Emsambles en Sistemas de Recomendación

Denis Parra

PUC Chile

IIC3633

Combining Predictions for Accurate Recommender Systems

Michael Jahrer
commendo
research&consulting, Graz
University of Technology
8580 Köflach
Austria
michael.jahrer@
commendo.at

Andreas Töscher
commendo
research&consulting, Graz
University of Technology
8580 Köflach
Austria
andreas.toescher@
commendo.at

Robert Legenstein
Institute for Theoretical
Computer Science, Graz
University of Technology
8010 Graz
Austria
robert.legenstein@igi.tugraz.at

Introducción

- Si bien es posible hacer competir diferentes métodos, el artículo de Jahrer et al. indica que hay beneficios en mezclar diferentes recomendadores.

- Los autores lo usaron de forma efectiva en el Netflix Prize.

Modelos

- El artículo describe varios modelos en el rating prediction problem del netflix prize:
  - KNNitem
  - KNNuser
  - SVD
  - AFM
  - AVDe
  - RBM
  - GE

algorithm	RMSE (approx.)	training time	prediction time	memory
KNNitem	0.92	$O(U \cdot M^2)$	$O(Mlg(M))$	$O(M^2)$
KNNuser	0.93	$O(M \cdot U^2)$	$O(Ulg(U))$	$O(U^2)$
SVD	0.90	$O(\|\mathcal{L}\|)$	$O(1)$	$O(M+U)$
AFM	0.92	$O(\|\mathcal{L}\|)$	$O(1)$	$O(M)$
SVDe	0.88	$O(\|\mathcal{L}\|)$	$O(1)$	$O(M+U)$
RBM	0.90	$O(\|\mathcal{L}\|)$	$O(1)$	$O(M)$
GE	0.95	$O(\|\mathcal{L}\|)$	$O(1)$	$O(M+U)$
Blend	<0.87			

- Se indica que una mezcla (blend) obtiene error < 0.87

Alternativas de mezcla

- El artículo describe varias alternativas de “blending”:
  - Linear regression
  - Binned linear regression
  - Neural Network
  - Bagged Gradient Boosted Decision Tree
  - Kernel ridge regression blending
  - K-Nearest Neighbors

Alternativas de mezcla

blending algorithm is formally a function $\Omega : \mathbb{R}^F \mapsto \mathbb{R}$. The input $\mathbf{x}$ is a vector of individual predictions, the output is a scalar. We want to minimize the prediction RMSE on a test set

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\Omega(\mathbf{x}_i) - y_i)^2}. \quad (1)$$

Linear Regression

For any input vector $\mathbf{x}$,

the prediction is $\Omega(\mathbf{x}) = \mathbf{x}^T\mathbf{w}$. Weights $\mathbf{w}$ are calculated with ridge regression, $\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$, where $\mathbf{I}$ denotes the identity matrix. Cross-validation is used in order to select a proper ridge regression constant λ.

Neural Network

Small neural networks can be trained efficiently on huge data sets. The training of neural networks is performed by stochastic gradient descent. The output neuron has a sigmoid activation function with an output swing of -1 to $+1$. To generate rating predictions in the range of $[1, 5]$ we

$+1$. To generate rating predictions in the range of $[1, 5]$ we use a simple output transformation. For example the output is multiplied by $\alpha = 3.6$ and the constant $\beta = 3.0$ is added (works well on our experiments). The learning rate is η and every epoch the constant $\eta^{(-)}$ is subtracted, which

Neural Network

Small neural networks can be trained efficiently on huge data sets. The training of neural networks is performed by stochastic gradient descent. The output neuron has a sigmoid activation function with an output swing of -1 to $+1$. To generate rating predictions in the range of $[1, 5]$ we

$+1$. To generate rating predictions in the range of $[1, 5]$ we use a simple output transformation. For example the output is multiplied by $\alpha = 3.6$ and the constant $\beta = 3.0$ is added (works well on our experiments). The learning rate is η and every epoch the constant $\eta^{(-)}$ is subtracted, which

Bagged Gradient Boosted Decision Tree - BGBDT

- Boosting
- Bagging
- Gradient Boosted Trees

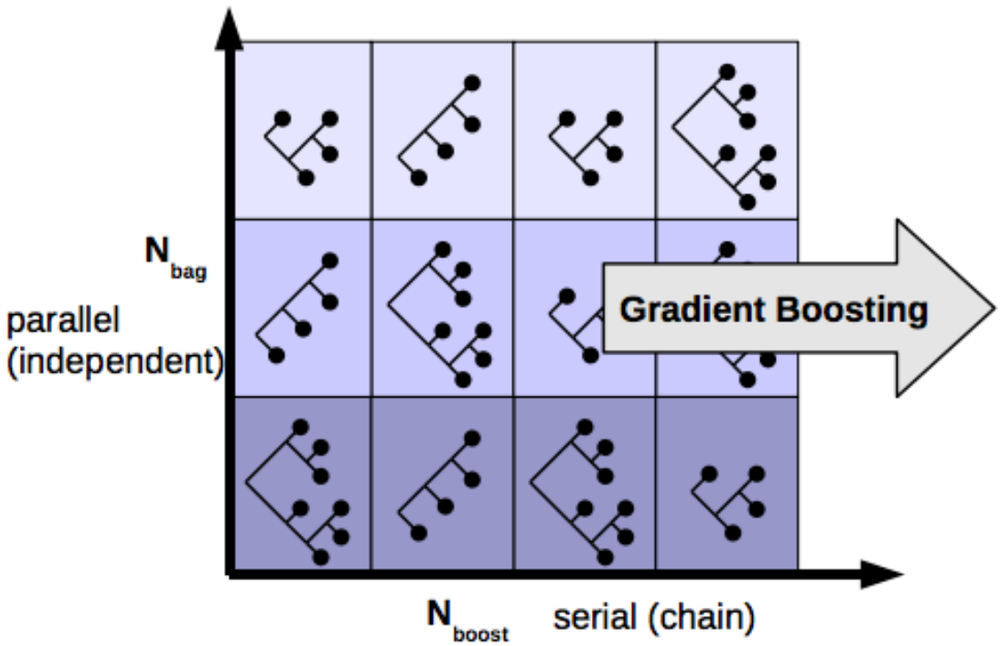
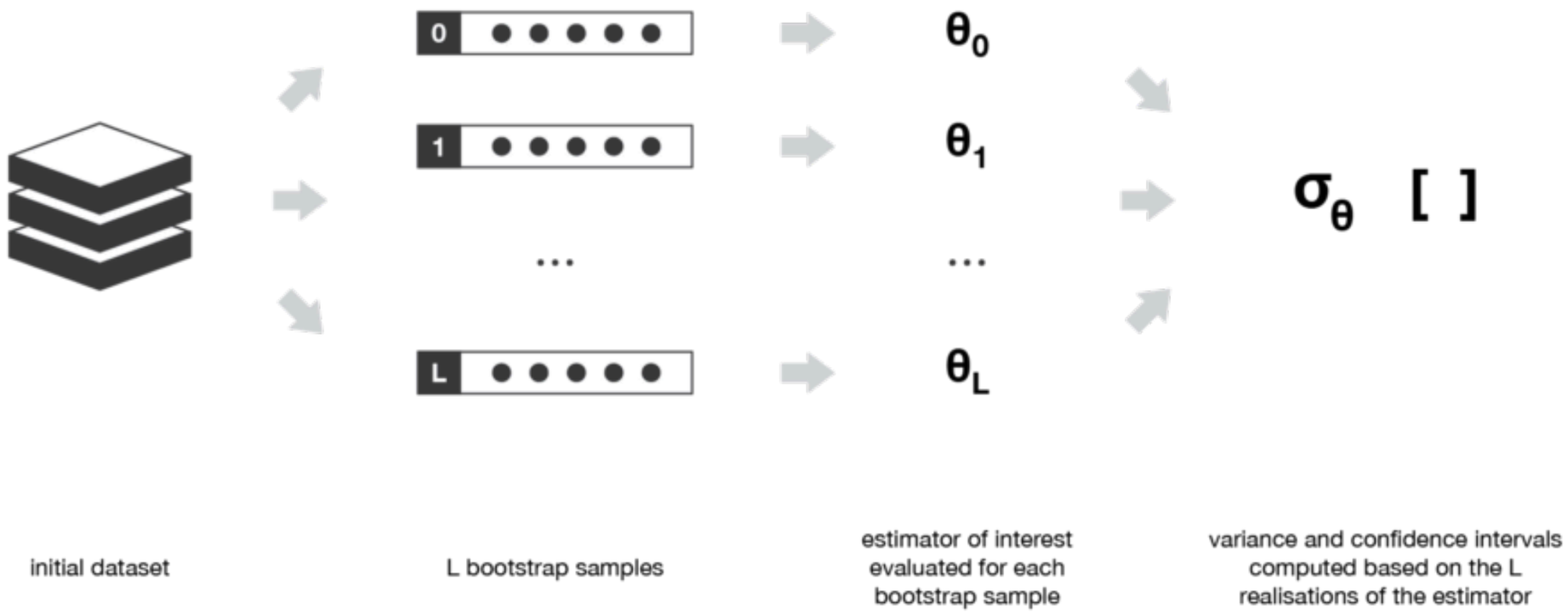


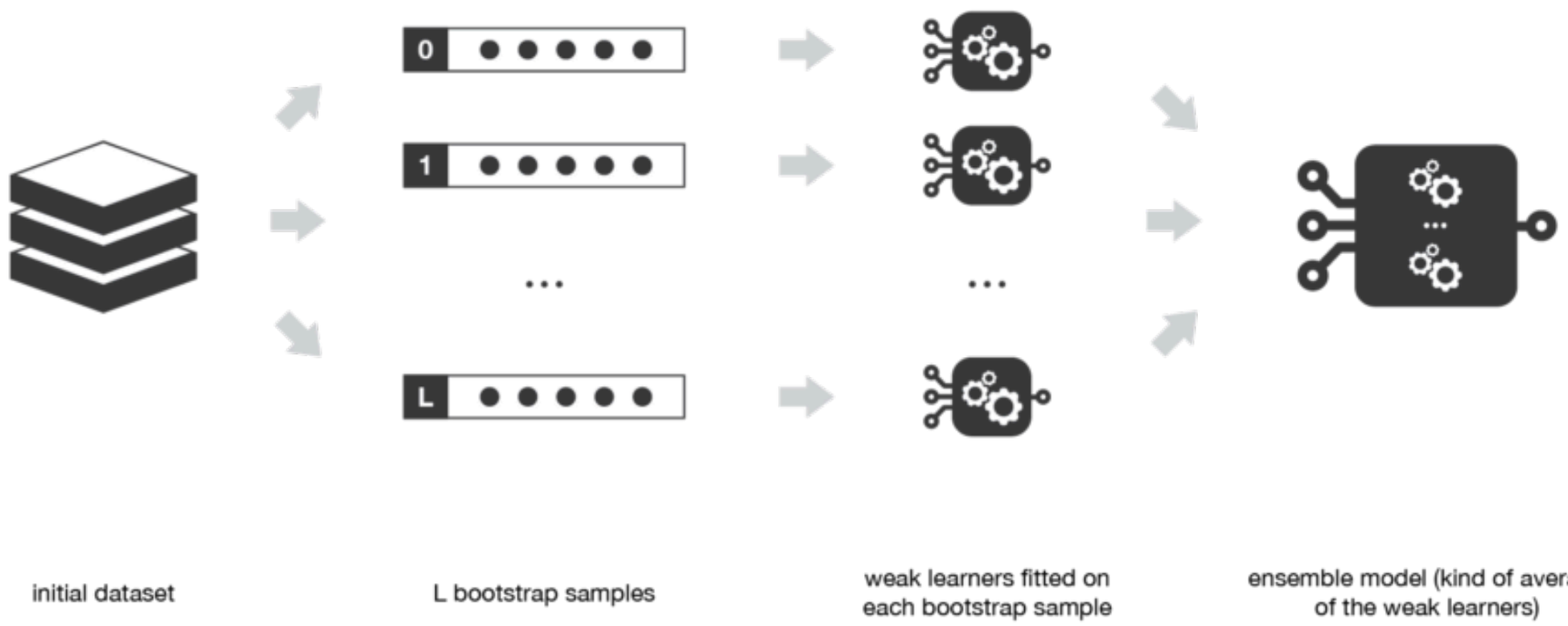
Figure 2: Bagged Gradient Boosted Decision Tree. A prediction of the BGBDT consists of results by $N_{bag} \cdot N_{boost}$ single decision trees. Both, bagging and gradient boosting improves the accuracy.

Boosting



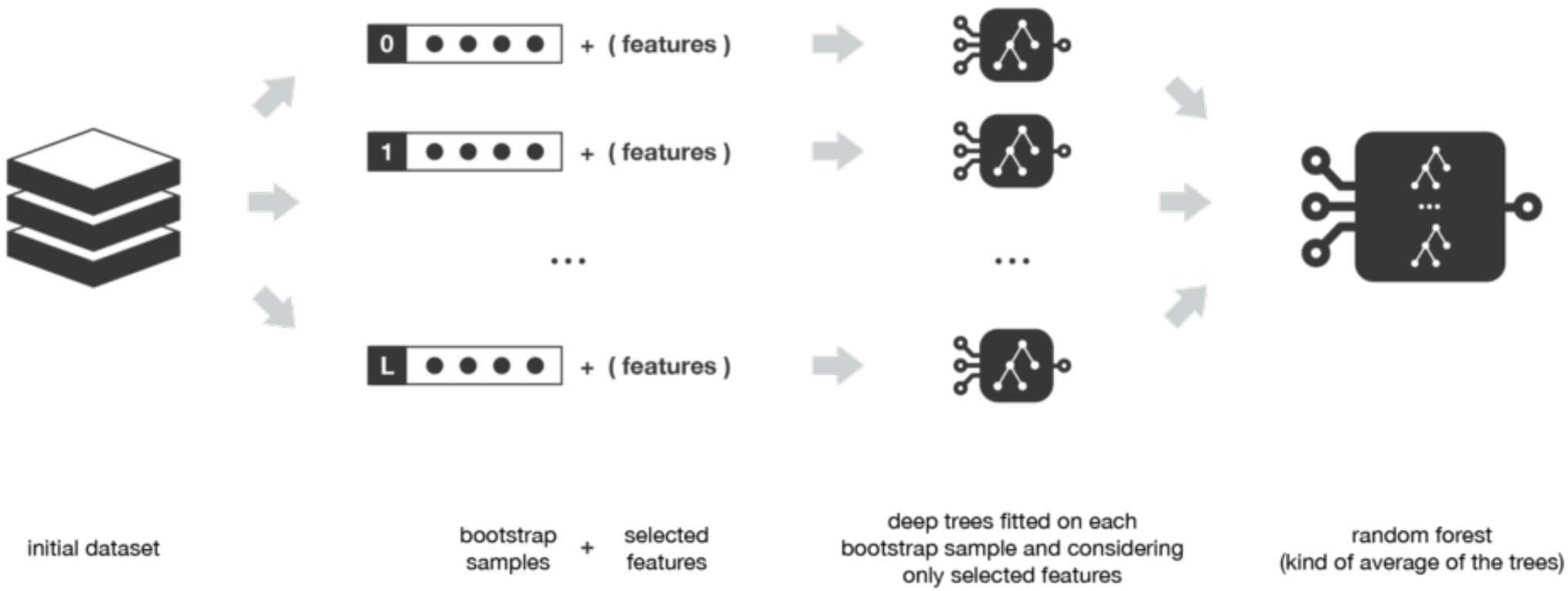
Fuente: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Bagging



Fuente: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

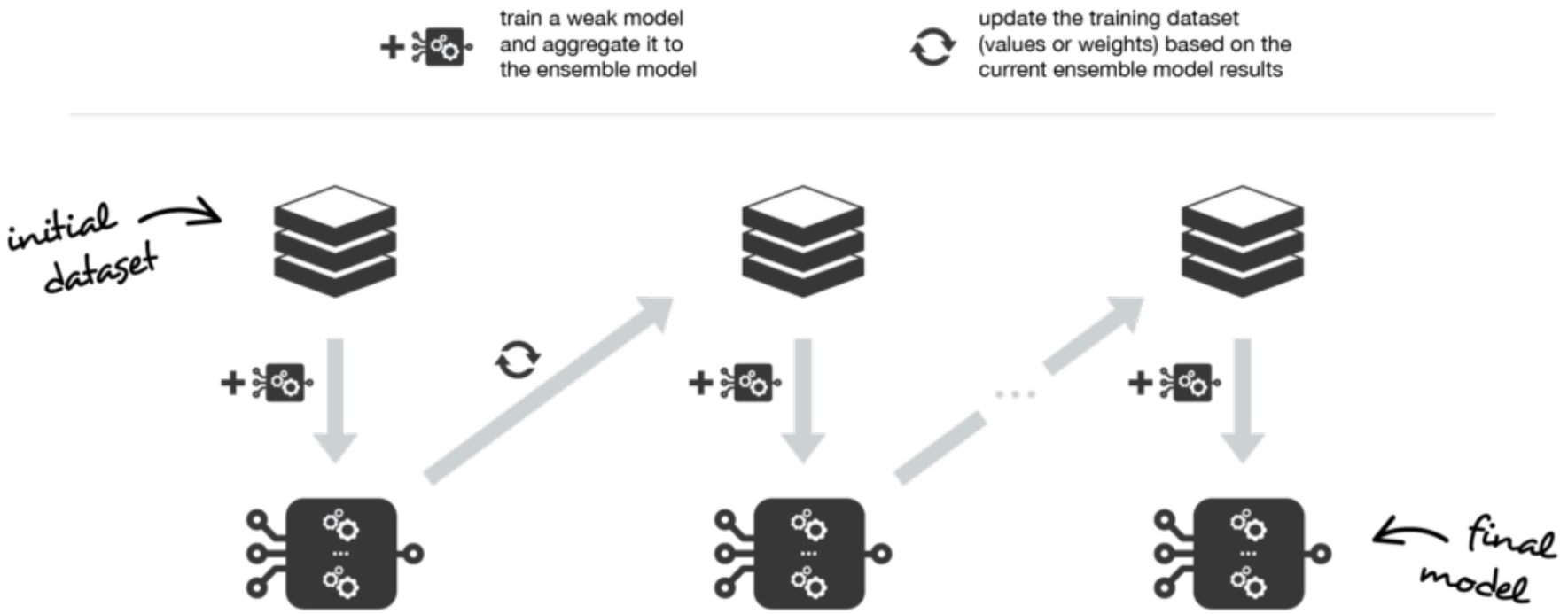
Bagging – Random Forests



Fuente: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

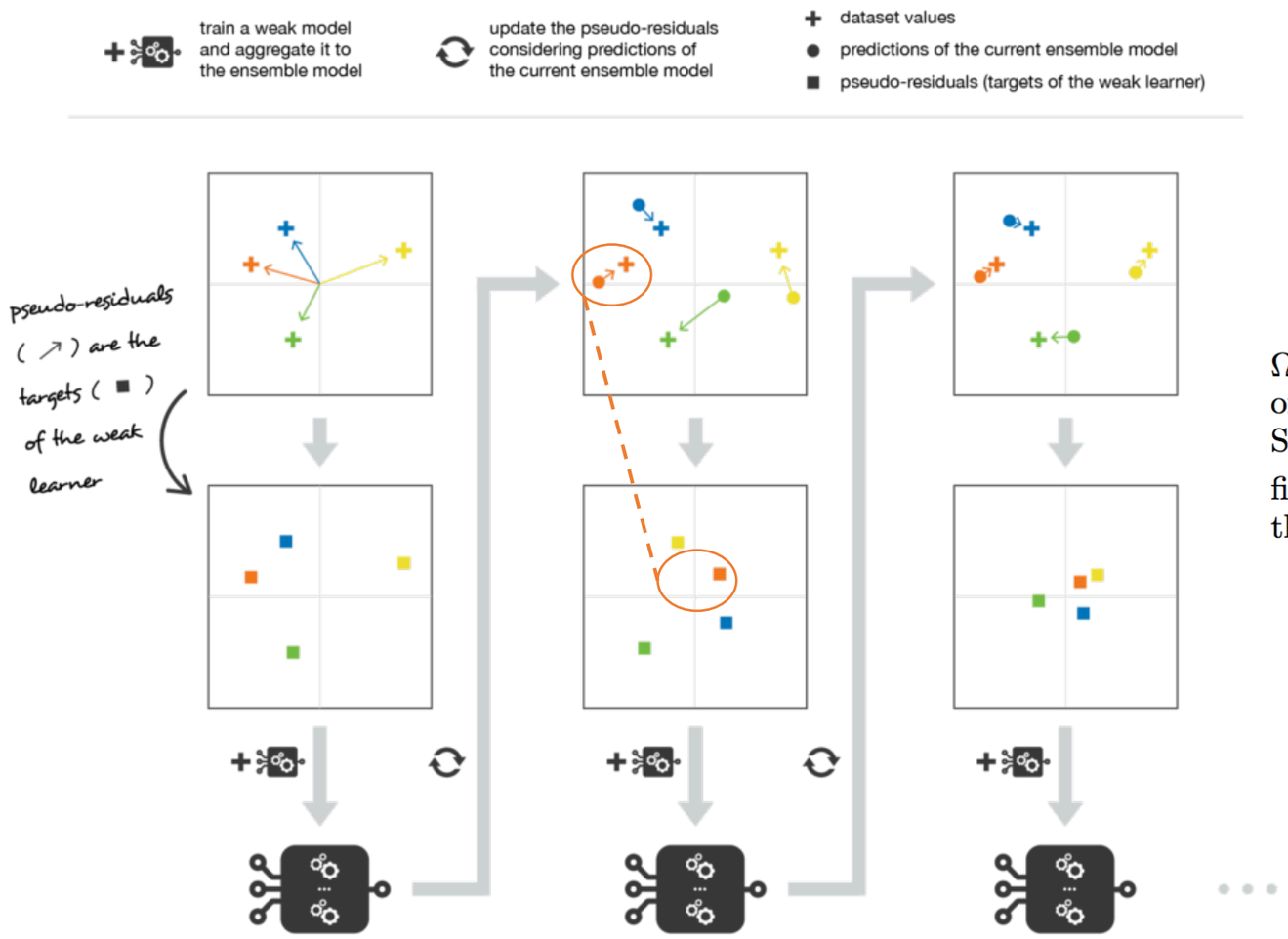
Boosting como ML

Ω_1 in the chain is trained on the unmodified targets $\mathbf{y}_1 = \mathbf{y}$ of the dataset. The second model trains on $\mathbf{y}_2 = \mathbf{y} - \eta\,\mathbf{y}_1$. So targets in each cascade are $\mathbf{y}_i = \mathbf{y} - \sum_{j=1}^{i-1} \eta\,\mathbf{y}_j$. The final prediction model is $\Omega(\mathbf{x}) = \sum_{i=1}^{M} \eta\,\Omega(\mathbf{x})$, where M is the length of the boosting chain.



Fuente: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Gradient Boosted Trees



Ω_1 in the chain is trained on the unmodified targets $\mathbf{y}_1 = \mathbf{y}$ of the dataset. The second model trains on $\mathbf{y}_2 = \mathbf{y} - \eta\,\mathbf{y}_1$. So targets in each cascade are $\mathbf{y}_i = \mathbf{y} - \sum_{j=1}^{i-1} \eta\,\mathbf{y}_j$. The final prediction model is $\Omega(\mathbf{x}) = \sum_{i=1}^{M} \eta\,\Omega(\mathbf{x})$, where M is the length of the boosting chain.

Fuente: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Revisitemos BGBDT

- Boosting
- Bagging
- Gradient Boosted Trees

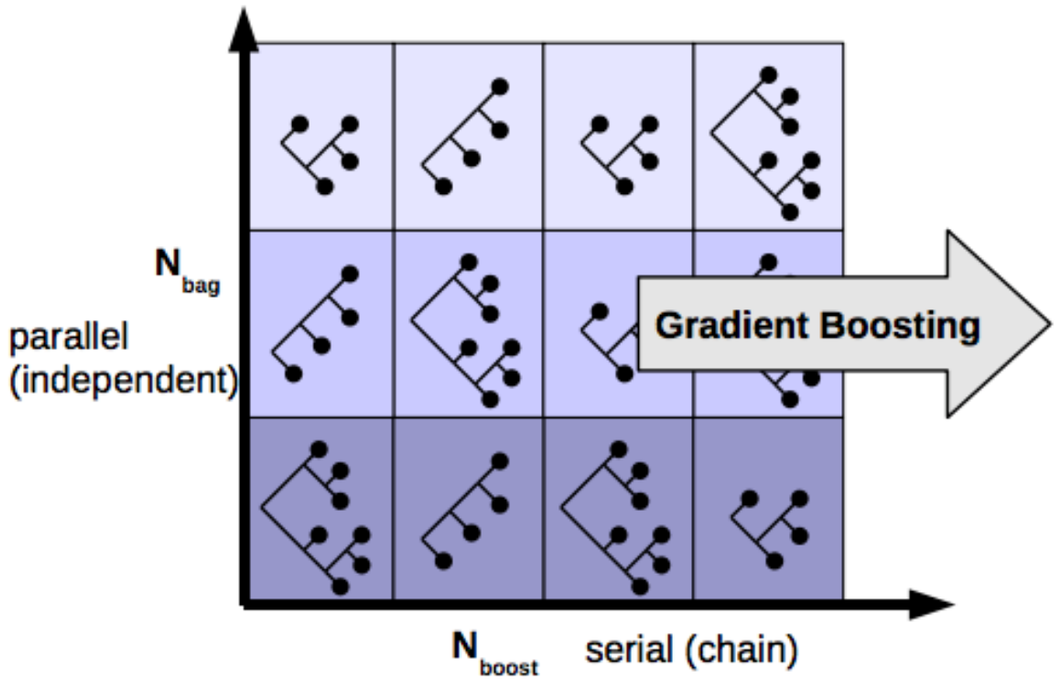


Figure 2: Bagged Gradient Boosted Decision Tree. A prediction of the BGBDT consists of results by $N_{bag} \cdot N_{boost}$ single decision trees. Both, bagging and gradient boosting improves the accuracy.

Resultados - LR

AFM-1 (0.9362)	−0.083
AFM-2 (0.9231)	−0.084
AFM-3 (0.9340)	−0.077
AFM-4 (0.9391)	+0.088
GE-1 (0.9079)	+0.098
GE-2 (0.9710)	−0.003
GE-3 (0.9443)	−0.081
GE-4 (0.9209)	+0.176
KNN-1 (0.9110)	+0.029
KNN-2 (0.8904)	+0.272
KNN-3 (0.8970)	−0.094
KNN-4 (0.9463)	+0.010
RBM-1 (0.9493)	+0.025
RBM-2 (0.9123)	+0.066
SVD-1 (0.9074)	−0.008
SVD-2 (0.9172)	+0.094
SVD-3 (0.9033)	+0.080
SVD-4 (0.8871)	+0.227
log(support)	−0.008
const. 1	+3.673

Table 3: Blending weights of an optimal linear combination. This leads to 0.875258 RMSE on the pTest set. The RMSE on the cross validation set is 0.87552. The number in the brackets is the probe RMSE per model.

Resultados - NN

network setup	validation type	RMSE validation	train time	RMSE pTest
19-30-1	retraining 8-CV	0.873633	1.5[h]	0.873365
19-30-1	cross valid. mean 8-CV	0.873633	0.88[h]	0.873316
19-30-1	bagging size=32	0.87347	4.3[h]	0.873191
19-30-1	bagging size=128	0.873436	17.3[h]	0.873185
19-70-1	bagging size=128	0.87342	33.6[h]	0.873163
19-150-1	bagging size=128	0.873473	65.8[h]	0.873169
19-50-30-1	bagging size=128	0.873455	48.6[h]	0.87318

Table 5: Results from different neural network blends. We use in all networks the same learning rate $\eta = 5e{-}4$, $\eta^{(-)} = 5e{-}7$. Output transformation constants are $\alpha = 3.6$, $\beta = 3.0$.

Resultados - BGBDT

fixed:	$\eta = 0.1$	$\eta = 0.05$	$\eta = 0.03$	$\eta = 0.02$
$N_{bag} = 32$	0.874783	0.87467	0.874624	0.874593
$K = 300$	0.87437	0.874352	0.87433	**0.874309**
$S = 2$	0.62[h]	1.21[h]	1.94[h]	3.27[h]
fixed:	$K = 500$	$K = 300$	$K = 200$	$K = 100$
$N_{bag} = 32$	0.874838	0.874783	0.874767	0.874934
$\eta = 0.1$	0.874427	**0.87437**	0.874399	0.874546
$S = 2$	0.48[h]	0.62[h]	0.73[h]	1.39[h]
fixed:	$N_{bag} = 16$	$N_{bag} = 32$	$N_{bag} = 64$	$N_{bag} = 128$
$\eta = 0.02$	0.874936	0.874593	0.874554	0.874517
$K = 300$	0.874381	0.874309	0.874293	**0.874288**
$S = 2$	1.1[h]	3.27[h]	7.01[h]	14.66[h]
fixed:	$S = 1$	$S = 2$	$S = 4$	$S = 8$
$\eta = 0.1$	0.874838	0.874784	0.87477	0.874841
$K = 500$	0.874427	**0.874377**	0.874405	0.874504
$N_{bag} = 32$	0.48[h]	0.42[h]	0.68[h]	1.11[h]

We analyzed Bagged Gradient Boosted Decision Trees with varying bagging size N_{bag}, varying subspace size S, varying number of leaves K, and varying learning rate η.

Table 6: BGBDT blending results. The first column denotes the fixed parameters. In the next columns the first line is the tested parameters, second line is the validation RMSE, third line is the pTest RMSE and fourth line is the training time. We vary the learn rate η, the subspace size K and the bagging size N_{bag}. For all results we use optimal splits in training a single tree.

Resultados – Bag with NN, LR, GBDT

model	RMSE (blend)	weight	parameters
const. 1	-	-0.014	-
NN	0.87345 (0.873445)	0.170	19-100-1, $\alpha = 3.6$, $\beta = 3.0$, $\eta = 5e{-}4$, $\eta^{(-)} = 5e{-}7$, 870 epochs, 44.4[h]
GBDT	0.874111 (0.873387)	0.054	$S = 20$, $K = 50$, $\eta = 0.1$, 226 epochs, randomSplitPoint, 6.6[h]
GBDT	0.874603 (0.873384)	0.098	$S = 2$, $K = 300$, $\eta = 0.02$, 267 epochs, optSplitPoint, 8.1[h]
PR	0.874358 (0.87336)	0.141	order=2, $\lambda = 2.4e{-}6$, with cross interactions, 1.9[h]
PR	0.895951 (0.873351)	-0.033	order=3, $\lambda = 0.054$, no cross interactions, 0.3[h]
NN	0.87345 (0.873296)	0.202	19-100-1, $\alpha = 2$, $\beta = 3.0$, $\eta = 5e{-}4$, $\eta^{(-)} = 5e{-}7$, 998 epochs, 47.1[h]
NN	0.873449 (0.873227)	0.371	19-50-30-1, $\alpha = 2$, $\beta = 3.0$, $\eta = 5e{-}4$, $\eta^{(-)} = 5e{-}7$, 952 epochs, 49.8[h]
blend	0.873227 pTest:0.87297		total train time: 158.2[h] total prediction time 4.5[h]

Table 7: Bagging and linear combination of many blending models applied to collaborative filtering for the Netflix Prize dataset. The first column indicates the model type. The second column reports the individual out-of-bag RMSE estimate, the number in brackets below is the blend RMSE. The third column shows the weight of the model. The fourth column shows metaparameters and the training time of each model. Accurate blending methods receive higher weights. $N_{bag} = 128$ in all models.

Tiempo y recomendaciones

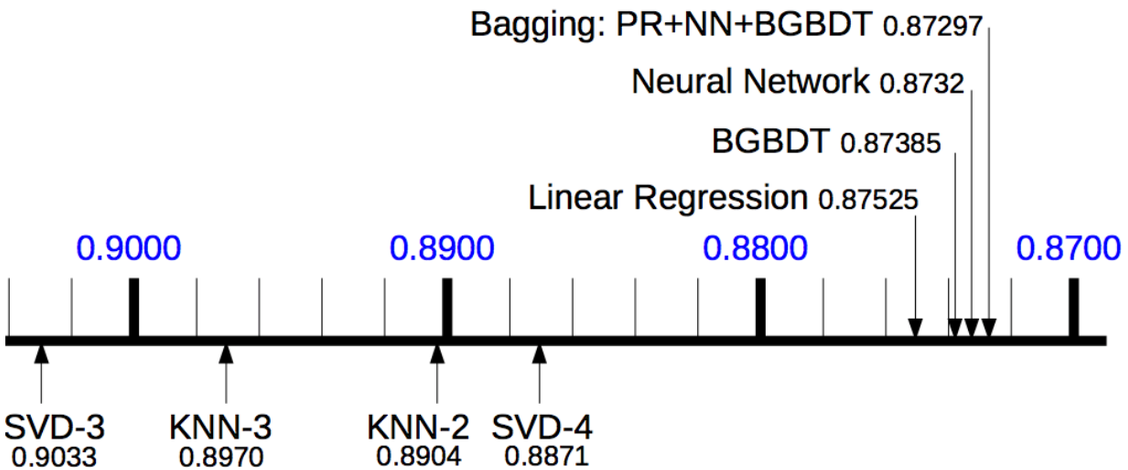


Figure 5: RMSE values on the Netflix Prize probe set. Shown are various blending methods (above the scale) and the best performing single algorithms (below the scale).

For practical applications we recommend to use a neural network in combination with bagging due to the fast prediction speed. A set of 10^6 samples can be predicted in a few seconds with this technique. We showed that a large ensemble of different collaborative filtering models leads to an accurate prediction system.

Gracias!

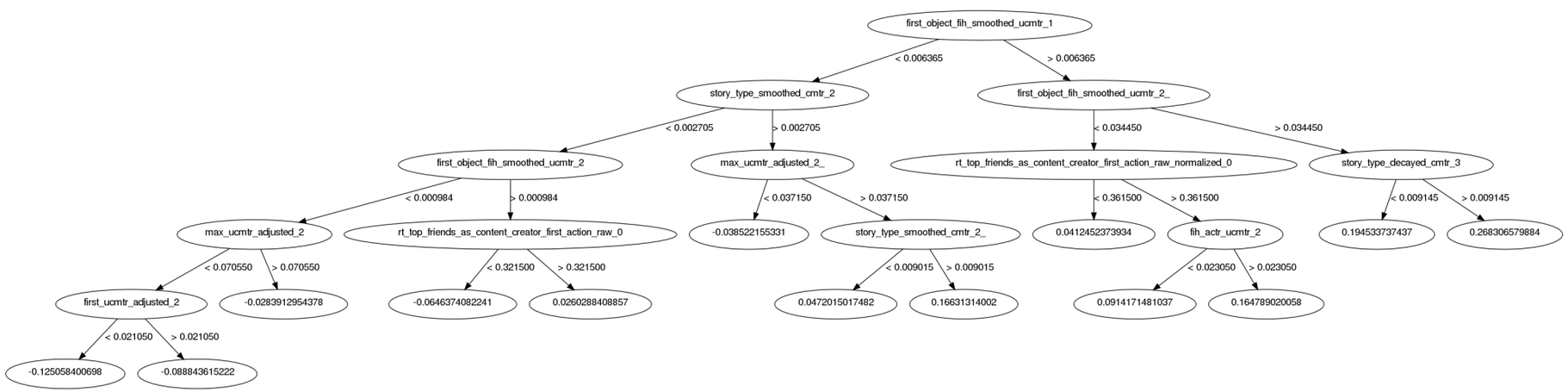
- dparra@ing.puc.cl

Facebook

- Presentación en LSRS 2016, Komal Kapoor

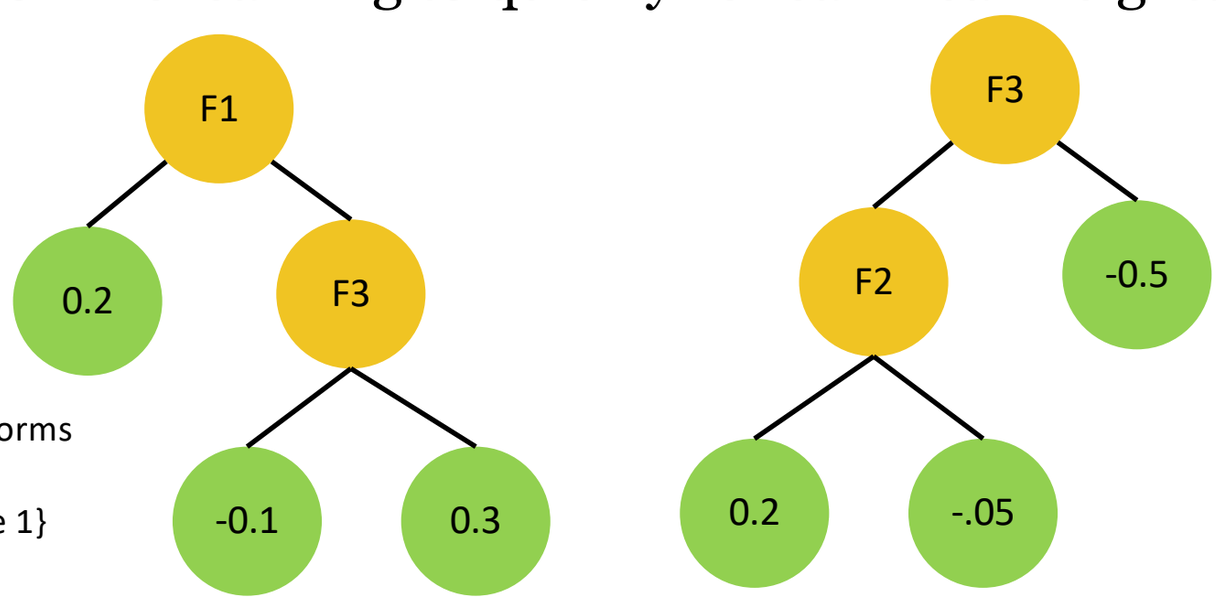
Feature Selection (BDTs)

- Prune to the most important features (~2K)
  - Training time is proportional to number of examples * number of features
  - Under-sample negative examples (impressions, no action) to help with # of examples
  - Reduce noise and results in simpler trees
- Do this for each feed event type: train many forests
- Historical counts and propensity are some of the strongest features



Model Training (Logistic regression)

- We need to **react quickly** and incorporate new content - use a simple model
- Logistic regression is **simple, fast and easy to distribute**
- Treat the trees as feature transforms, each one turning the input features into a set of categorical features, one per tree.
- Use logistic regression for online learning to quickly re-learn leaf weights



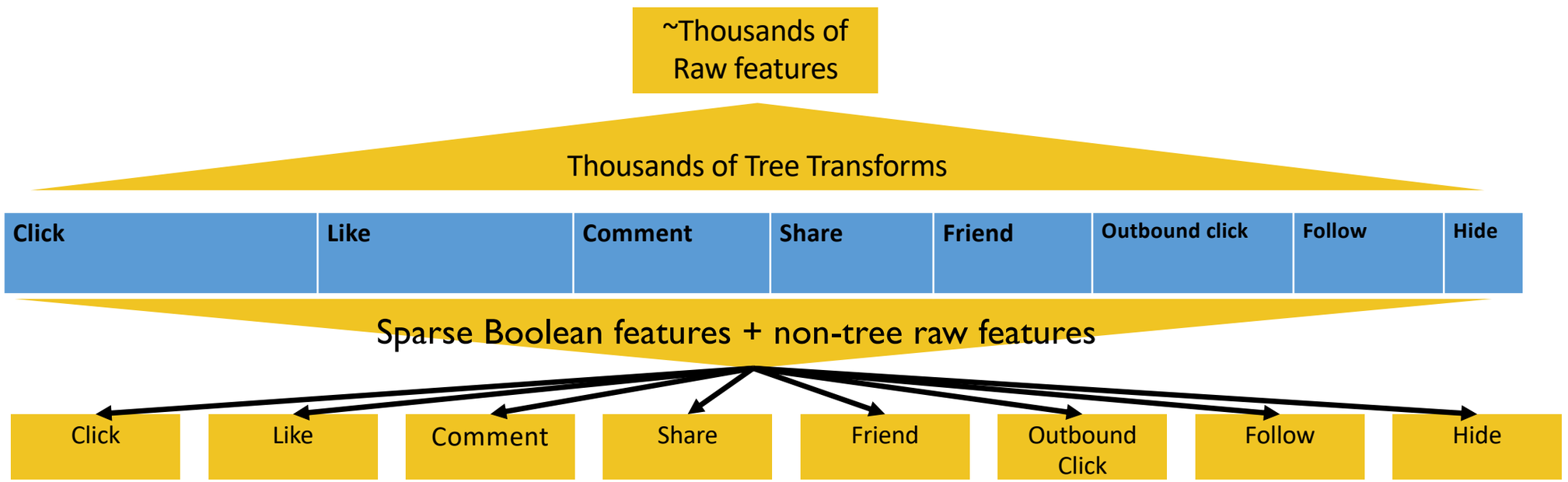
Throw out boosted tree weights, use only transforms

Input: (F1, F2, F3)

Output (T1, T2) where T1 ∈ {Leaves of tree 1}

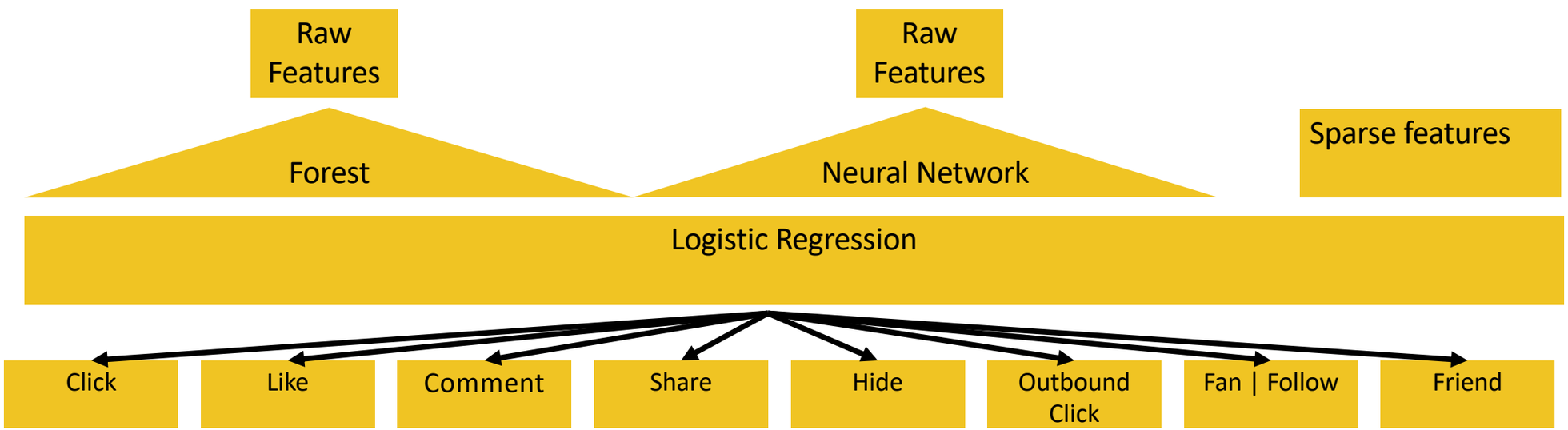
Stacking: Combined Tree + LR Model

- Main Advantage: Tree application is computationally resource intensive and slow
  - Reuse click tree to predict likes, comments, etc.
- Only slightly more resource intensive than independent models; better prediction performance – transfer learnings



Other models + sparse features

- Train Neural nets to predict events
  - Discard final layer, use **final layer outputs** as features
- Add sparse features such as text or content ID



Learnings

- Data freshness matters – simple models allows for online learning and twitch response
- Feature generation is part of the modeling process
- Stacking
  - Supports plugging-in new algorithms and features easily
  - Works very well in practice
- Use skewed sampling to manage high data volumes
- Historical counters as features provides highly predictive features, easy to update online