Transfer Learning: Using the Data You Have, not the Data You Want.

October, 2013

Brian d'Alessandro

@delbrians

Motivation

Personalized Spam Filter

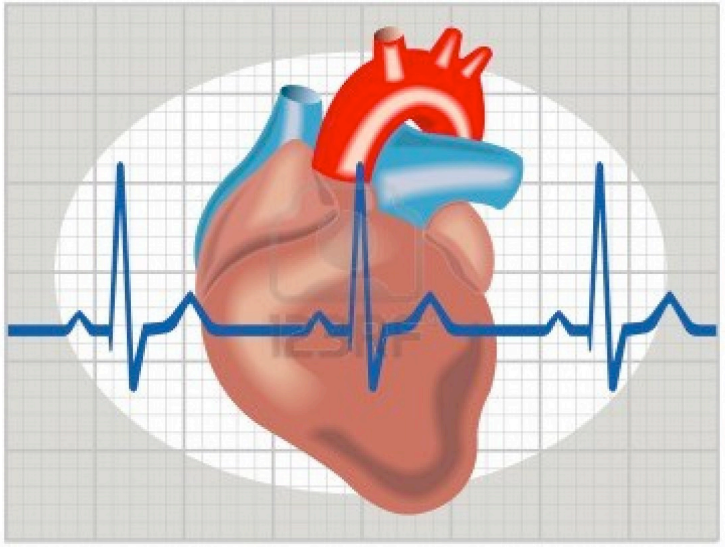
Goal:
You want to build a personalized SPAM filter

Problem:
You have many features but relatively few examples per user. And for new users, you have no examples!

Disease Prediction

Goal:
Predict whether a patients EKG reading exhibit Cardiac Arrhythmia.

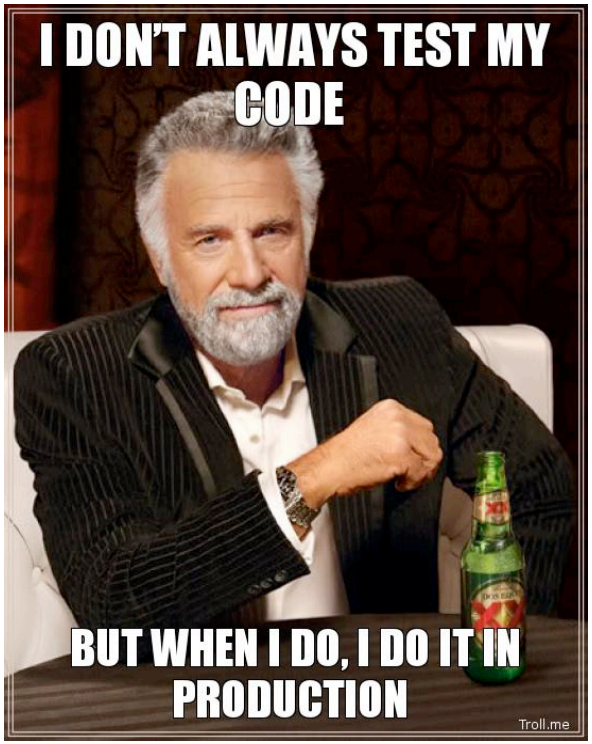
Problem:
CAD is generally rare and labeling is expensive. Building a training set that is large enough for a single patient is invasive and expensive.



Ad Targeting

Goal:
Predict whether a person will convert after seeing an ad.

Problem:
Training data costs a lot of money to acquire, conversion events are rare, and feature sets are large.



Common Denominator

Goal:
Classification on one or multiple tasks.

Problem:
Little to no training data but access to other potentially relevant data sources.

Definition, Notation, Technical Jargon etc.

Some notation…

Target Data (the data you need):

$$Domain^T = \{X^T, P^T(X)\} \qquad Task^T = \{Y^T, f^T(.)\}$$

Source Data (the data you have):

$$Domain^S = \{X^S, P^S(X)\} \qquad Task^S = \{Y^S, f^S(.)\}$$

A *Domain* is a set of features and their corresponding distribution.
A *Task* is a dependent variable and the model that maps X into Y.

Definition

Given source and target domains, and source and target tasks, transfer learning aims to improve the performance of the target predictive function $f^T(.)$ *using the knowledge from the source domain and task, where* $Task^S \neq Task^T$ *and* $Domain^S \neq Domain^T$.

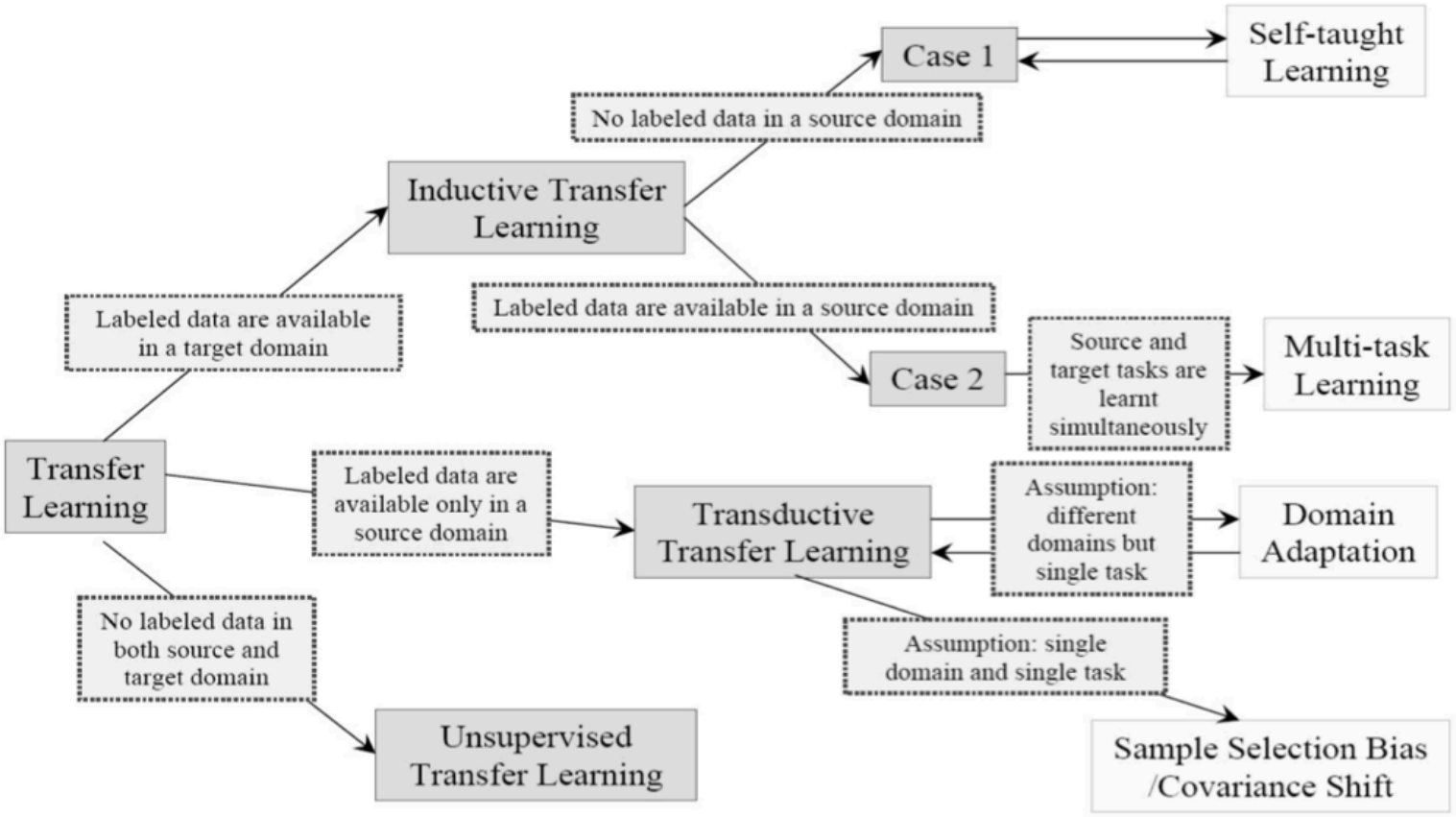
Source:
Title: "A Survey on Transfer Learning"
Authors: Pan, Yang
Published: IEEE Transactions on Knowledge and Data Engineering, October 2010

In other words…

If you don't have enough of the data you need, don't give up:
Use something else to build a model!

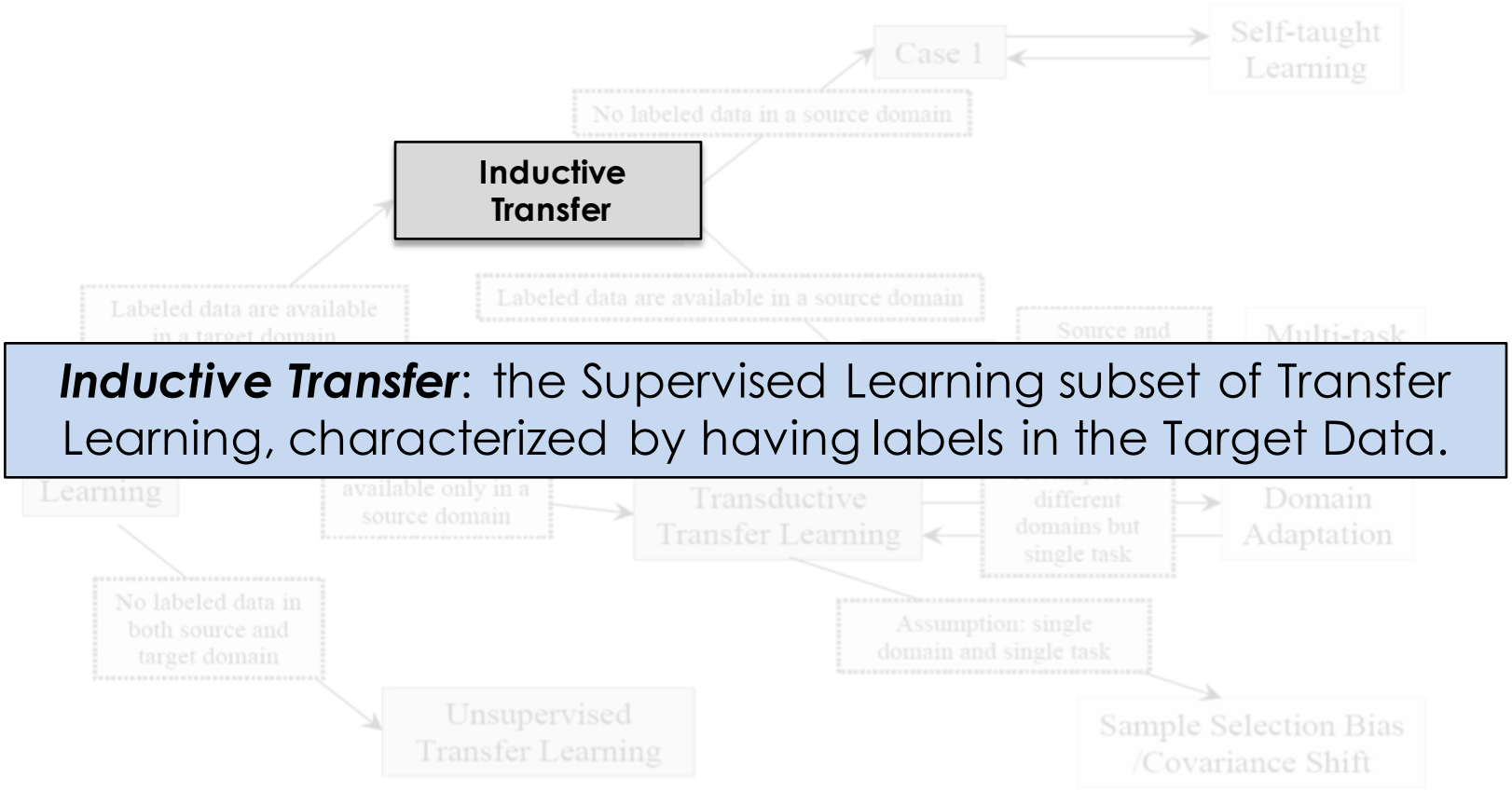
Source ***Target***

There's more…



***Source**: "A Survey on Transfer Learning"*

Alright, focus…



Inductive Transfer: the Supervised Learning subset of Transfer Learning, characterized by having labels in the Target Data.

Source: *"A Survey on Transfer Learning"*

How does this change things?

Classic vs. Inductive Transfer Learning

Inductive Transfer Learning follows the same train/test paradigm, only we dispense of the assumption that train/test are drawn from the same distribution.

Classic

Train/Test: "same" data

Training Layer

ID	Y	X_1	X_2	X_3	...	X_k
5	1	1	0	0	...	1
1	0	1	0	1	...	1
3	1	1	0	1	...	0
...	...	...	...	...	...	...
N_train	0	1	1	1	...	1

Testing Layer

ID	Y	X_1	X_2	X_3	...	X_k
2	1	0	0	1	...	0
4	0	0	1	1	...	0
6	1	0	0	0	...	1
...	...	...	...	...	...	...
N_test	0	0	1	0	...	1

Transfer

Train/Test: different data

ID	Y	X_1	X_2	X_3	...	X_k
1	0	0	0	1	...	0
2	1	0	1	1	...	0
3	1	0	1	1	...	0
...	...	...	...	...	...	...
N_source	1	0	0	1	...	0

ID	Y	X_1	X_2	X_3	...	X_k
1	0	1	0	0	...	1
2	1	0	0	0	...	0
3	0	1	0	0	...	1
...	...	...	...	...	...	...
N_target	0	1	1	1	...	0

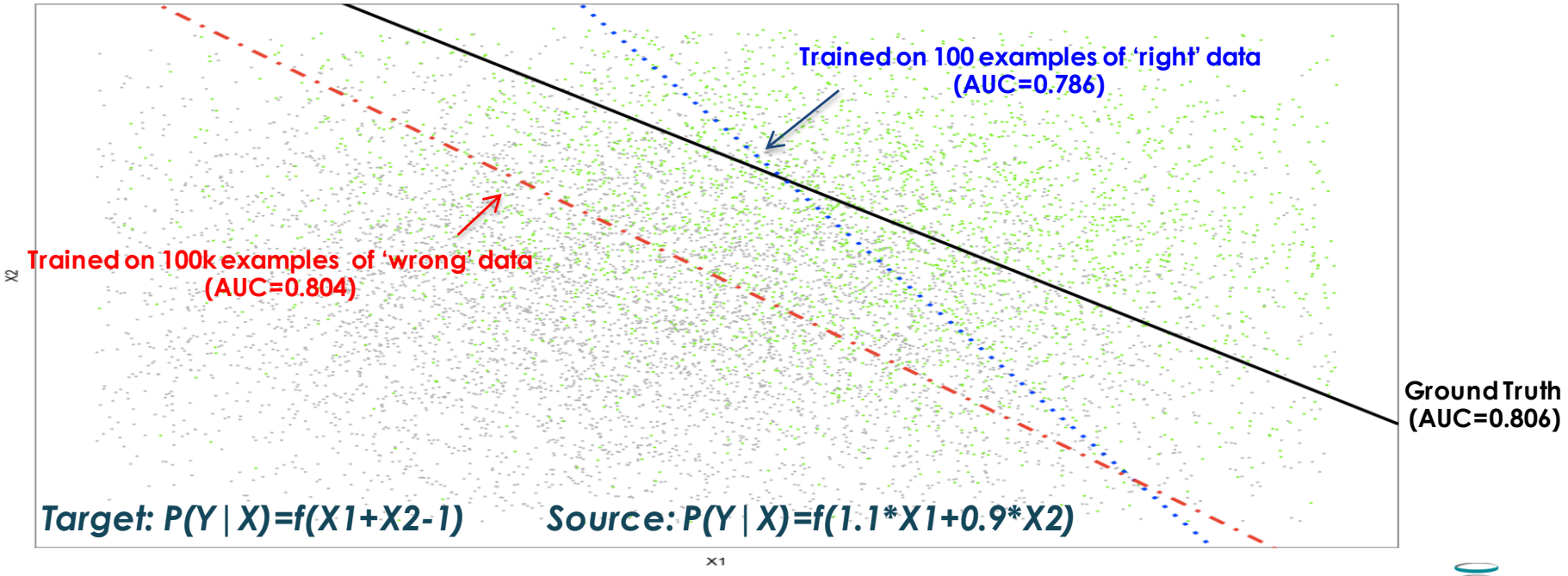
@delbrians

dstillery

Why does this Work?

The Bias-Variance Tradeoff

A biased model trained on a lot of the 'wrong' data is often better than a high variance model trained on the 'right' data.



dstillery

Transfer Learning in Action

TL in Action

Multi-Task Learning for SPAM detection

Source:
Title: "Feature Hashing for Large Scale Multitask Learning"
Authors: Weinberger, Dasgupta, Langford, Smola, Attenberg @ Yahoo!Research
Published: Proceedings of the 26th International Conference on Machine Learning, Montreal, Canada, 2009.

Multi-task Learning

Task	Y	X_1	X_2	X_3	…	X_k
1	1	0.6	0.3	0.3	…	0.3
1	0	0.5	0.6	0.4	…	0.9
1	0	0.0	0.2	0.2	…	0.8
1	1	0.8	0.6	0.5	…	0.4
2	0	0.9	0.1	0.7	…	0.4
2	0	0.8	0.1	0.9	…	0.6
2	0	0.7	0.3	0.3	…	0.4
2	1	0.9	0.7	0.6	…	0.7
3	1	0.8	0.2	0.6	…	0.9
3	1	0.0	0.6	0.4	…	0.5
3	0	0.5	0.2	0.8	…	0.2
3	0	0.5	0.3	0.3	…	0.8
…	…	…	…	…	…	…
T	0	0.0	0.4	0.9	…	0.6
T	0	0.6	0.2	0.2	…	0.7
T	0	0.4	0.3	0.4	…	0.7
T	1	0.6	0.4	0.6	…	0.8

Multitask learning involves joint optimization over several related tasks, having the same or similar features.

Intuition: Let individual tasks borrow information from each other

Common Approaches:

- Learn joint and task-specific features
- Hierarchical Bayesian methods
- Learn a joint subspace over model parameters

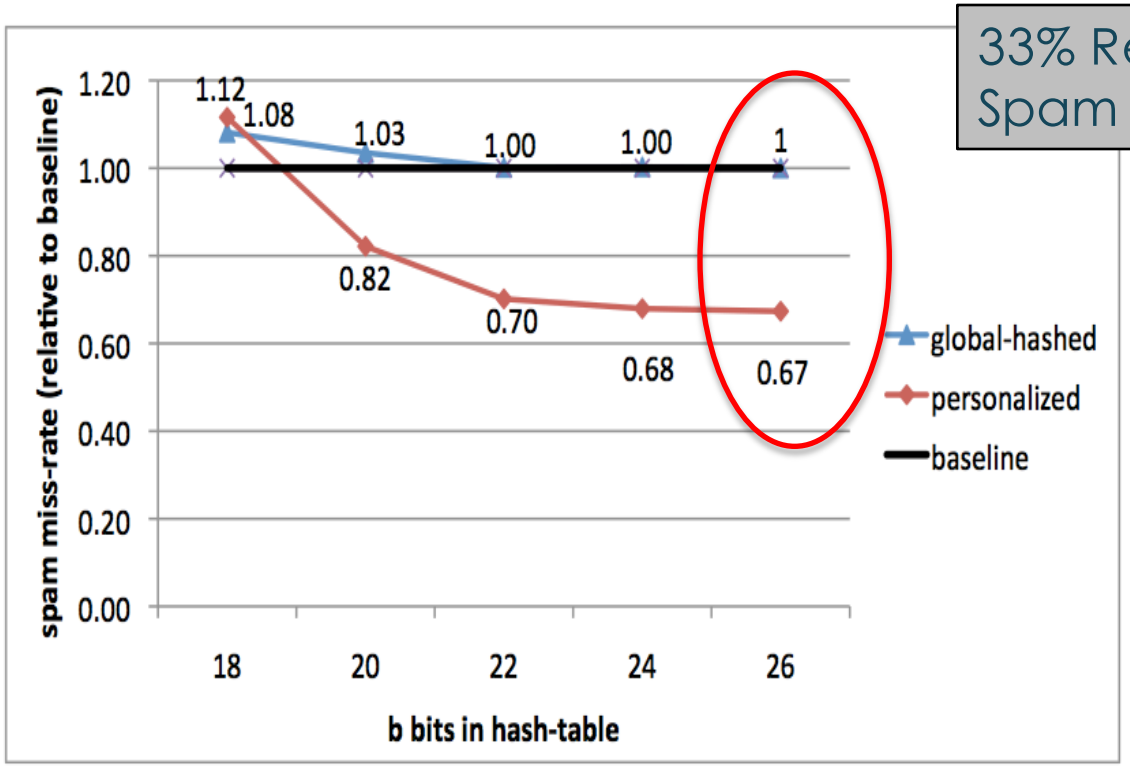
SPAM Detection - Method

Learn a user level SPAM predictor.
Predict if email is 'SPAM/Not SPAM'

Methodology:

1. Pool users
2. Transform <Bag of Words> feature into binary term features
3. Create User-Term interaction features
4. Hash features for scalability
5. Learn model

SPAM Detection - Performance



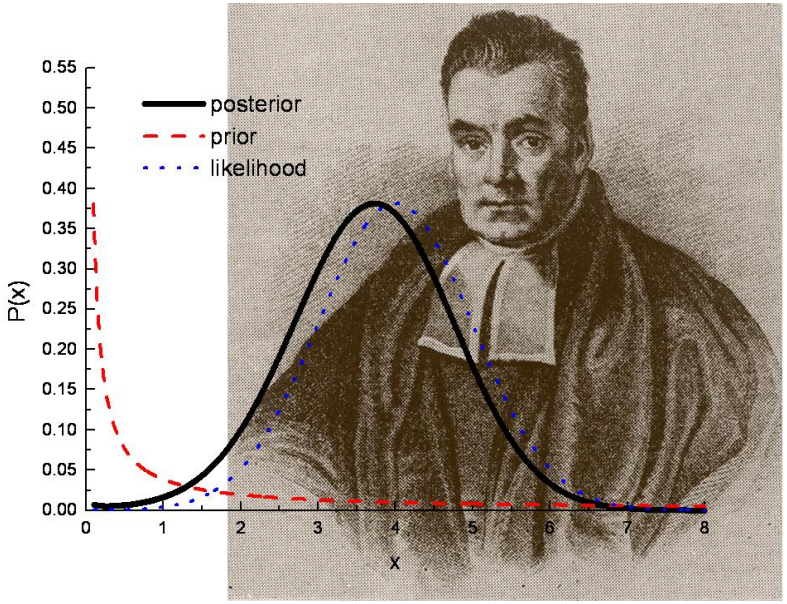
33% Reduction in Spam Misclassification

- **MTL makes individual predictions more accurate**
- **Hashing trick makes MTL scalable**

Graph Source: "Feature Hashing for Large Scale Multitask Learning"

TL in Action

Bayesian Transfer for Online Advertising



Data

A Consumer's Online Activity…

The Non-Branded Web

Conversion/Br and Actions

Gets recorded like this.

UserID	Y	URL_1	URL_2	URL_3	…	URL_k
1	1	0	0	0	…	1
2	1	0	1	1	…	1
3	0	1	0	1	…	1
4	0	1	0	0	…	0
5	1	0	1	1	…	1
…	…	…	…	…	…	…
N	0	0	0	1	…	1

Two Sources of Data

We collect data via different data streams…

UserID	Y	URL_1	URL_2	URL_3	…	URL_k
1	0	1	0	1	…	1
2	0	0	1	0	…	1
3	0	1	1	1	…	0
4	1	0	1	1	…	0
5	1	1	1	0	…	0
…	…	…	…	…	…	…
N	0	1	1	0	…	0

General Web Browsing (Source): $

By pixeling client's web site, we can see who interacts with the site independent of advertising.

UserID	Y	URL_1	URL_2	URL_3	…	URL_k
1	1	0	0	0	…	1
2	1	0	1	1	…	1
3	0	1	0	1	…	1
4	0	1	0	0	…	0
5	1	0	1	1	…	1
…	…	…	…	…	…	…
N	0	0	0	1	…	1

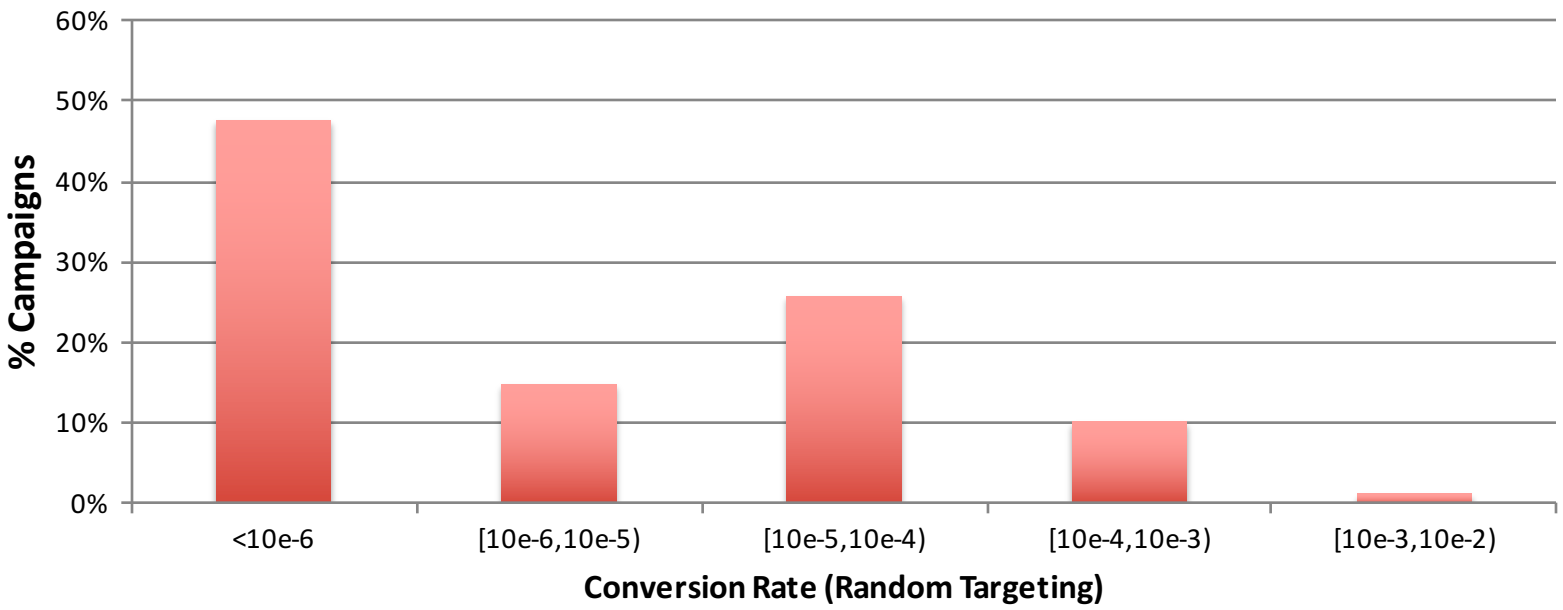
Ad Serving (Target): $$$$$$$$$$$

For every impression served, we track user features and whether user converted after seeing an ad.

dstillery

The Challenges

Because conversion rates are so low (or don't exist prior to campaign), getting enough impression training data is expensive or impossible.

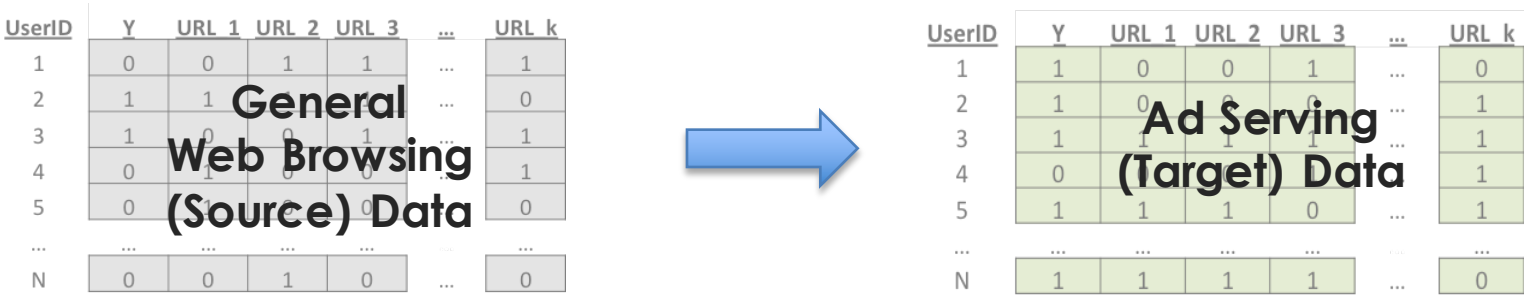


dstillery

Transfer Learning

Assume $f^S(X) \approx f^T(X)$ *and* $f^T(X)$ *is high variance*

Solution: Use $f^S(X)$ *as a regularization prior for* $f^T(X)$



Intuition: The auxiliary model is biased but much more reliable. Use this to inform the real model. The algorithm can learn how much of the auxiliary data to use.

Example Algorithm

1. *Run a logistic regression on source data (using your favorite methodology)*

$$\hat{\mu} = \underset{\mu \in R^k}{\operatorname{argmin}}\ SourceLL(\mu)$$

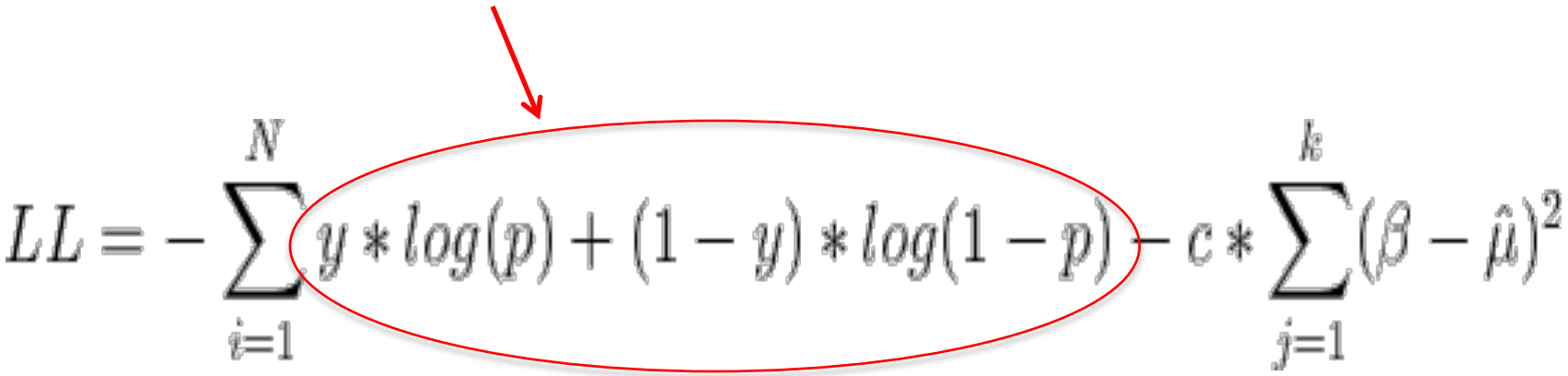
2. *Use results of step 1 as informative-prior for Target model*

$$\hat{\beta} = \underset{\beta \in R^k}{\operatorname{argmin}}\ TargetLL(\beta)$$

$$LL = -\sum_{i=1}^{N} y * log(p) + (1-y) * log(1-p) - c * \sum_{j=1}^{k} (\beta - \hat{\mu})^2$$

Breaking it Down

Standard Log-Likelihood for Logistic Regression



$$LL = -\sum_{i=1}^{N} y * log(p) + (1-y) * log(1-p) - c * \sum_{j=1}^{k} (\beta - \hat{\mu})^2$$

Breaking it Down

Standard Log-Likelihood for Logistic Regression

$$LL = -\sum_{i=1}^{N} y * log(p) + (1-y) * log(1-p) - c * \sum_{j=1}^{k} (\beta - \hat{\mu})^2$$

Prior knowledge from source model is transferred via regularization.

Breaking it Down

Standard Log-Likelihood for Logistic Regression

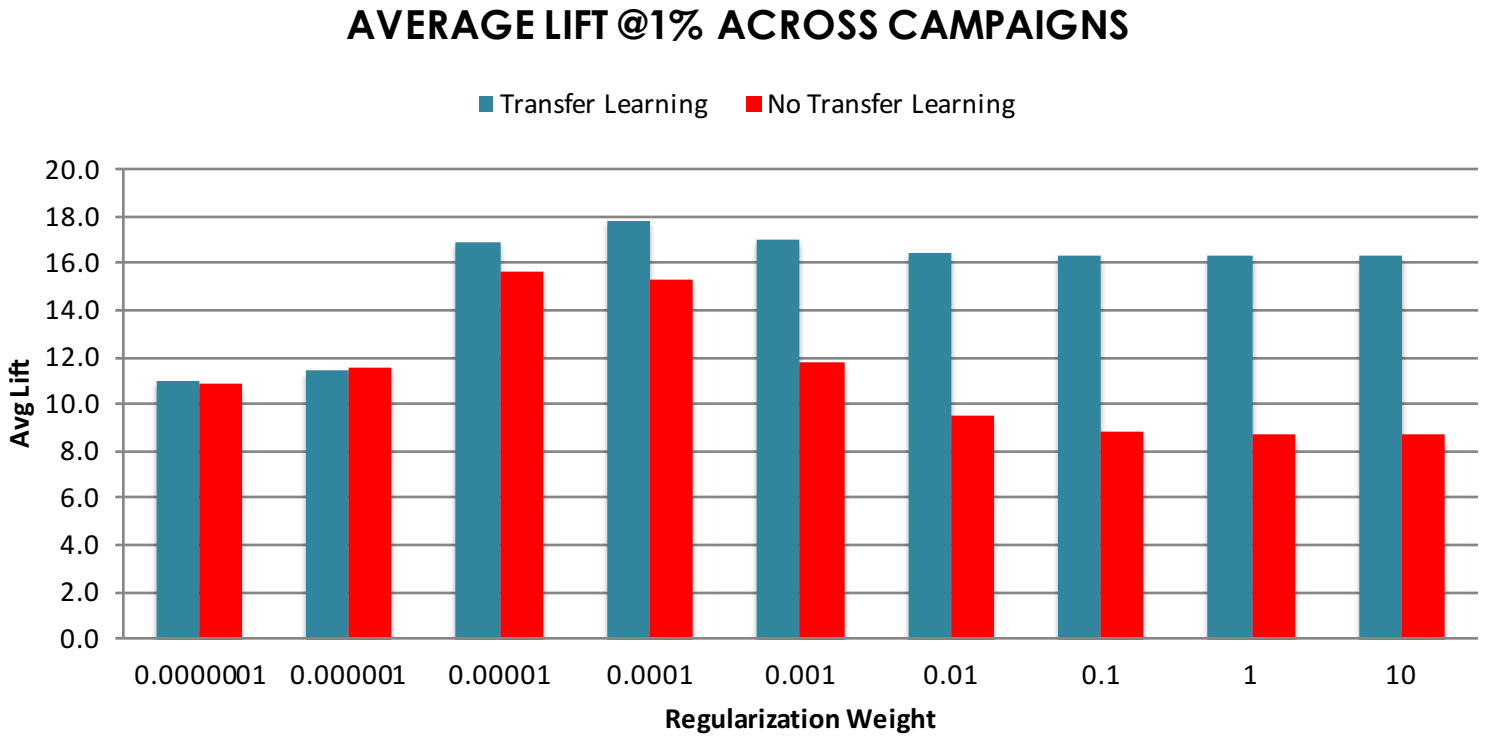
$$LL = -\sum_{i=1}^{N} y * log(p) + (1-y) * log(1-p) - c * \sum_{j=1}^{k} (\beta - \hat{\mu})^2$$

The amount of transfer is determined by the regularization weight.

Prior knowledge from source model is transferred via regularization.

Performance

On average, combining source and target models outperforms using either one by itself. Also, transfer learning is very robust to regularization.



Summary

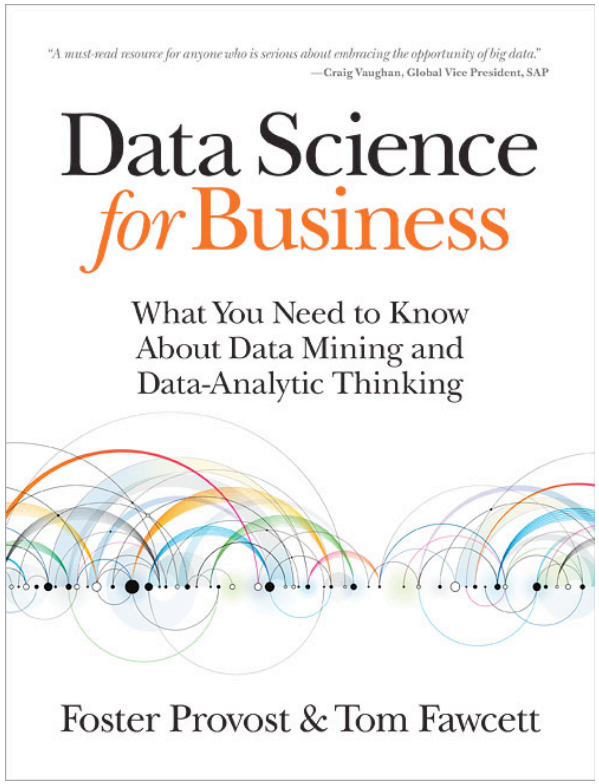
Key Questions

What to transfer?

How to transfer?

When to transfer?

Thinking in Terms of ROI



"data, ***and the capability to extract useful knowledge from data****, should be regarded as key strategic assets"*

Thinking in Terms of ROI

Transfer Learning is another tool for extracting better ROI from existing data assets.

All models are wrong…
Some are useful.

- George E. P. Box

dstillery