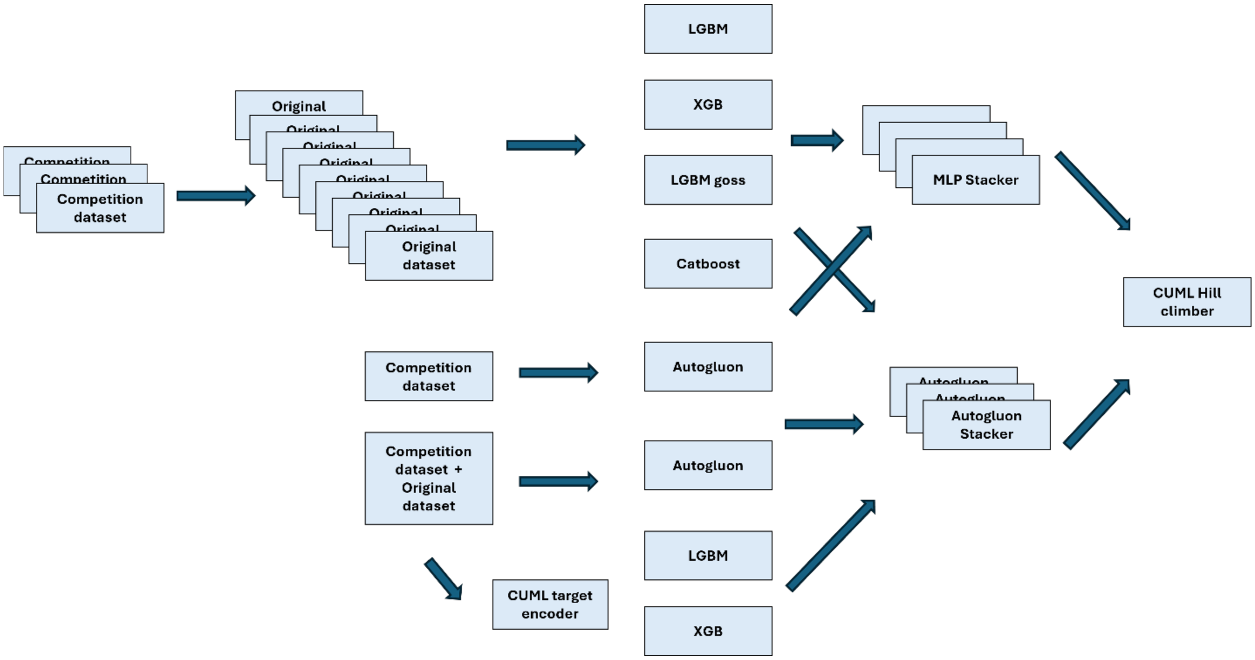
Rank 28 approach - diversity and CV prevail!

Hello all,

I am thankful to Kaggle for a reasonably good playground episode the past month. I am sure a lot of us worked with the MAP@3 metric for the first time and I think the experience was quite good! I wish to extend sincere thanks to the forum contributors for their generosity as well and congratulate the high scoring solution participants as well.

My approach herewith tried to infuse as much diversity as feasible, keeping CV-LB relation in mind. Diversity can be ensued with features and model choices and I chose both these elements for my pipeline. Details are illustrated in the figure below-



Feature Engineering

As indicated in the figure, I chose to ensure-

- No feature engineering and just the training features as is, but with string datatype
- Multiple train + multiple original datasets appended, ranging from 1 train, 0 original to 2 trains + 19 originals
- Limited experiments with bigrams + trigrams and target encoder using CuML

Model training

- I used a 10-fold cv scheme as below-
  KFold(10, random_state = 42, shuffle = True)

- I trained a lot of varied boosted tree models across feature set options and built more than 70 single models with varied feature sets
- I observed that a single xgboost model with depth = 8, learning_rate = 0.01 on category represented dataset performed the best. *The model elicited the best cv score when I appended 1 training data + 6-8 original datasets*
- Most of the models performed poorly with early stopping with auc/ logloss proxy metrics. Using a higher estimator number (aka 10000) helped a lot, albeit with a longer training time
- Catboost was the worst performer with training features, but performed very well as a stacker model
- LightGBM and LightGBM goss performed moderately well and provided needed diversity to the ensemble
- I also used an Autogluon pipeline with 1 training + 1 original data and secured a cv score of 0.345. I used a few single models from this pipeline for diversity, despite its poor individual performance

Ensemble

This did not yield a significant benefit here. I choose the below approaches with varying elements of success-

- Autogluon stacker - built 3 autogluon staker models using different model choices
- Torch NN - built 4 staker models using different model choices
- Finally, I chose to blend the ensemble models and all the single models with a CuML Hill Climber. It is the same as any other hill climber but it uses cupy and cudf instead of pandas and numpy. I have a private setup for this and often use it across competitions

CV details

I shall highlight a few CV score ranges across my models as below-

Model type/ algorithm	**CV score**	**Selected for blending**
Starter public work	0.355	N
Catboost	0.325- 0.33	Y
LightGBM	0.34 - 0.375	Y
LightGBM goss	0.335 - 0.3725	Y
XgBoost	0.34 - 0.3795	Y
Torch NN	0.31-0.32	N
Autogluon - direct application	0.345	Y
Autogluon stacker model	0.3810 - 0.3812	Y

Model type/ algorithm	CV score	Selected for blending
Torch NN model	0.3790 - 0.3805	Y
Logistic blend	0.3815-0.3818	N
CuML Hill Climber blend	0.3820 - 0.3830	Y

GPUs used

I used the below GPU suite for the model training -

Model type/ algorithm	GPU
Catboost	A5000
LightGBM	A5000
LightGBM goss	A5000
XgBoost	A6000 Ada

Model type/ algorithm	**GPU**
Torch NN	A5000
Autogluon - direct application	L4 - Google Colab
Autogluon stacker model	L4 - Google Colab
Torch NN model	L4 - Google Colab
Logistic blend	None
CuML Hill Climber blend	4090 local GPU

Key learnings

- Stick to your CV, don't chase non-CV backed blends and stay diligent and success will be yours!

Concluding remarks

Wishing you the best for the upcoming competitions and in your professional life also!
Happy learning and best regards!

References

- https://www.kaggle.com/datasets/ravi20076/playgrounds5e6models -- trained models

- https://www.kaggle.com/code/ravi20076/playgrounds5e6-public-baseline-v1 -- baseline work, most private models were trained with this code

Regards,
Ravi Ramakrishnan