# Flavours of Physics Challenge: Physics Prize Transfer Learning approach

Alexander Rakhlin rakhlin@gmx.net

Heavy Flavour Data Mining workshop
18 February 2016, Zurich

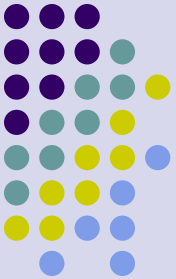
kaggle University of Zurich CERN THE UNIVERSITY OF WARWICK Yandex Data Factory intel LHCb SCHOOL OF DATA ANALYSIS

# Abstract

When modeling a classifier on simulated data, it is possible to reach a high performance by picking features that are not perfectly modeled in the simulation. To avoid this problem in Kaggle's "Flavours of Physics: Finding $\tau^- \to \mu^- \mu^- \mu^+$ " competition a proxy channel with similar topology and well-studied characteristics was introduced.

This proxy channel was intended for validation of classifier trained on another channel of interest. We show that such validation scheme is questionable from a point of view of statistical inference as it violates fundamental assumption in Machine Learning that training and test data follow the same probability distribution.

# Proposed solution: Transfer Learning



We relate the problem to known paradigm in Machine Learning – Transfer Learning between different underlying distributions.

We propose a solution that brings the problem to transductive transfer learning (TTL) and simple covariate shift, a primary assumption in domain adaptation framework.

Finally, we present transfer learning model (one of a few) that finished the competition on the 5 place.

# 1. INTRODUCTION

# Description

Like 2014’ Higgs Boson Kaggle Challenge, this competition dealt with the physics at the Large Hadron Collider.

However, the Higgs Boson was already known to exist. The goal of 2015’ challenge was to design a classifier capable to discriminate a phenomenon that is not already known to exist – charged lepton flavour violation – thereby helping to establish "new physics" (from the competition description)

# The data

For training the classifier participants were given ~800,000 signal and background samples of labeled data from $\tau^- \to \mu^- \mu^- \mu^+$ channel.

Background events come from real data, whereas signal events come from Monte-Carlo simulation. Hereinafter:

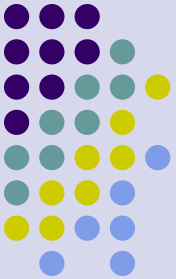
- $\tau^- \to \mu^- \mu^- \mu^+$ channel shall be referred to as $\tau$ channel
- its background data samples – as $X^{\tau}_{data}$
- its Monte-Carlo simulated signal data samples – as $X^{\tau}_{MC}$
- The signal being “1” for events where the decay did occur, “0” for background events where it did not.

# Control channel

Since the classifier was trained on simulated data for the signal and on real data for the background, it was possible to reach a high performance by picking features that are not perfectly modeled in the simulation.

To address this problem a Control channel with known characteristics was introduced, and classifier was required not to have large discrepancy when applied to $\tau$ and control channel. Kolmogorov–Smirnov test (Agreement test) was used to make sure that classifier performance does not vary significantly between the channels.

# Control channel (continued)



Control channel $D_s^+ \to \phi(\to \mu^- \mu^+)\pi^+$ has a similar topology as the $\tau$ channel, and is a much more well-known, well-observed behavior, as it happens more frequently.

Just like for $\tau$ channel participants were provided with both Monte-Carlo simulated and real data for $D_s^+ \to \phi(\to \mu^- \mu^+)\pi^+$ decay to which the classifier can be verified. In addition, sPlot weights were assigned to real data. Higher weight means an event is likely to be signal, lower weight means it's likely to be background. Real data is a mix of real decay and background events with large prevalence of background. Hereinafter:

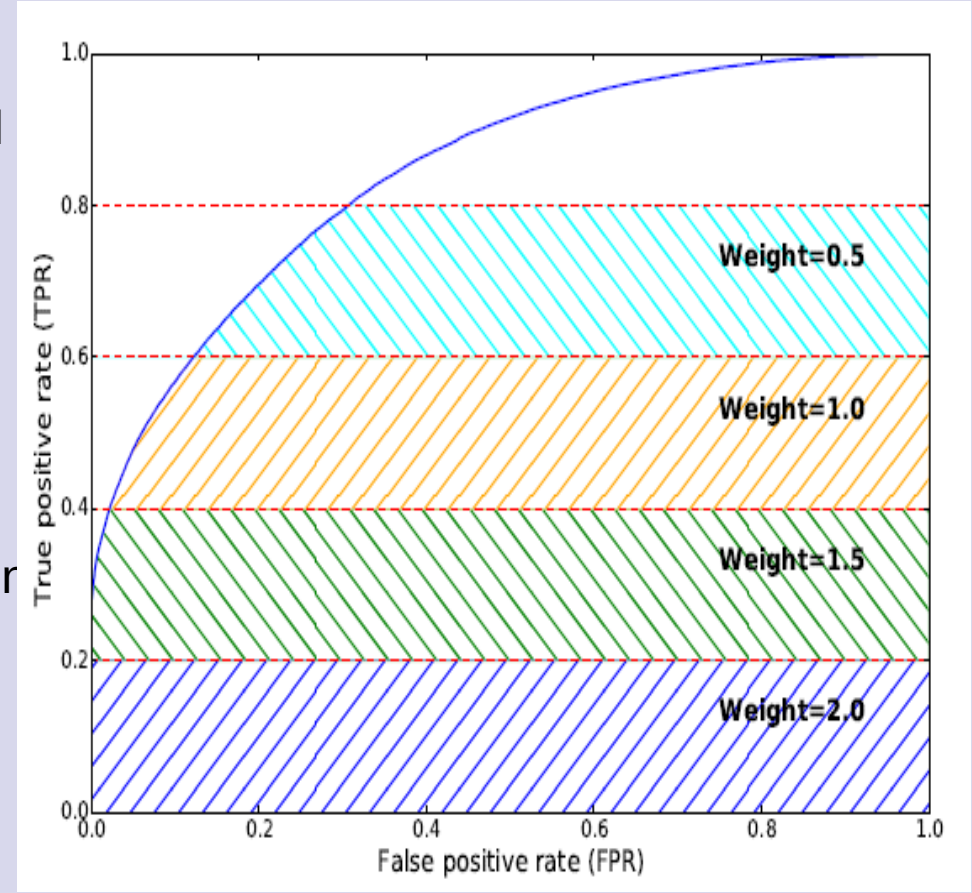
- $D_s^+ \to \phi(\to \mu^- \mu^+)\pi^+$ channel shall be referred to as $Ds$ −channel
- its real data samples (mainly background) – as $X_{data}^{Ds}$
- its Monte-Carlo simulated signal samples – as $X_{MC}^{Ds}$

# Evaluation

The test dataset consists of mix of samples from $\tau$ and $Ds$ channels (real data and Monte Carlo simulated).

The evaluation metric for this competition is Weighted Area Under the ROC Curve (AUC).

Only $\tau$ samples are used for evaluation of AUC. $Ds$ samples are ignored for ROC scoring and used in Agreement test.




From documentation of the challenge

# No training on $Ds$ channel allowed

$Ds$ channel was not intended for training, its purpose was *validation* of already trained classifier in KS-test.

This requirement became the main challenge of "Flavours of Physics: Finding $\tau^- \to \mu^- \mu^- \mu^+$", and constitutes central point of this presentation.

# The problem: validation on $Ds$ channel is questionable

In the following we demonstrate that $\tau$ and $Ds$ channels have significantly different probability distributions.

On the other hand, Machine Learning theory shall remind us why testing on data that has different probability distribution than data where the model was created violates …

> … the assumption of a single underlying generative process between the train set and the test set.


This assumption is the cornerstone of statistical learning and, in general, of inference.

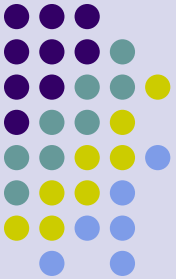
# Our plan

- The following section is dedicated to foundations of Machine Learning theory. We demonstrate there that $\tau$ and $Ds$ channels have different underlying probability distributions and argue why it is bad.
- In "Transfer learning" section we propose a framework for learning from different distributions.
- "Implementation" section discusses solution presented for this competition and sets some guidelines for further development.
- "Kolmogorov–Smirnov test" section is optional but provides some interesting insight.

Let us briefly recall the fundamentals of Machine Learning theory.

# 2. BRIEF REVIEW OF MACHINE LEARNING FOUNDATIONS

# Empirical risk minimization

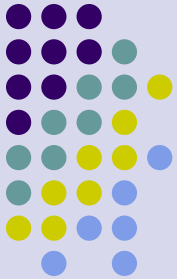


Standard Machine Learning approach is to try to minimize the training error. This algorithm picks hypothesis such that:

$$\hat{h} = \arg\min_{h \in H} \hat{\varepsilon}(h)$$

This process is called empirical risk minimization (ERM). Let us recall how it works and WHY it works

# Building blocks of the general (supervised) statistical learning process



$(X_i, y_i)$ – set of independent and identically distributed (iid) training examples drawn from unknown input distribution $P(X)$

$\mathcal{H}$ – hypothesis set, or the set of all classifiers considered by learning algorithm. This can be, for example, a set of all linear classifiers in $R^n$

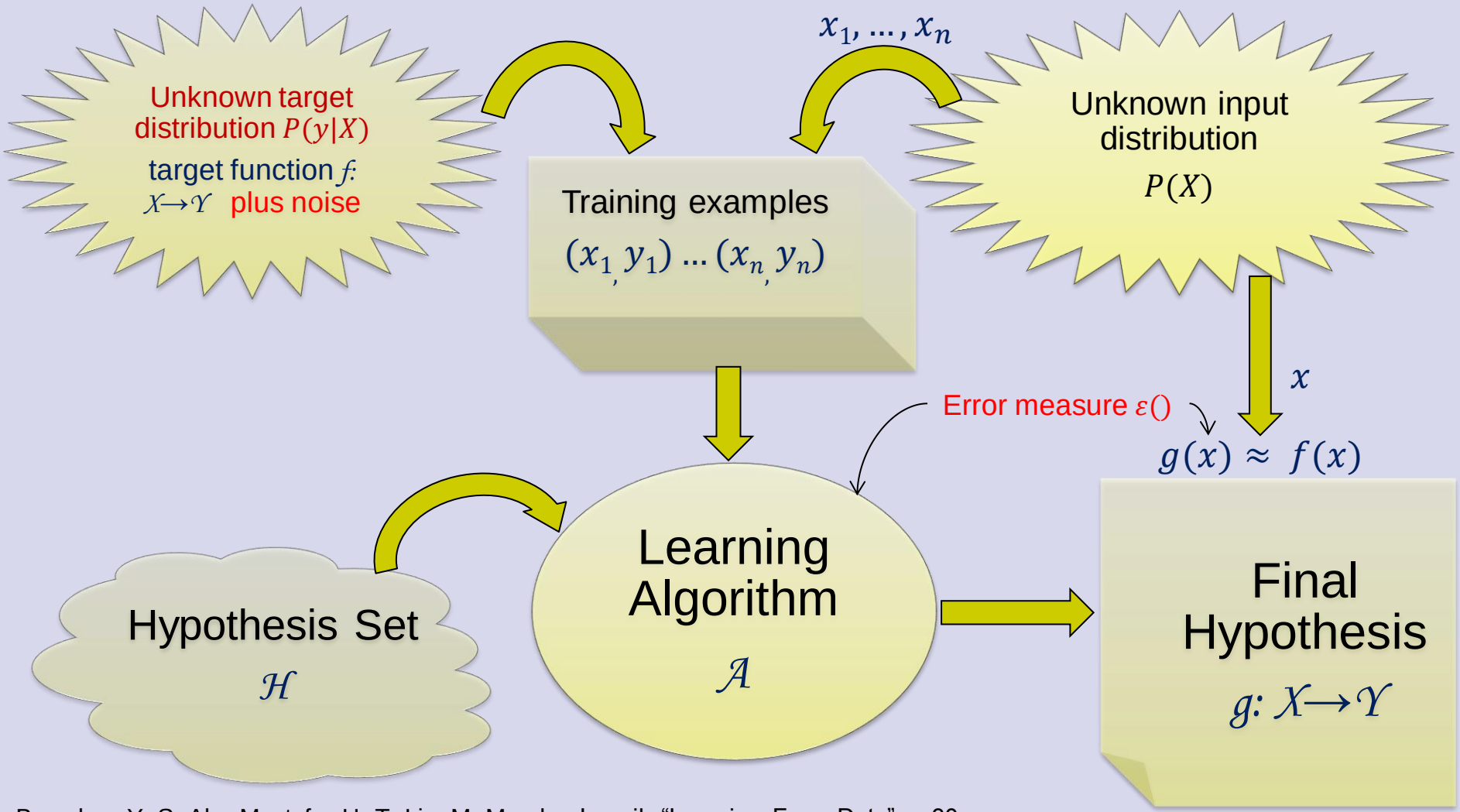
$f(x)$ – unknown target function

$g(x)$ – hypothesis that learning algorithm picks to approximate $f(x)$

$\varepsilon(h, f)$ error measure that quantifies how well each hypothesis $g(x)$ approximates the target function f

$\mathcal{A}$ – the learning algorithm that uses the training set to pick a hypothesis $g$: $X \to Y$ that approximates $f$.

# The learning diagram




Based on Y. S. Abu-Mostafa, H.-T. Lin, M. Magdon-Ismail, "Learning From Data", p.30

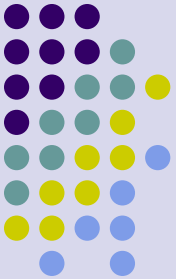
# Generalization

Most learning algorithms fit their models to the training set, but what we really care about is **generalization error,** or expected error on unseen examples**.**

Generalization is ability to perform well on unseen data, is what Machine Learning is ultimately about.

Why should doing well on the training set tell us anything about generalization error? Can we prove that learning algorithms will work well?

# Hoeffding's inequality



It is possible to prove this most important result in learning theory using just two lemmas [2], [3]:

- the union bound
- Hoeffding's inequality (Chernoff bound) which provides an upper bound on the probability that the sum of independent and identically distributed (iid) random variables deviates from its expected value.

$$P(|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| > \gamma) \leq 2ke^{-2\gamma^2 m}$$

This means that for all $h_i \in H$ (*) generalization error $\varepsilon$ will be close to training error $\hat{\varepsilon}$ with "high" probability, assuming $m$ - the number of training examples - is "large".

* for simplicity we assume $H$ is discrete hypothesis set consisting of $k$ hypotheses, but in learning theory this result extrapolates to infinite $H$ [2], [3]

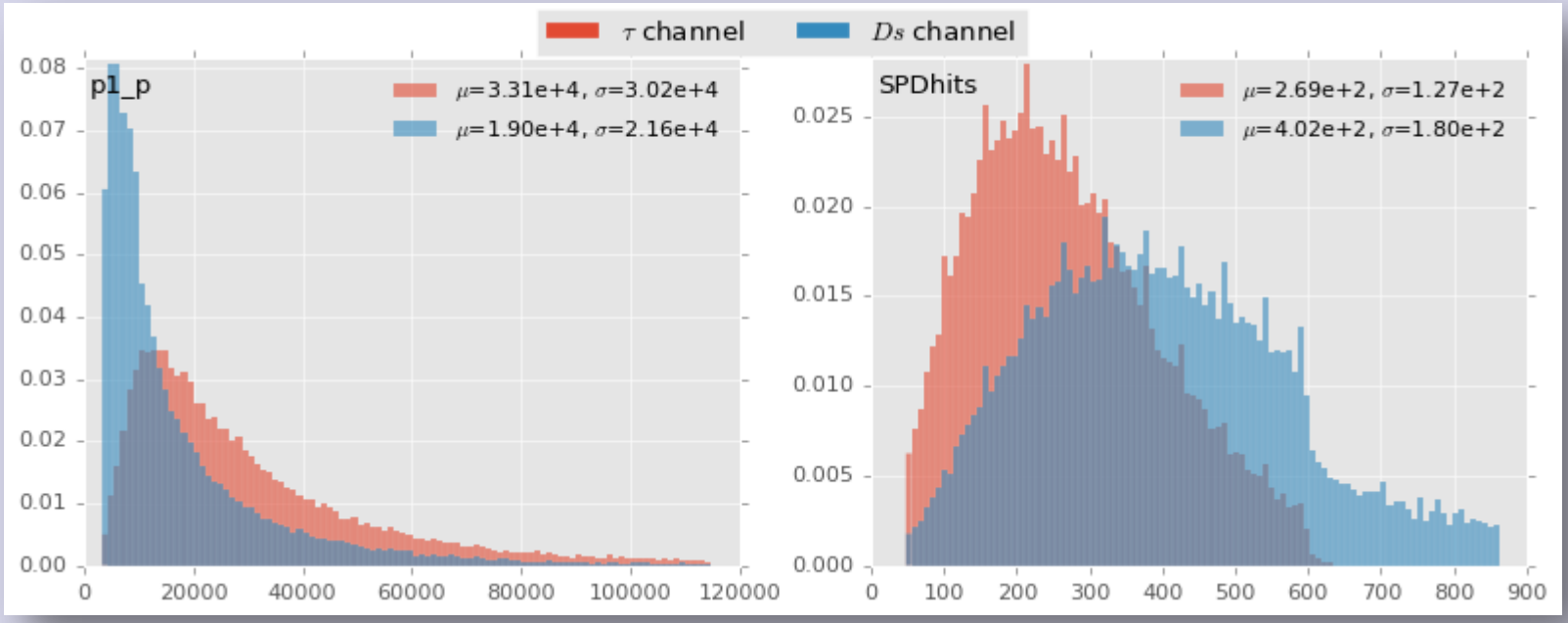
# iid assumption

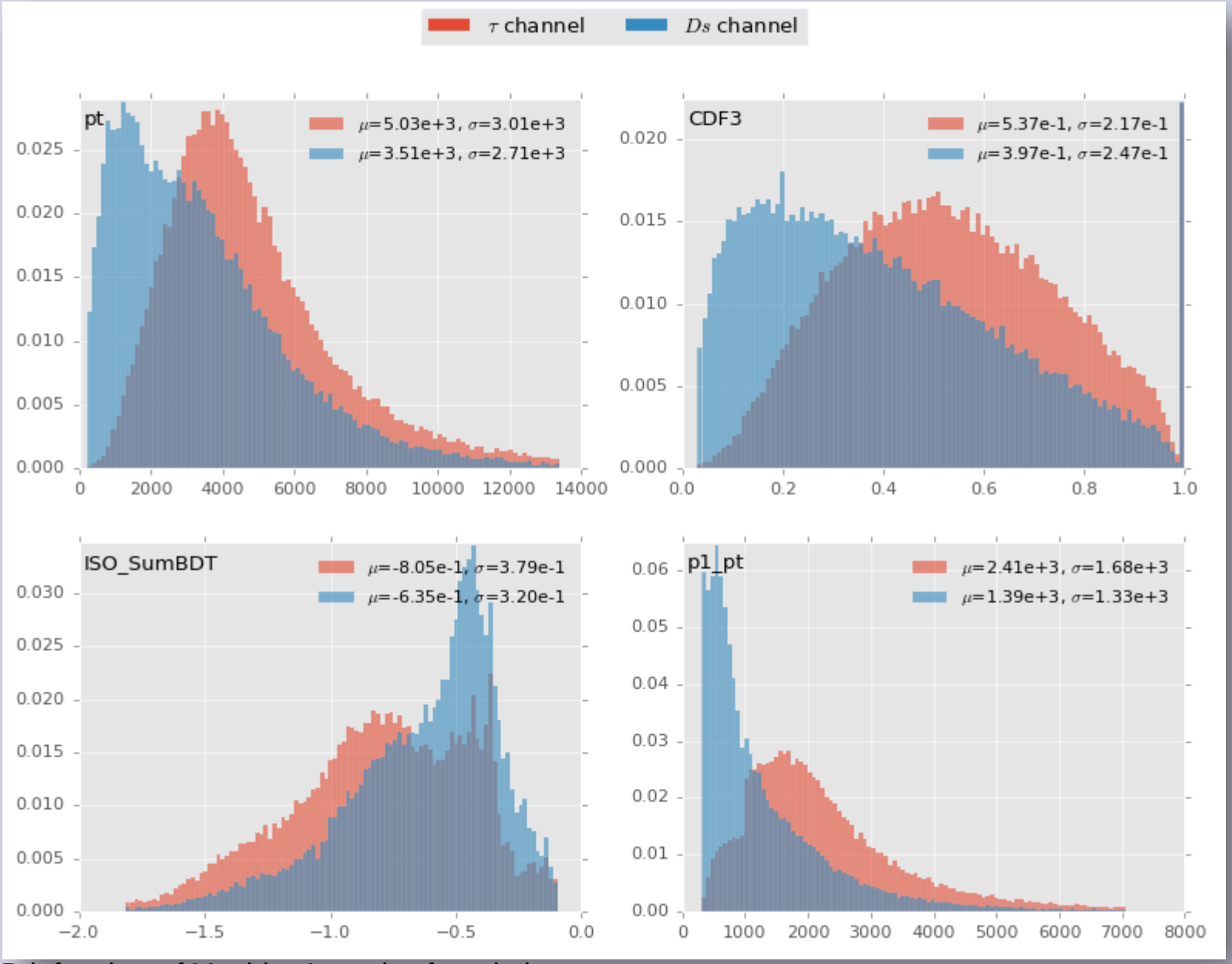
Note that Hoeffding's inequality does not make any prior assumption about the data distribution. Its only assumption is that the training and testing data are independent and identically distributed (iid) random variables drawn from the *same* distribution $\mathcal{D}$

**The assumption of training and testing on the same distribution is the most important in a set of assumptions under which numerous results on learning theory were proved.**

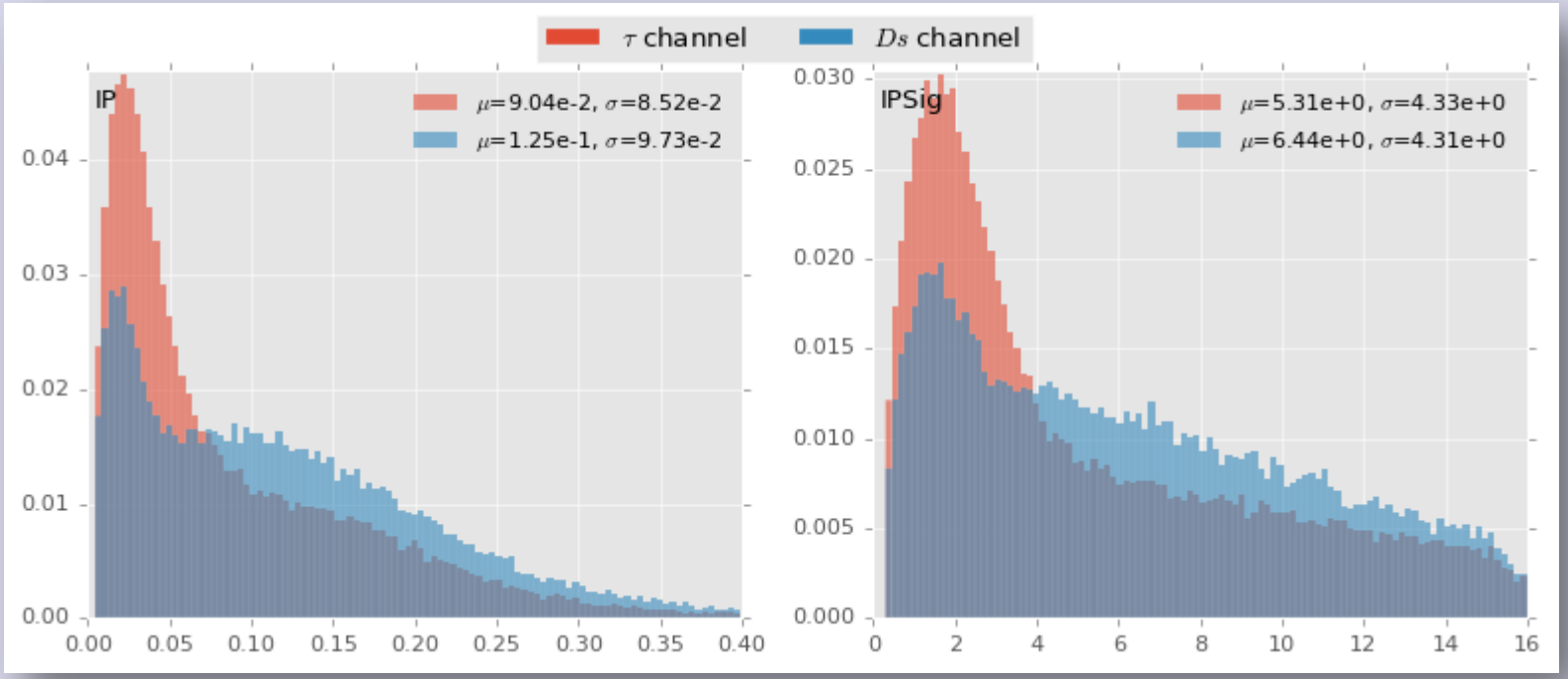
# Comparison of $\tau$ and $Ds$ channel distributuions

Let us return to our problem and look at (some of) input distributions of $\tau$ and $Ds$ channel



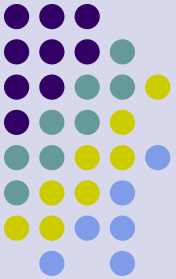

τ channel
Ds channel
pt
μ=5.03e+3, σ=3.01e+3
μ=3.51e+3, σ=2.71e+3
CDF3
μ=5.37e-1, σ=2.17e-1
μ=3.97e-1, σ=2.47e-1
ISO_SumBDT
μ=-8.05e-1, σ=3.79e-1
μ=-6.35e-1, σ=3.20e-1
p1_pt
μ=2.41e+3, σ=1.68e+3
μ=1.39e+3, σ=1.33e+3

# The distributions are different…




It is evident that the underlying probability distributions significantly differ between the two channels, therefore violating the assumption of a single underlying generative process between the train set and the test set, which is the cornerstone of statistical learning and, in general, of inference.

# …but why should we care?



Indeed, we train classifier on $\tau$ channel, but we do not test it (as classifier) on $Ds$ channel. Or we do?

Essentially, when we perform KS-test and require its value to be *small* we make sure that classifier scores follow ***certain*** probability distribution on $Ds$ channel. Such distribution is ***derivative*** of classifier score function just like, for example, area under ROC curve is.

Hence, does KS-value inherit Hoeffding's guarantee obtained for score function during training?

it would… if the distributions were the same

# What does it mean for us?

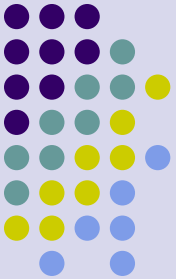
Practically this means that Kolmogorov–Smirnov test provides no guarantee [in statistical sense] for KS value on $Ds$ channel where a model was not trained. At the best, low KS value can be achieved by chance, and there is no guarantee it remains small with some other $Ds$ data.

To draw conclusions from different distribution that did not participate in inference is generally useless - as long as we hold on Machine Learning paradigm.

Before we propose a solution to the problem of different distributions let us review nuances of Kolmogorov–Smirnov test that played interesting role in this competition.

# 3. KOLMOGOROV–SMIRNOV TEST

# Kolmogorov–Smirnov test



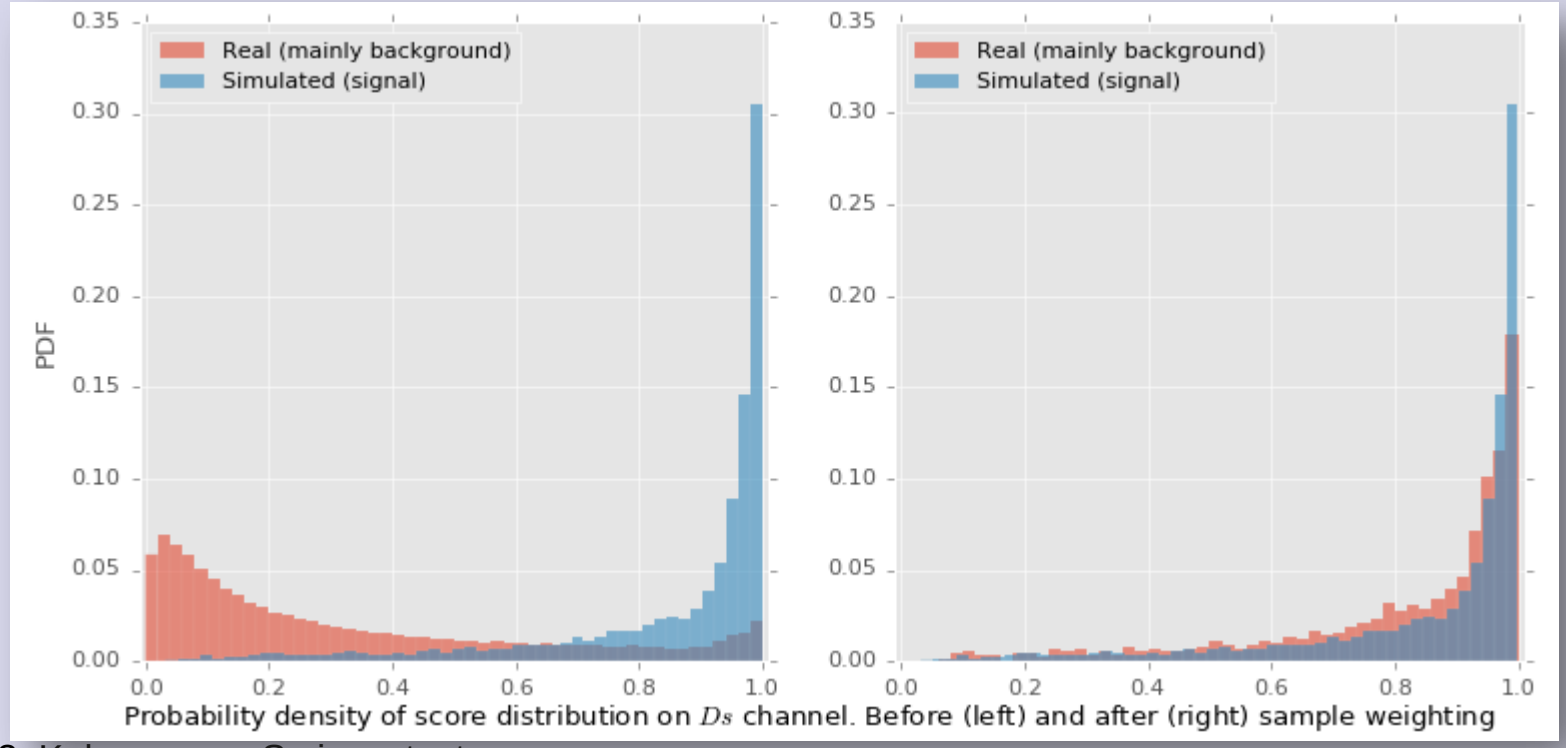
One of the competition requirements for the classifier is small discrepancy when applied to data and to simulation. Kolmogorov–Smirnov test [9, Agreement test] evaluates distance between the empirical distributions of the classifier scores on $Ds$ channel:

$$KS = \sup |F_{MC} - F^*_{data}|$$

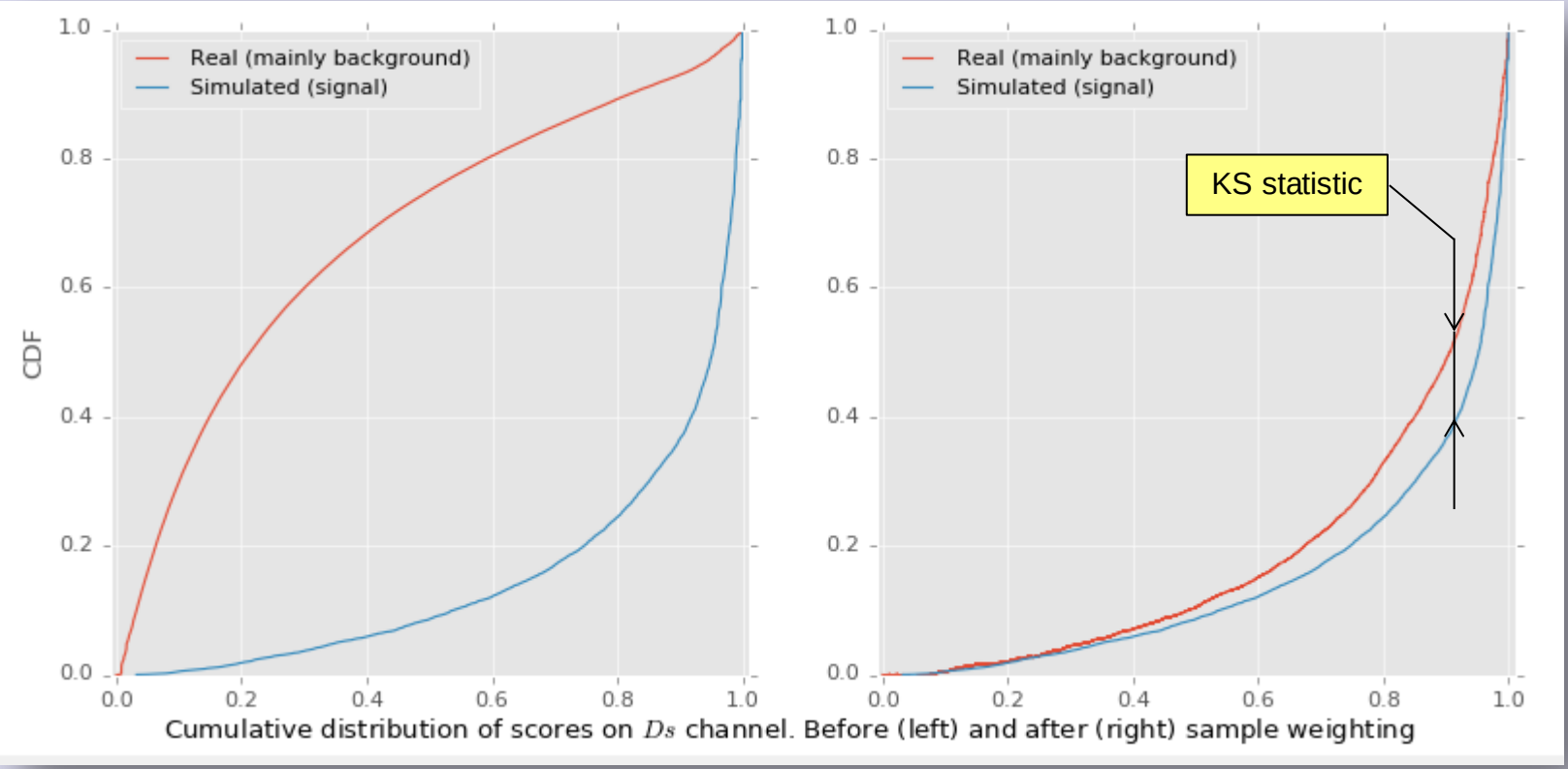
where $F_{MC}$ and $F_{data}$ are cumulative distribution functions (CDF) of the classifier scores $s(X)$ for $X^{Ds}_{data}$ (real, mainly background) and $X^{Ds}_{MC}$ (simulated, signal). Asterisk in $F^*_{data}$ stands for sPlot adjusted distribution $F_{data}$ (see next slide)

# Score distribution agreement on $Ds$ channel

Score densities of real and simulated data on $Ds$ channel are naturally different – see PDFs below. In order to compare them in KS-test, distribution of $s(X_{data}^{Ds})$ is scaled with pre-computed weighs obtained from sPlot method [Agreement test, 10]



# $KS$ statistic

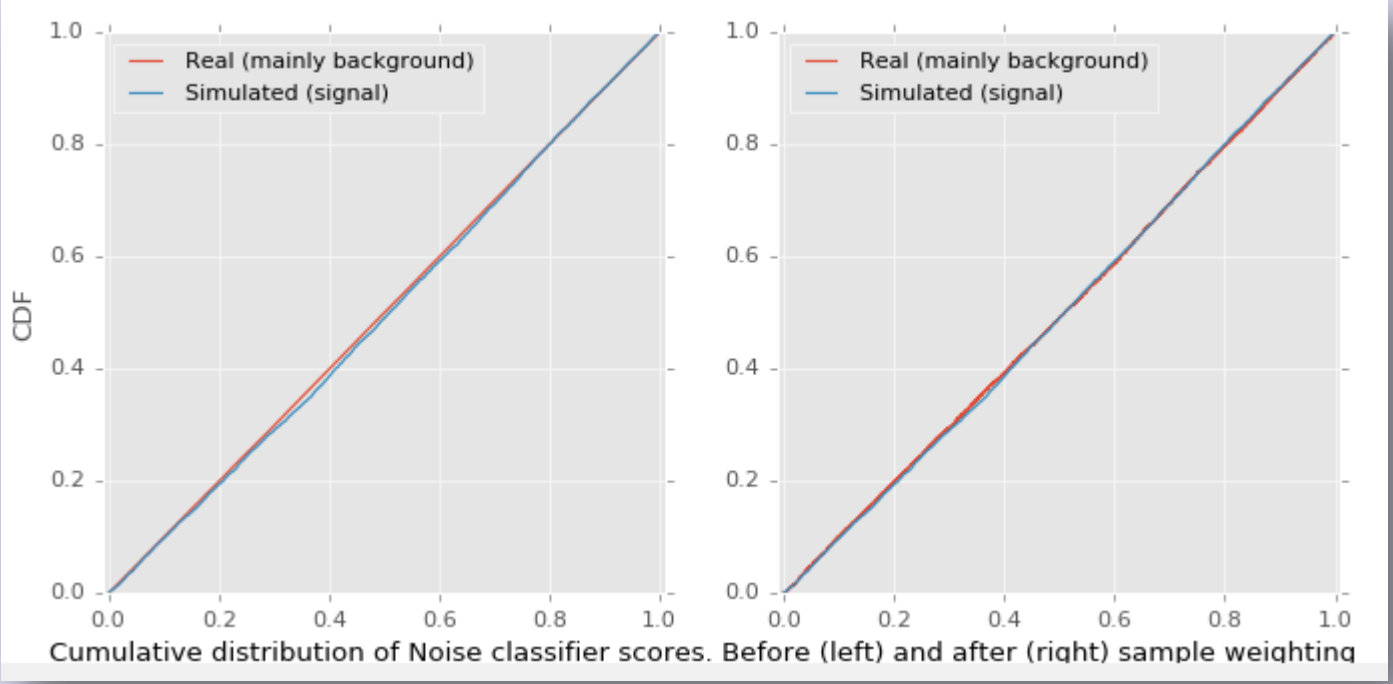


# A degenerate case

There is an amusing degenerate case. If we assign a classifier random score uniformly distributed on $[0, 1]$ and independent of underlying data $X$:

$$P(Y|X) = P(Y) \sim unif\,(0, 1)$$

then CDF $\boldsymbol{F_{data}}$ of such classifier retains its linear shape regardless of scaling:



Cumulative distribution of Noise classifier scores. Before (left) and after (right) sample weighting

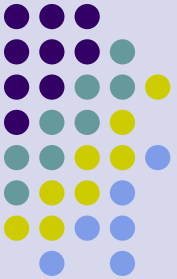
# A degenerate case (continued)

Not affected by sample weighting, such "classifier" can easily pass KS-test on $Ds$ channel. To demonstrate this, author of this presentation has implemented a simple model, combination of "strong" classifier and random noise. This model works as follows:

1. "Real" or "strong" classifier assigns scores to all test samples.
2. Samples with scores below certain (rather high) threshold $T < 1$ change their score to random noise $\sim unif(0, T)$
3. Due to imperfect MC-simulation on $\boldsymbol{\tau}$ channen and different $\boldsymbol{\tau}$ and $\boldsymbol{Ds}$ distributions most $X^{\tau}_{MC}$ samples get high scores $> T$, while most $X^{\tau}_{data}$, $X^{Ds}_{MC}$, $X^{Ds}_{data}$ get random scores $< T$ This randomness is enough for $X^{Ds}_{MC}$, $X^{Ds}_{data}$ samples to satisfy KS test on $\boldsymbol{Ds}$ channel.

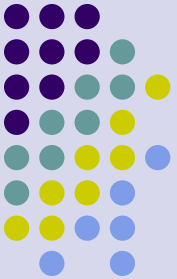
To summarize, different samples get scores from different classifiers. $\boldsymbol{Ds}$ channel is de facto controlled by "random" classifier, $\boldsymbol{\tau}$ channel – by "strong" classifier. Although $X^{Ds}_{MC}$, $X^{Ds}_{data}$ pass KS test on $\boldsymbol{Ds}$ channel, this test has nothing to do with $\boldsymbol{\tau}$ channel and can not assure quality of $X^{\tau}_{MC}$. Please see this discussion for details.

Of course, this is very artificial example, but it shows practical evidence against testing on different distribution.



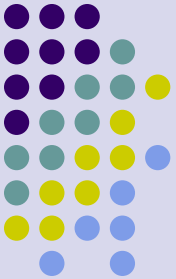
# 4. SOLUTION: TRANSFER LEARNING

# Preamble



The idea of attesting model on control channel is certainly reasonable and can be implemented in theoretically sound way. In Machine Learning the problem of different data sets is well known, and solution is called **Transfer learning.** It aims at transferring knowledge from a model created on the train data set to the test data set, assuming they differ in some aspects, e.g. in distribution.

# Transfer learning applications



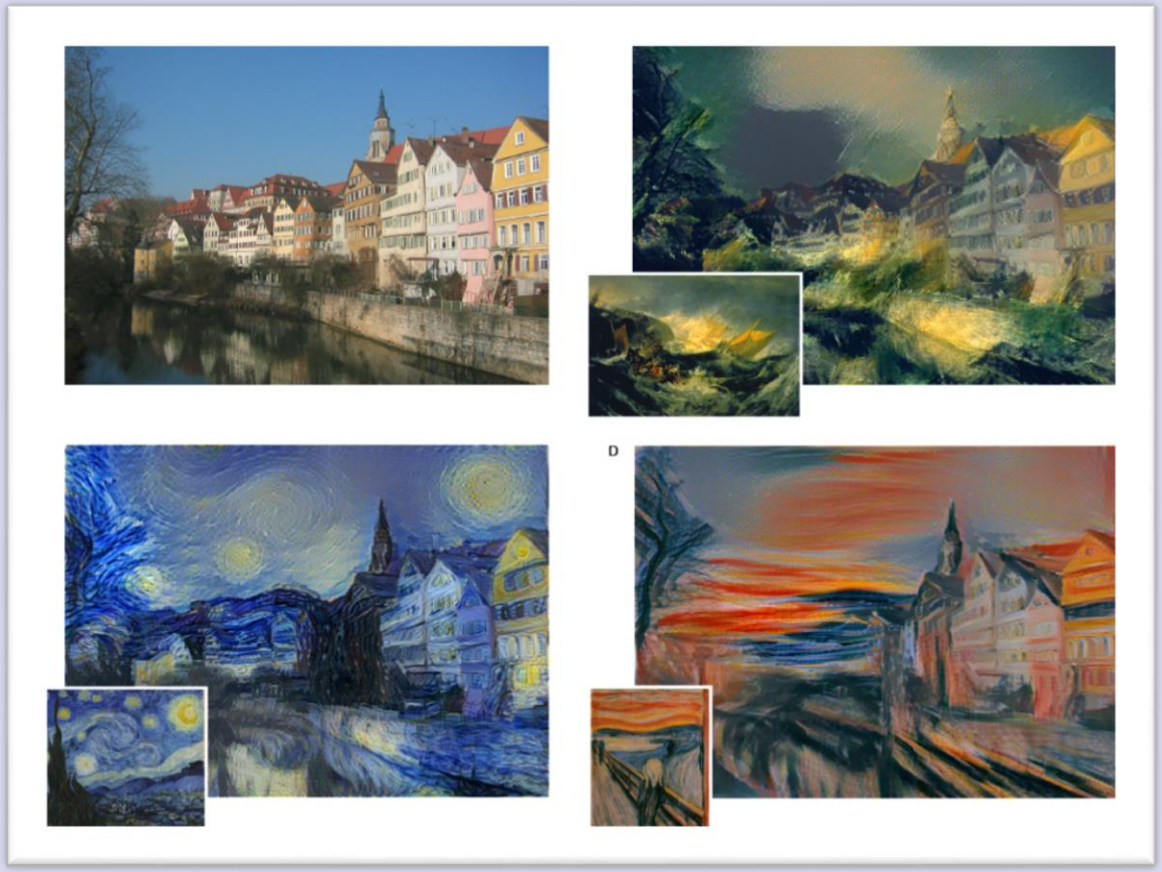
The need for data set adaptation arises in many Machine Learning applications. For example, spam filters can be trained on some public collection of spam, but when applied to an individual mail box may require personalization, i.e. adaptation to specific distribution of emails.

Medicine is another field for transfer learning, as models trained on general population do not fit well individual subjects. The solution is to transfer such models to individuals with adaptation. Brain-computer interface (BCI) is another example of successful application of transfer learning [4]

One common characteristic of these problems is that target domain of interest either has different distribution and little or no labeled data, or is unavailable at training time at all.

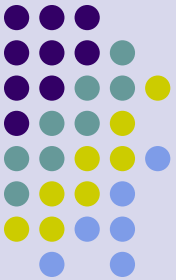
# Example

Often domain adaptation is hidden in disguise of unsupervised features learning. It is common practice to apply convolutional filters acquired from some image- or audio data set to another data set, sometimes from different domain.



Example of artistic style transfer. From “A Neural Algorithm of Artistic Style”, link

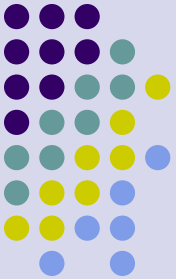
# Setting our goals



Our goal is formulated as follows: we have data from two channels of similar topology and different distribution. The first channel we consider “reliable” (well-studied behavior, reliable data), we use it as *Source* for transfer learning. The second – “experimental” (less reliable data), we call it *Target*. We train logistic regression on *Source* channel and transfer it to *Target* channel. In addition, we may require our classifier to satisfy some other criteria, like KS and CvM tests.

*Only physicists can decide* which of the two channels – $\tau$ and $Ds$ – is the *Source*, and which is the *Target*. In this section we depart from original competition defaults and assume the reverse order: $Ds$ is the *Source* (as more reliable data) and $\tau$ is the *Target*. It looks more natural to us, but we can be wrong in this assumption. “Implementation” section follows the original order set by the competition design, just like our real solution.

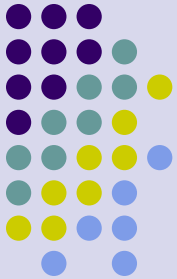
# Formal definitions



Let us introduce the notation of transfer learning between channels with focus on our problem of discriminating signal and background events.

- let $X \in \mathcal{X}$ be a data sample, and $(X, y)$ an event, where $\mathcal{X}$ represents feature space and $y$ represents category of event $\{0, 1\}$ (background vs. signal)
- let $P(X)$ be the marginal probability distribution of $X$
- following [5] we call $\mathcal{D} = \{X, P(X)\}$ a domain, which in our case is a channel from which we received data sample.
- we have data for source domain $\mathcal{D}_{Ds}$ and for target domain $\mathcal{D}_{\tau}$
- let $f: \mathcal{X} \to \mathcal{Y}$ be the predictive target function. A task is defined as a predictive function and its output space $\mathcal{T} = \{\mathcal{Y}, f\}$

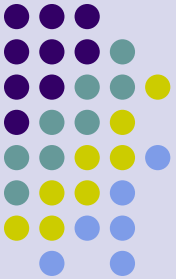
# Transductive Transfer Learning



As stated in [5], “transfer learning aims to help improve the learning of the target predictive function $f_T$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$ where, in general, $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S \neq \mathcal{T}_T$”.

Our problem has two different but related domains $\mathcal{D}_{Ds}$ and $\mathcal{D}_\tau$, and the tasks are the same in both domains: discriminating signal from background events, $\mathcal{T}_{Ds} = \mathcal{T}_\tau = \mathcal{T}$. Moreover, the source domain $\mathcal{D}_{Ds}$ has labeled data set, and the target domain has unlabeled data $\mathcal{D}_\tau$ (notice the difference with original competition formulation). This specific setting of the transfer learning problem is called *transductive transfer learning* (TTL) [4]

# Covariate Shift

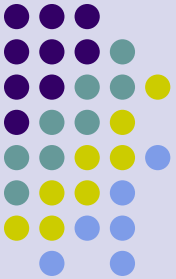


We have shown in "Review of Machine Learning" section that the marginal probability distributions of source and target differ: $P(X_{Ds}) \neq P(X_{\tau})$ The TTL approach pertinent to different distributions goes in literature under the name *covariate shift* [7, 8], *domain adaptation* or *sample selection bias.* Its assumption is that $P_{Source}(Y|X) = P_{Target}(Y|X)$. That is, given the same observation, the conditional distributions of $Y$ are the same in the two domains.

We have also shown that in the empirical risk minimization framework learning is the process of minimizing the loss function over training data:

$$f^* = \underset{f \in F}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} l(f, x_i, y_i)$$

# TTL framework



If the train dataset is drawn from $P_{Ds}(X,\ Y)$ but we are interested in predictions when the test data come from $P_{\tau}(X,\ Y)$, then each term can be penalized according to how likely each trial belongs to the target domain $\mathcal{D}_{\tau}$:

$$f^{*} = \underset{f \in F}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \frac{P_{\tau}(x_i, y_i)}{P_{Ds}(x_i, y_i)} l(f, x_i, y_i)$$

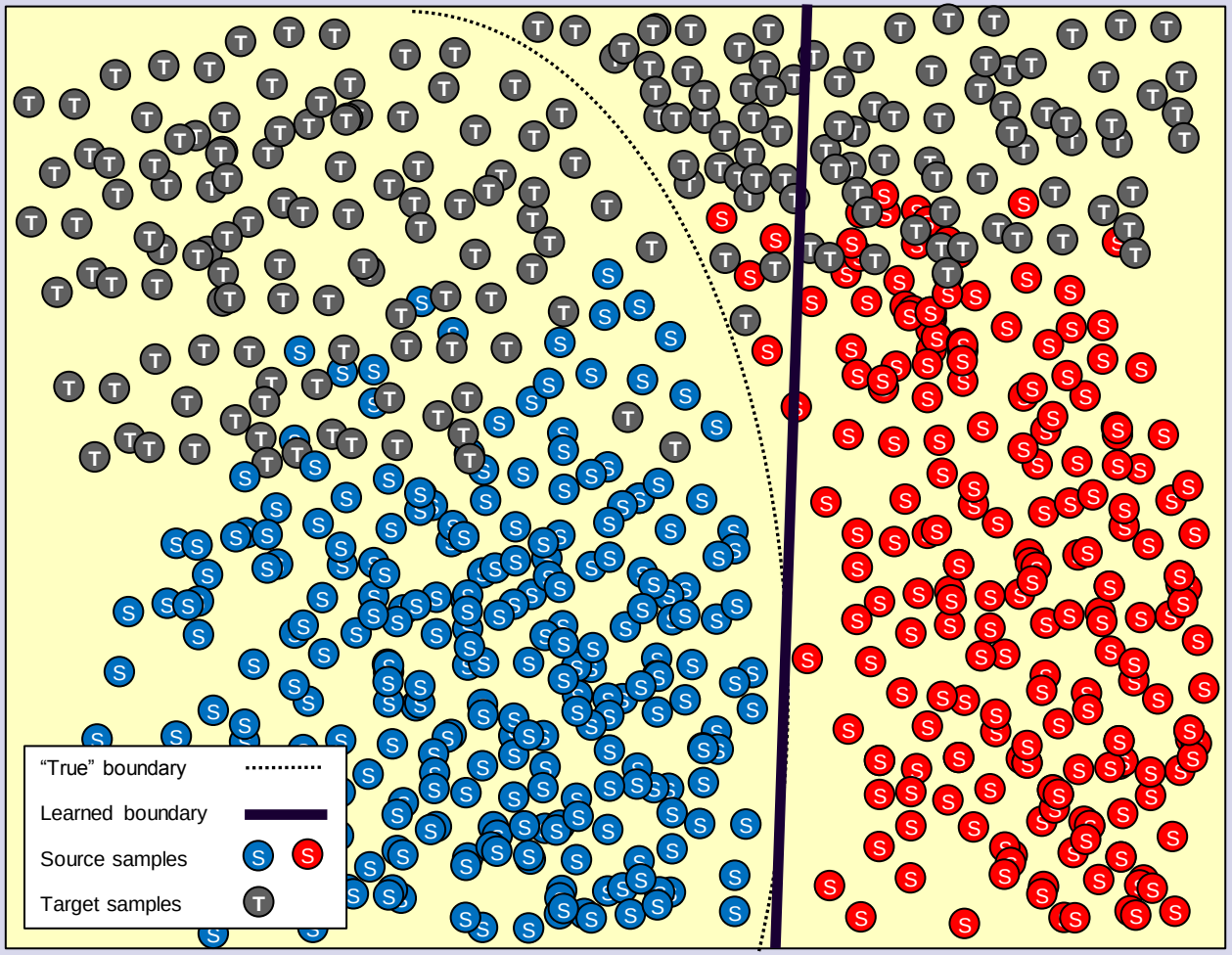
The covariate shift assumption is that $P_{Ds}(Y|X) = P_{\tau}(Y|X)$, applying Bayes' rule $\frac{P_{\tau}(x_i,y_i)}{P_{Ds}(x_i,y_i)} = \frac{P_{\tau}(x_i)}{P_{Ds}(x_i)}$. Then, the risk minimization problem becomes:

$$f^{*} = \underset{f \in F}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \frac{P_{\tau}(x_i)}{P_{Ds}(x_i)} l(f, x_i, y_i)$$

# Covariate shift: Toy example

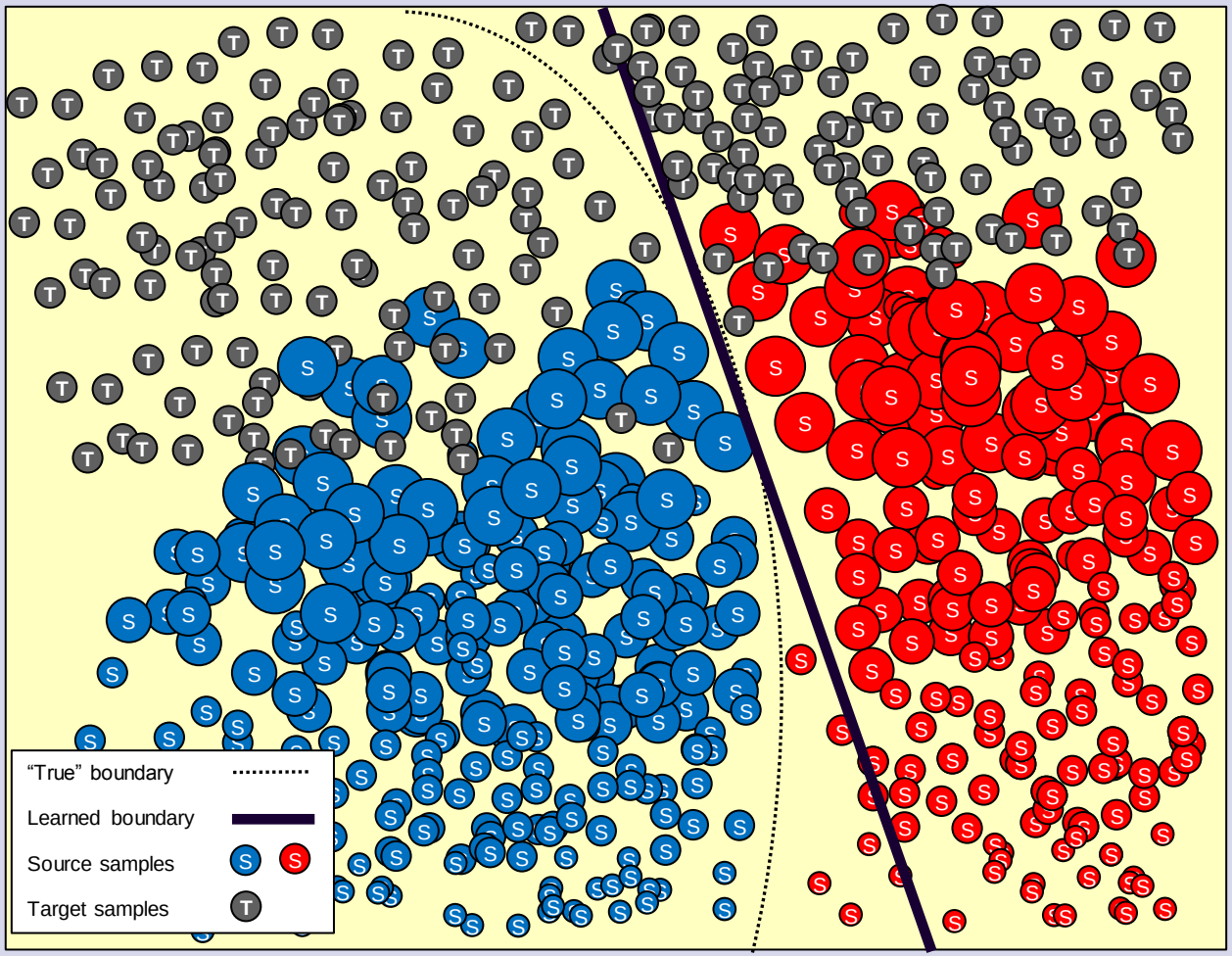
Target samples left unlabeled intentionally, but we assume those on the left of the “True” boundary are blue, and those on the right are red.

Only source samples participate in training.



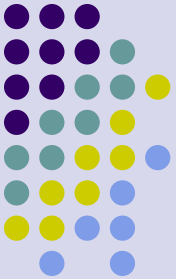

Linear separation boundary for Source without covariate shift

# Toy example (continued)




Linear separation boundary under covariate shift

# Constraining the model on KS and CvM values



Finally, because we are limited in KS and CvM values we have to constrain our solution. To this end we incorporate KS and CvM into TTL framework:

$$f^* = \underset{f \in F}{\operatorname{argmin}} \left( \frac{1}{n} \sum_{i=1}^{n} \frac{P_\tau(x_i)}{P_{Ds}(x_i)} l(f, x_i, y_i) + \rho(f, ks, cvm) \right)$$

where $\rho(f, ks, cvm)$ - regularization term

In order to estimate $\frac{P_\tau(x_i)}{P_{Ds}(x_i)}$ we may, for example, set up a new logistic regression model to discriminate trials belonging to the $\tau$ channel from those of the $Ds$ channel, which requires just unlabeled data [4].

# 5. IMPLEMENTATION

# Implementation

For the competition we have submitted several different models. The transfer learning model achieved AUC = 0.996802 on Public Leader Board. It was improved to 0.998150 post competition. Source code can be downloaded from GitHub page https://github.com/alexander-rakhlin/flavours-of-physics

As we remember, training on $Ds$ channel was prohibited by the competition requirements, what dictated us certain amendments to the Transductive Transfer Learning framework outlined in Section 4.

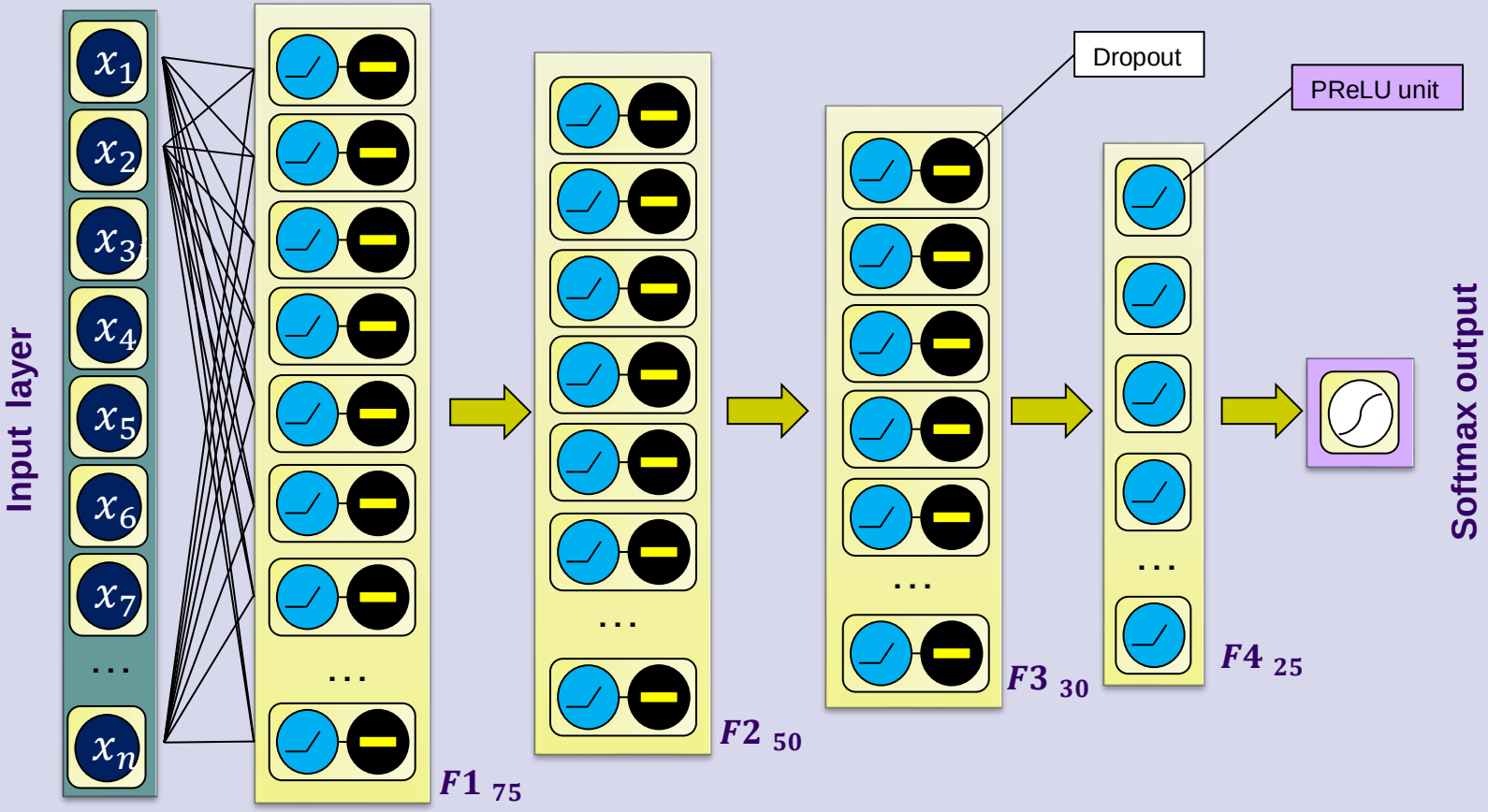
In this implementation we transfer model from $\tau$ channel to $Ds$ channel. The other difference is how domain adaptation is implemented: instead of covariate shift we do calibration.

# Initial training

We implemented two-stage process: 1) Initial training and 2) Adaptation stage.

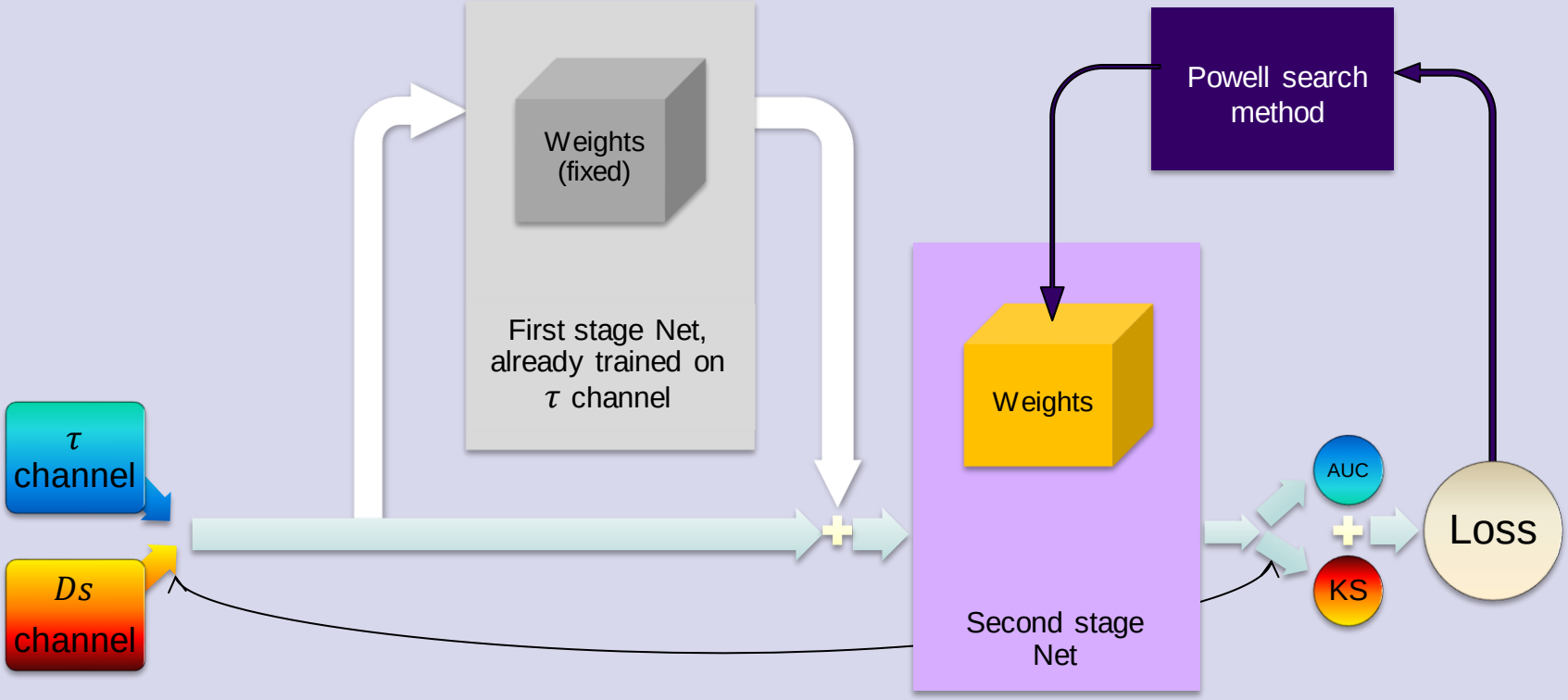
On the Initial stage the model is trained for maximum performance on labeled data from $\tau$ channel. This model is based on ensemble of 20 feed forward fully connected neural nets written in Python with Keras library. Each net comprises 4 fully connected hidden layers F1-F4 with PReLU activations fed to a 2-way softmax output layer which produces a distribution over the 2 class labels. The hidden layers contain 75, 50, 30, and 25 neurons respectively. To avoid overfitting, dropout is used in F1, F2 and F3 layers:

# Neural Net architecture

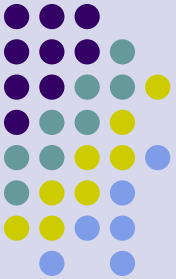


# Adaptation scheme

On the Adaptation stage first Net is fixed, its output is stacked with original features and cascaded to the second net of similar configuration:



# Training the 2-nd Net

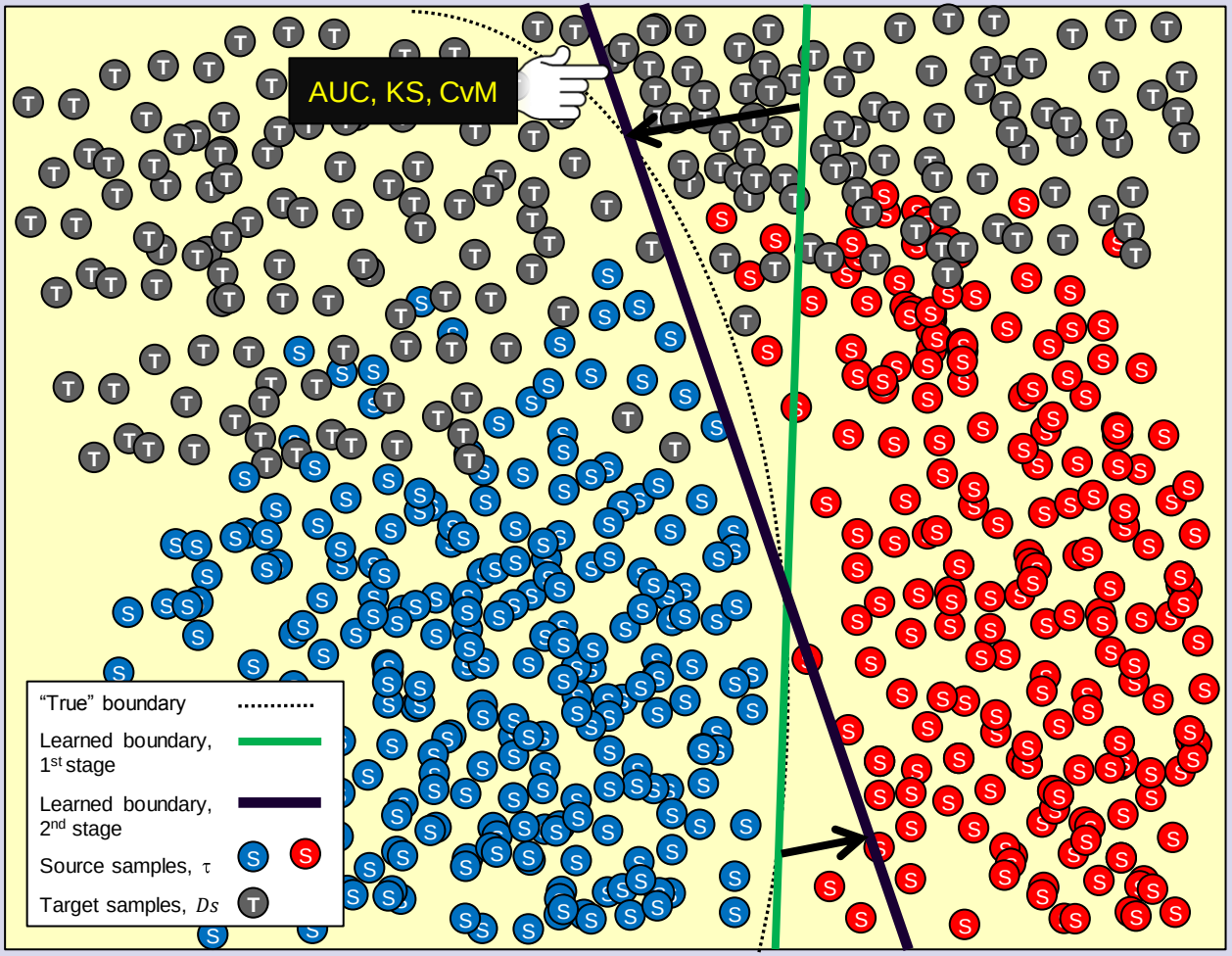


$Ds$ channel was labeled for other purpose than training, and training on $Ds$ was prohibited. But calculation of KS and CvM values - allowed. The other metric to satisfy is AUC on $\tau$ channel. All 3 metrics are global and do not suit for stochastic gradient descent (SGD). Therefore we train this second Net in a special way (see code):

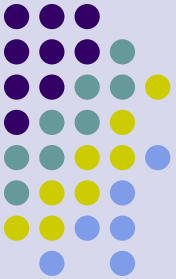
1. We set initial weights of the second Net to reproduce output of the first Net. This is accomplished after 1-3 epochs of SGD with cross entropy loss on $\tau$ samples.
2. Then adaptation of weights is done with help of Powell's search method in `minimize()` function from `scipy.optimize` library. On this step the Net is fed with $\tau$ and $Ds$ samples and optimized directly for AUC, KS and CvM metrics.

# Adaptation (toy illustration)

The purpose of the second Net is to adopt the model to $Ds$ channel with minimal and controlled loss of performance



# Training the 2-nd Net (code)

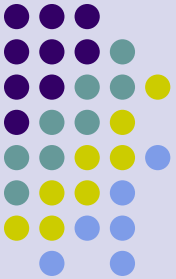


transductor_train.py

```
from scipy.optimize import minimize
…
pretrain = True
if pretrain:
    # pretrain model
…
    model.fit(Xt, yt_categorical, batch_size=64, nb_epoch=1,
              validation_data=None, verbose=2, show_accuracy=True)
…

x0 = get_weights(model)
print("Optimize %d weights" % len(x0))
objective = create_objective(model, pt.transductor_model_file,
                             Xt, yt, Xa, ya, wa, Xc, mc, verbose=True)
minimize(objective, x0, args=(), method='Powell')
```

# Loss function



On the second stage loss function incorporates AUC, KS, CvM metrics:

transductor_train.py

```
loss = -auc + ks_importance * ks_loss + cvm_importance * cvm_loss
```

As a result, we control all three values simultaneously. But the most important thing is that this approach provides statistical guarantee that the model satisfies all target metrics (AUC, KS, CvM) not by coincidence but as a result of statistical inference in their respective domains – in accordance with Machine Learning principles outlined in Section 2.

# Closing remarks and suggestions

The model appears more complex than it should be! Striving to satisfy some competition requirements we overdone fairly simple Transfer Learning Framework. We encourage the Organizers to lift “no training on control channel” requirement and try to create a model the other way round: on $Ds$ channel and transfer it to $\tau$ channel…. If this has physical meaning, of course

In general, we suggest to consider transfer learning as a method for modeling rare events based on models built in domains with more known and well-observed behavior.

Some other ML approaches to consider for rare events detection [11]:

- One-class classification (a.k.a. outlier detection, anomaly detection, novelty detection)
- PU learning

# References

[1] Blake T., Bettler M.-O., Chrzaszcz M., Dettori F., Ustyuzhanin A., Likhomanenko T., “Flavours of Physics: the machine learning challenge for the search of $\tau^- \to \mu^- \mu^- \mu^+$ decays at LHCb” link

Machine Learning:

[2] Y. S. Abu-Mostafa, H.-T. Lin, M. Magdon-Ismail, “Learning From Data”. AMLBook (March 27, 2012), ISBN-10: 1600490069

[3] Andrew Ng, “Part VI Learning Theory”. Stanford University CS229 Lecture notes link

Transfer learning:

[4] Olivetti, E., Kia, S.M., Avesani, P.: “Meg decoding across subjects.” In: Pattern Recognition in Neuroimaging, 2014 International Workshop on (2014) link

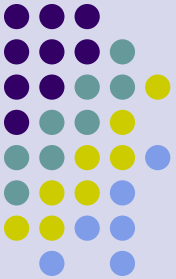
[5] S. J. Pan and Q. Yang, “A Survey on Transfer Learning,” Knowledge and Data Engineering, IEEE Transactions on, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[6] B. Zadrozny, “Learning and Evaluating Classifiers under Sample Selection Bias” In Proceedings of the Twenty-first International Conference on Machine Learning, ser. ICML ’04. New York, NY, USA: ACM, 2004, pp. 114+.

[7] Jiang J., “A Literature Survey on Domain Adaptation of Statistical Classifiers” link

[8] Adel T., Wong A., “A Probabilistic Covariate Shift Assumption for Domain Adaptation”. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence link

# References (continued)



Kolmogorov-Smirnov test:

[9] Wikipedia, “Kolmogorov–Smirnov test” link

[10] Andrew P. Bradley, “ROC Curve Equivalence using the Kolmogorov-Smirnov Test”. The University of Queensland, School of Information Technology and Electrical Engineering, St Lucia, QLD 4072, Australia link

Other methods:

[11] Wikipedia, “One-class classification” link

# Acknowledgements

This work is dedicated in gratitude to Yaser S. Abu-Mostafa, Professor of Electrical Engineering and Computer Science at the California Institute of Technology. To great mentor, who taught me foundations of Machine Learning.

*Claude E. Shannon with a young Yaser Abu-Mostafa*

I would like to thank Tatiana Likhomanenko for her valuable comments that helped to improve the quality of this presentation. I want to thank respected organizers of Kaggle's "Flavours of Physics: Finding $\tau^- \to \mu^- \mu^- \mu^+$ " for very interesting and challenging competition, and for the recognition they awarded my work. It means a lot.

## Thank you for your attention!