OPEN
DATA
SCIENCE
CONFERENCE_

#ODSC

BOSTON
2015
@OPENDATASCI

OPEN SOURCE TOOLS & DS COMPETITIONS

Owen Zhang

Open Source Tools and DS Competitions

Owen Zhang
Data Scientist @ DataRobot Inc.

Agenda

- Acknowledgement
- Why Open Source
- Retrospective
- Some tools I (am learning to) use
  - Vowapl Wabbit, Xgboost, LibFFM, Neural Networks
- Randomness is good
- Putting them all together
- Further reading

Acknowledgement

- “Thank you” to the authors and the open source community
- Special thanks to my colleague Xavier Conort
- I learned everything I know about machine learning / statistical modeling from the community
  - I was trained as an Engineer!

Xavier Conort
Chief Data Scientist @ DataRobot

Open source tools are perfect for data science competitions

- Imagine a world where there is only SAS/SPSS

- They are free
- They are open so we can see what is under the hood and tweak them
- They are open so we can share our tweaks with each other

Open Competition + Quick Feedback + Open Sharing = Rapid Progress!

The good old days

A few years ago, we could do well (even won prize) with:

- Categorical feature encoder
- (GBM + GLMNET) / 2
- Training + Validation

Today

- People post the above as “Beat the Benchmark” code in the forum

Vowpal Wabbit

- By John Langford
- Very fast online SGD -- super flexible with many options
- My favorite -- predictor interactions (quadratic and cubic features) on-the-fly
- Enables fast iterative development of features and model structure

- https://github.com/JohnLangford/vowpal_wabbit/wiki

Vowpal Wabbit

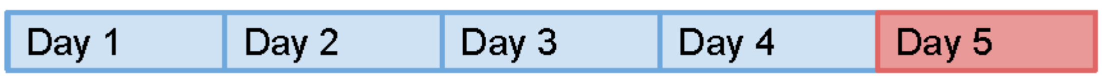
- We can see the quality of the prediction as soon as the algorithm starts running.
  - Realize something works (or does not) very quickly
- Being able to experiment with different features/interactions without re-creating huge data
- It is possible to try out 100s of different combinations of regularization/feature interaction in a few hours on 10s of GB of data.

Tuning Vowpal Wabbit

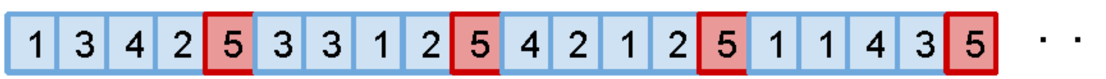
- -l2 : L2 regularization
  - Still worth trying to combine really infrequent levels
- -q : quadratic interaction
- -b : # of bits in feature table
  - Sometimes lower -b value can be used as regularization
- -l : learning rate
- --passes: # of passes over data
- -cubic --interactions : 3rd order and higher interactions

Vowpal Wabbit validation setup

- --holdout_period : every k observation is used for validation. But this won’t work for out-of-time validation
- Naive set up will give misleading indication of model performance



--holdout_period=5



XGBoost - a worthy successor to GBM

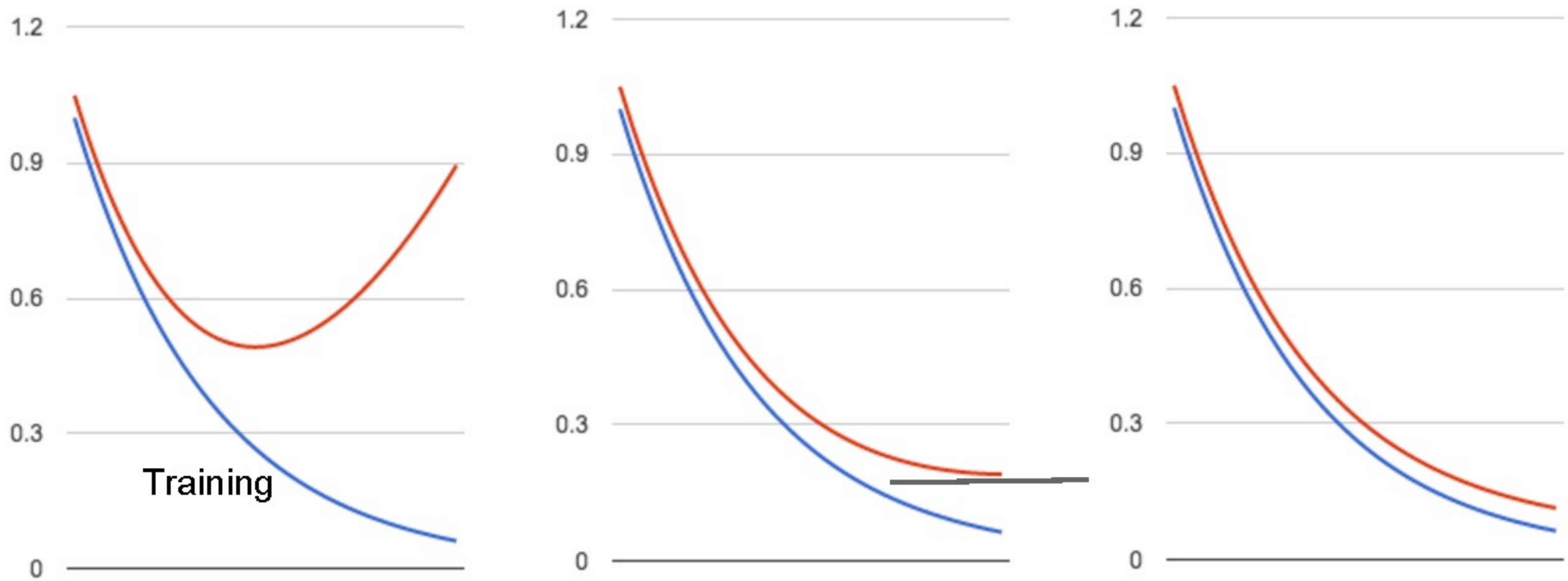
- GBM package in R (Greg Ridgeway) was my “go to” tool for quite a while, but I am hooked on xgboost now.
- XGBoost was created by Tianqi Chen at U of Washington
  - Parallel - openMP based multi-core
  - Feature sampling (in addition to row sampling)
- The only thing missing -- partial dependence plots?

- https://github.com/dmlc/xgboost

Tuning XGBoost

- eta (learning rate) + num_round (number of trees)
  - Examine objective metric in training/validation to quickly find good configuration
  - Target around 100 trees
- max_depth (start with 6) -- This is different from R GBM
- min_child_weight (start with 1/sqrt(event rate))
- colsample_bytree (.3-.5)
- subsampling (leave at 1.0)
- gamma (usually is it OK to leave at 0.0)

Tuning Heuristics



Too complex:
eta --
max_depth --
min_child_weight ++

Just Right!

Too simple:
eta ++
max_depth ++
min_child_weight --

LibFFM

- By Machine Learning Group at NTU
- Structured matrix factorization but applicable beyond recommender systems
- Another way of modeling interactions
  - Especially effective when there are high cardinality categorical features
- Parallel with SSE optimization = really fast!

- http://www.csie.ntu.edu.tw/~cjlin/libffm/

Tuning LibFFM is (relatively) easy

- Lot of similarity to VW, but simpler
- -l : L2 regularization
- -k: latent factor
- -r: learning rate
- -t: # of iterations

- LibFFM expects all input to be categorical, it is usually a good idea to use tree (or boosted tree) based binning if there are numerical features -- more on this later

Neural Networks

- Back Propagation is back!
- With a few new tricks:
  - Dropout as an effective regularization approach
  - New form of activation functions
    - ReLU and its variants
- Most NN packages can be accelerated with a GPU -- a good excuse to upgrade your computer
  - 1024x1024x1024x1024x1024 network trained in minutes, instead of hours/days

Neural networks

- Theano and packages built on top: PyLearn2, Lasange
- And Keras https://github.com/fchollet/keras
- NNs are harder to tune due to its flexibility
- There are infinite possibilities of structure
- Some similarity with GBMs:
  - # of layers ⇔ depth of tree
  - size of layer ⇔ # of leaf nodes
  - Learning rate ⇔ learning rate
  - Training epochs ⇔ # of trees?

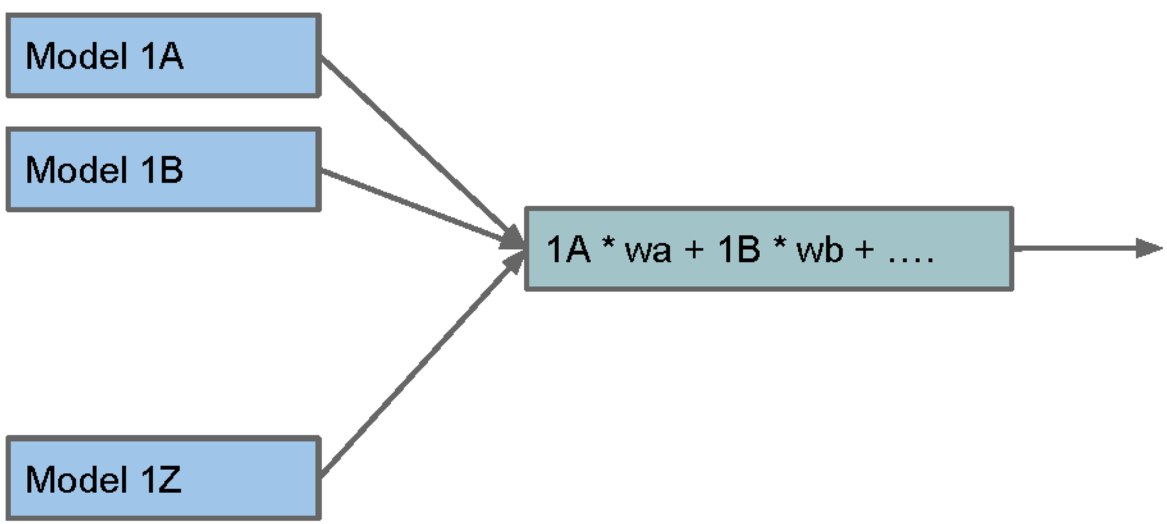
Randomness is good

- All these tools have inherent randomness
- This is a feature not a bug
- By simply running them with different seeds and average the output, we get superior results
- It also helps to make the process as random as possible:
    - Resample/resplit the data
    - Remember to change the random seed

- VW, LibFFM, and NN are also order dependent
    - Make sure to randomly shuffle input data

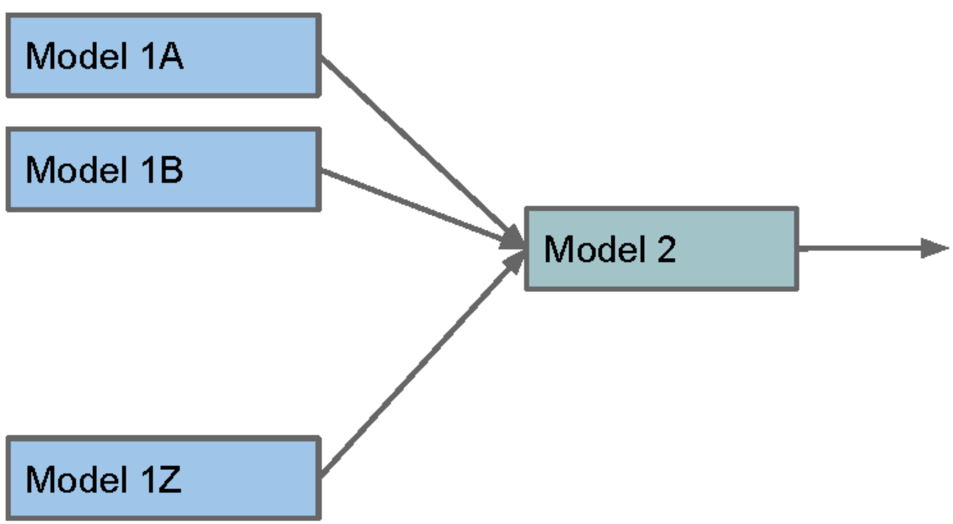
Beyond (GBM + GLMENT) / 2

- Simple model averaging was easy
  - And you can (over)fit the public Leaderboard



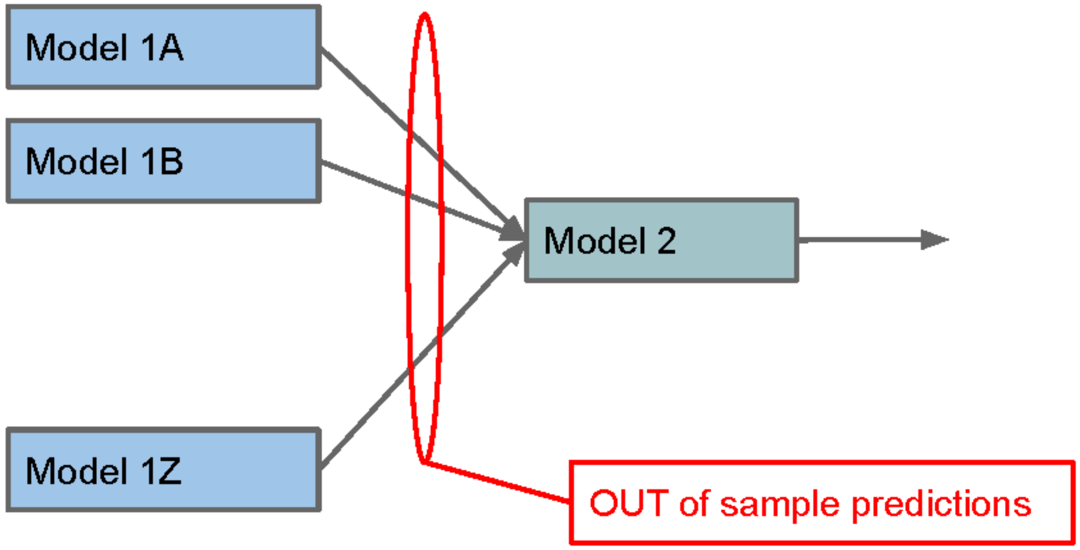
Stacking -- model on model predictions

- Stacking -- Another model on top of first level models



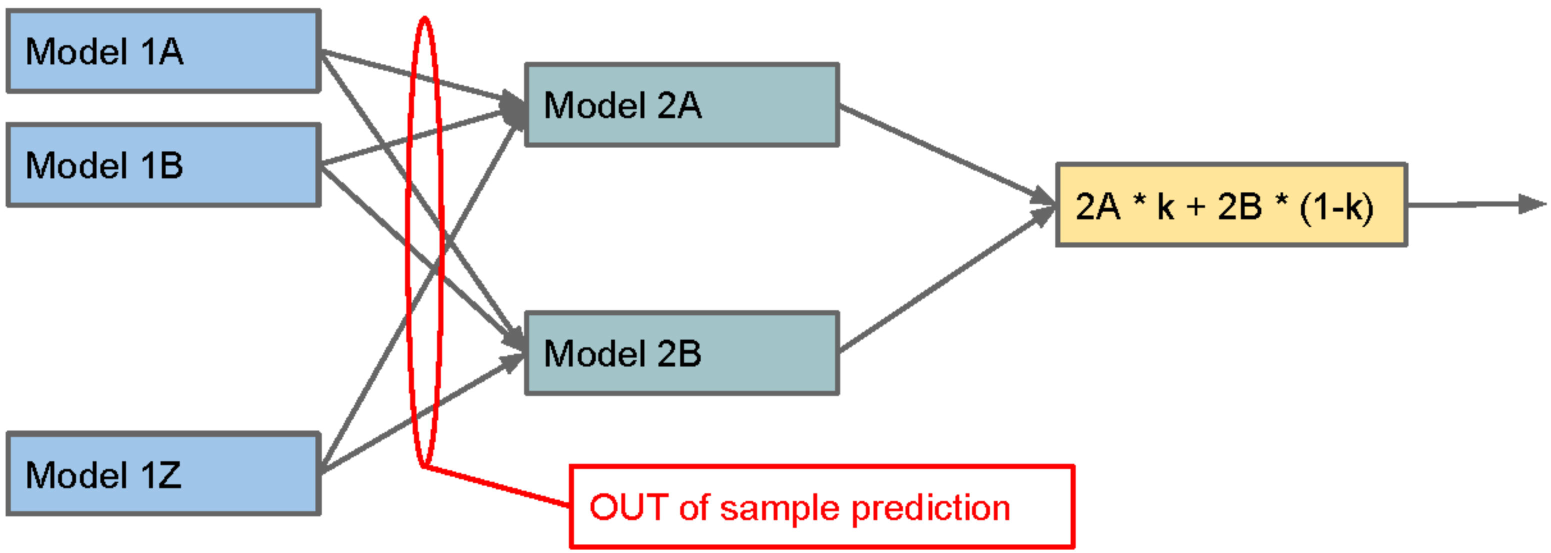
Do not Overfit

- Stacking -- MUST use OUT of sample predictions, usually CV predictions



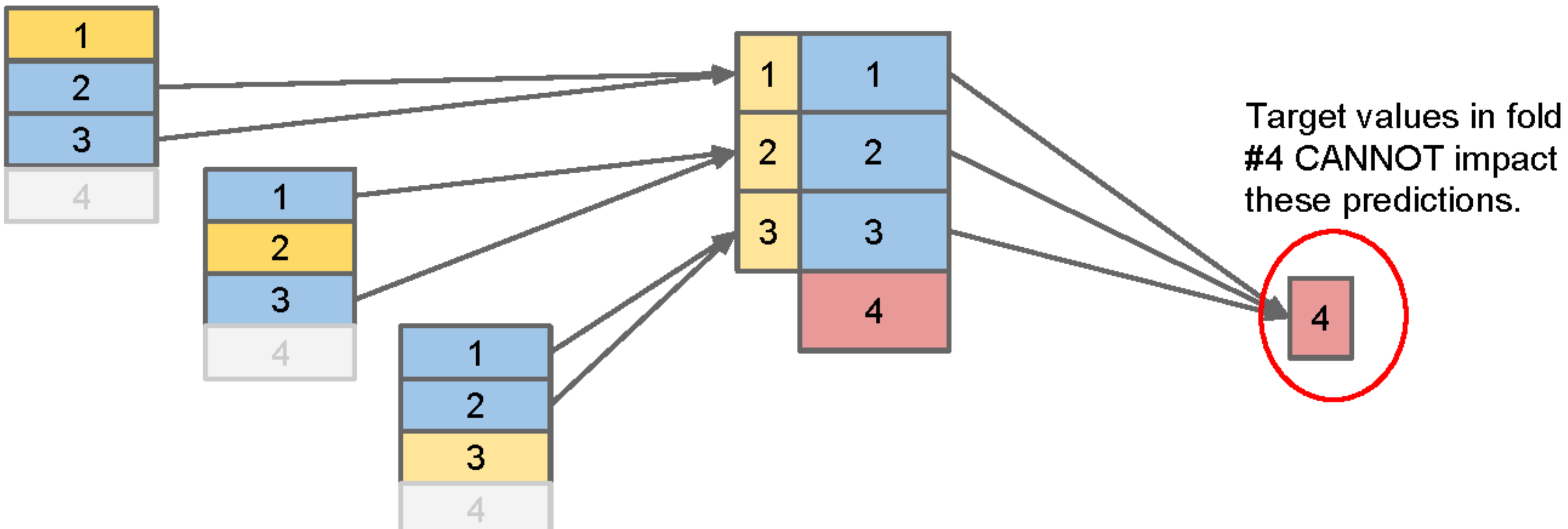
One more Level -- Model^3?

- Popular nowadays -- blend stacking predictions one more time



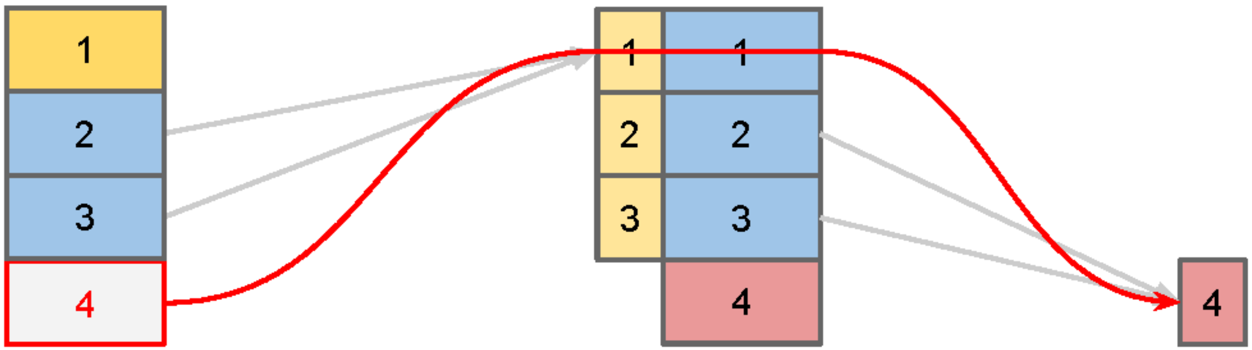
It is Great that computation is cheap!

- Proper K-fold cross validation of stacking model requires building K*(K-1) models. Below are models required for a single fold (4).

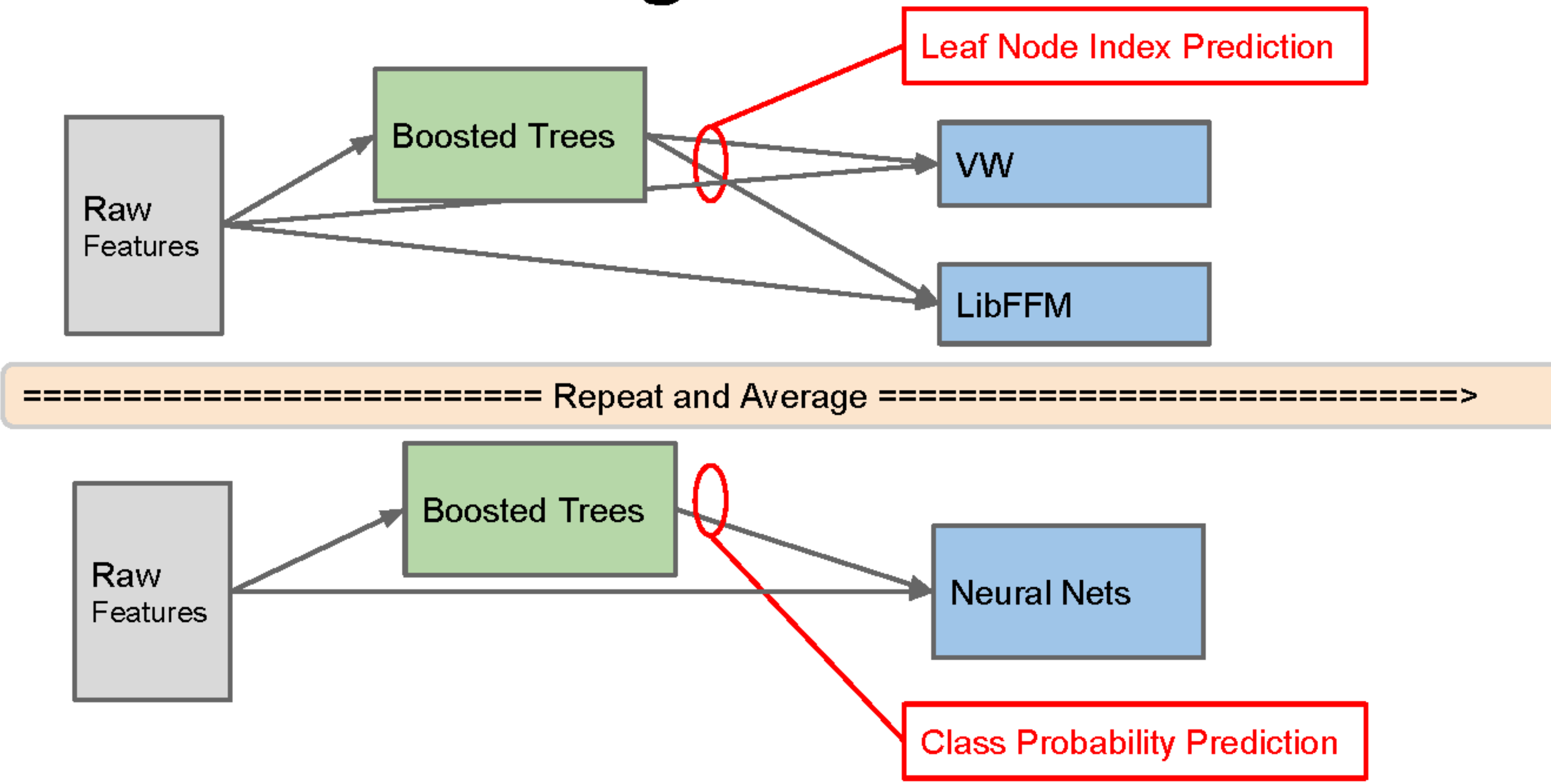


Otherwise we have indirect leakage

- Naive double cross validation -- target values in fold #4 can impact the supposed-to-be out of sample 2nd level predictions for that fold.



Put them all together



Some really impressive stuff

- For a great example of model stacking, check out
  - https://www.kaggle.com/c/otto-group-product-classification-challenge/forums/t/14335/1st-place-winner-solution-gilberto-titericz-stanislav-semenov
- Deep learning for rotation-invariant image classification by Sander Dieleman and team
  - Galaxy: http://benanne.github.io/2014/04/05/galaxy-zoo.html
  - Plankton: http://benanne.github.io/2015/03/17/plankton.html
- Winning solution to the Higgs Boson Challenge by Gabor Melis
  - https://github.com/melisgl/higgsml/blob/master/doc/model.md
- ...

The End

Thank you!