

Winning Data Science Competitions

Some (hopefully) useful pointers

Owen Zhang
Data Scientist

A plug for myself

Owen

Current

- Chief Product Officer




DataRobot

Previous


- VP, Science



MASTER  ?

1st
/221,590

916,638.7 points
Joined 4 years ago



1st/634



1st/367



2nd/1687



2nd/337



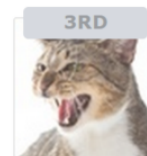
2nd/102



3rd/1568



3rd/418



3rd/215



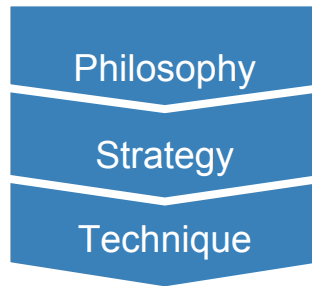
Competitions



DataRobot

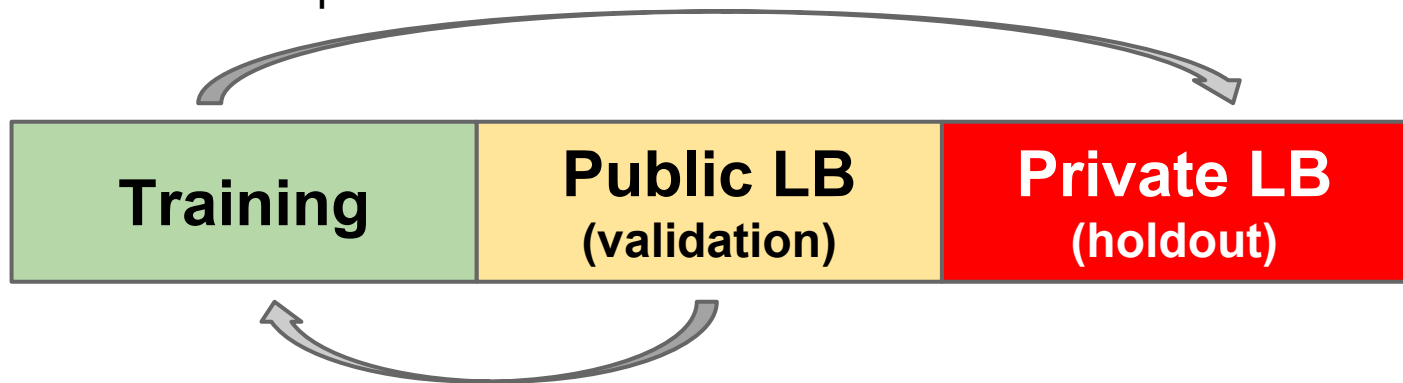
Agenda

- Structure of a Data Science Competition
- Philosophical considerations
- Sources of competitive advantage
- Some technical tips
- Two cases -- Amazon Allstate
- Apply what we learn out of competitions



Structure of a Data Science Competition

Build model using Training Data to
predict outcomes on Private LB Data



Quick but often misleading feedback

*Data Science Competitions remind us that the purpose of a predictive model is to predict on data that we have **NOT** seen.*

A little “philosophy”

- There are many ways to overfit
- Beware of “multiple comparison fallacy”
 - There is a cost in “peeking at the answer”,
 - Usually the first idea (if it works) is the best

“Think” more, “try” less

Sources of Competitive Advantage (the Secret)

- Discipline (once bitten twice shy)
 - Proper validation framework
- Effort
- (Some) Domain knowledge
- Feature engineering
- The “right” model structure
- Machine/statistical learning packages
- Coding/data manipulation efficiency
- Luck

Be Disciplined
+
Work Hard
+
Learn from
everyone
+
Luck

Technical Tricks -- GBM

- My confession: I (over)use GBM
 - When in doubt, use GBM
- GBM automatically approximate
 - Non-linear transformations
 - Subtle and deep interactions
- GBM gracefully treats missing values
- GBM is invariant to monotonic transformation of features

Technical Tricks -- GBM Tuning

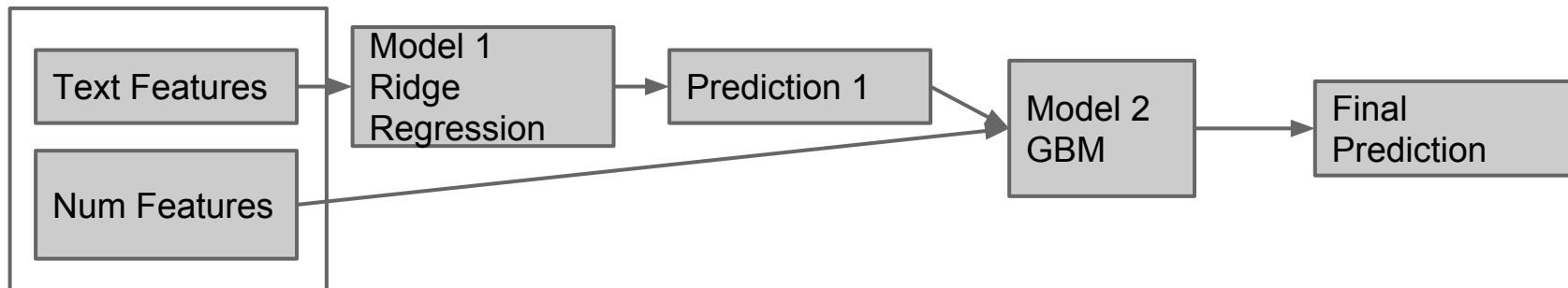
- Tuning parameters
 - Learning rate + number of trees
 - Usually small learning rate + many trees work well. I target 1000 trees and tune learning rate
 - Number of obs in leaf
 - How many obs you need to get a good mean estimate?
 - Interaction depth
 - Don't be afraid to use 10+, this is (roughly) the number of leaf nodes

Technical Tricks -- when GBM needs help

- High cardinality features
 - These are very commonly encountered -- zip code, injury type, ICD9, text, etc.
 - Convert into numerical with preprocessing -- out-of-fold average, counts, etc.
 - Use Ridge regression (or similar) and
 - use out-of-fold prediction as input to GBM
 - or blend
 - Be brave, use N-way interactions
 - I used 7-way interaction in the Amazon competition.
- GBM with out-of-fold treatment of high-cardinality feature performs very well

Technical Tricks -- Stacking

- Basic idea -- use one model's output as the next model's input



- It is NOT a good idea to use in sample prediction for stacking
 - The problem is over-fitting
 - The more “over-fit” prediction1 is , the more weight it will get in Model 2

Technical Tricks -- Stacking -- OOS / CV

- Use out of sample predictions
 - Take half of the training data to build model 1
 - Apply model 1 on the rest of the training data, use the output as input to model 2
- Use cross-validation partitioning when data limited
 - Partition training data into K partitions
 - For each of the K partition, compute “prediction 1” by building a model with OTHER partitions

Technical Tricks -- feature engineering in GBM

- GBM only APPROXIMATE interactions and non-linear transformations
- Strong interactions benefit from being explicitly defined
 - Especially ratios/sums/differences among features
- GBM cannot capture complex features such as “average sales in the previous period for this type of product”

Technical Tricks -- Glmnet

- From a methodology perspective, the opposite of GBM
- Captures (log/logistic) linear relationship
- Work with very small # of rows (a few hundred or even less)
- Complements GBM very well in a blend
- Need a lot of more work
 - missing values, outliers, transformations (log?), interactions
- The sparsity assumption -- L1 vs L2

Technical Tricks -- Text mining

- tau package in R
- Python's sklearn
- L2 penalty a must
- N-grams work well.
- Don't forget the “trivial features”: length of text, number of words, etc.
- Many “text-mining” competitions on kaggle are actually dominated by structured fields -- KDD2014

Technical Tricks -- Blending

- All models are wrong, but some are useful (George Box)
 - The hope is that they are wrong in different ways
- When in doubt, use average blender
- Beware of temptation to overfit public leaderboard
 - Use public LB + training CV
- The strongest individual model does not necessarily make the best blend
 - Sometimes intentionally built weak models are good blending candidates -- Liberty Mutual Competition

Technical Tricks -- blending continued

- Try to build “diverse” models
 - Different tools -- GBM, Glmnet, RF, SVM, etc.
 - Different model specifications -- Linear, lognormal, poisson, 2 stage, etc.
 - Different subsets of features
 - Subsampled observations
 - Weighted/unweighted
 - ...
- But, do not “peek at answers” (at least not too much)

Apply what we learn outside of competitions

- Competitions give us really good models, but we also need to
 - Select the right problem and structure it correctly
 - Find good (at least useful) data
 - Make sure models are used the right way

Competitions help us

- Understand how much “signal” exists in the data
- Identify flaws in data or data creation process
- Build **generalizable** models
- Broaden our technical horizon
- ...

Case 1 -- Amazon User Access competition

- One of the most popular competitions on Kaggle to date
 - 1687 teams
- Use anonymized features to predict if employee access request would be granted or denied
- All categorical features
 - Resource ID / Mgr ID / User ID / Dept ID ...
 - Many features have high cardinality
- But I want to use GBM

Case 1 -- Amazon User Access competition

- Encode categorical features using observation counts
 - This is even available for holdout data!
- Encode categorical features using average response
 - Average all but one (example on next slide)
 - Add noise to the training features
- Build different kind of trees + ENET
 - GBM + ERT + ENET + RF + GBM2 + ERT2
- I didn't know VW (or similar), otherwise might have got better results.
- <https://github.com/owenzhang/Kaggle-AmazonChallenge2013>

Case 1 -- Amazon User Access competition

“Leave-one-out” encoding of categorical features:

Split	User ID	Y	mean(Y)	random	Exp_UID
Training	A1	0	.667	1.05	0.70035
Training	A1	1	.333	.97	0.32301
Training	A1	1	.333	.98	0.32634
Training	A1	0	.667	1.02	0.68034
Test	A1	-	.5	1	.5
Test	A1	-	.5	1	.5
Training	A2	0			

Case 2 -- Allstate User Purchase Option Prediction

- Predict final purchased product options based on earlier transactions.
 - 7 correlated targets
- This turns out to be very difficult because:
 - The evaluation criteria is all-or-nothing: all 7 predictions need to be correct
 - The baseline “last quoted” is very hard to beat.
 - Last quoted 53.269%
 - #3 (me) : 53.713% (+0.444%)
 - #1 solution 53.743% (+0.474%)
- Key challenges -- capture correlation, and not to lose to baseline

Case 2 -- Allstate User Purchase Option Prediction

- Dependency -- Chained models
 - First build stand-alone model for F
 - Then model for G, given F
 - $F \Rightarrow G \Rightarrow B \Rightarrow A \Rightarrow C \Rightarrow E \Rightarrow D$
 - “Free models” first, “dependent” model later
 - In training time, use actual data
 - In prediction time, use most likely predicted value
- Not to lose to baseline -- 2 stage models
 - One model to predict which one to use: chained prediction, or baseline

Useful Resources

- <http://www.kaggle.com/competitions>
- <http://www.kaggle.com/forums>
- <http://statweb.stanford.edu/~tibs/ElemStatLearn/>
- <http://scikit-learn.org/>
- <http://cran.r-project.org/>
- https://github.com/JohnLangford/vowpal_wabbit/wiki
-