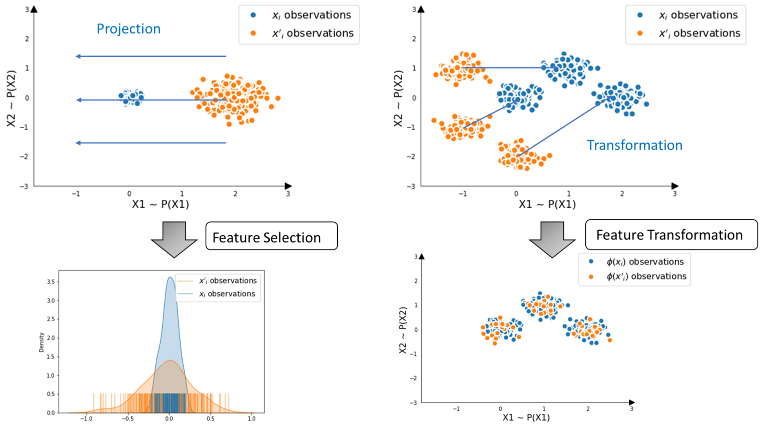








Opening : feature-based approach



Outline

Conclusion

- Domain shifts are encountered in many real-life scenario and can be detrimental for machine learning models.
- Transfer learning and domain adaptation tools propose a large pannel of approaches to correct the domain shifts.
- Instance-based approaches are particularly suited to handle unsupervised domain adaptation under covariate-shift.
- The challenge of this kind of method is to find the faster and more accurate way to estimate the density ratio.

Bibliography

Bickel, S., Brückner, M., and Scheffer, T. (2007).
Discriminative learning for differing training and test distributions.
In *Proceedings of the 24th international conference on Machine learning*, pages 81–88.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., and Smola, A. J. (2007).
Correcting sample selection bias by unlabeled data.
In Schölkopf, B., Platt, J. C., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*, pages 601–608. MIT Press.

Loog, M. (2012).
Nearest neighbor-based importance weighting.
In *2012 IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE.

Sugiyama, M., Nakajima, S., Kashima, H., Bünau, P. v., and Kawanabe, M. (2007).
Direct importance estimation with model selection and its application to covariate shift adaptation.
In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, page 1433–1440, Red Hook, NY, USA. Curran Associates Inc.

Zhang, J., Ding, Z., Li, W., and Ogunbona, P. (2018).
Importance weighted adversarial nets for partial domain adaptation.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8156–8164.