Ensemble Modeling

Toronto Data Mining Forum | November 2017

Helen Ngo

Agenda

- Introductions
- Why Ensemble Models?
- Simple & Complex ensembles
- Thoughts: Post-real-life Experimentation
- Downsides of Ensembles
- A Model-agnostic Methodology for Interpretability

Hello!

About Me

- data scientist
- telecommunications industry
- mathematics background
- coffee enthusiast
- women in STEM

www.linkedin.com/in/helen-ngo/

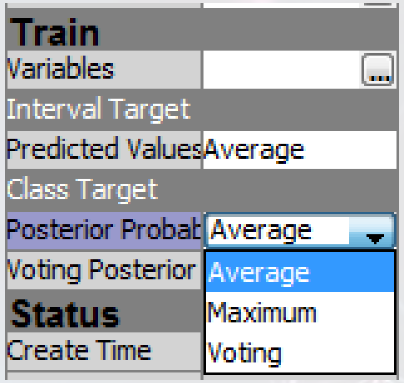
helen.ngo14@gmail.com

What’s more accurate than one great model?

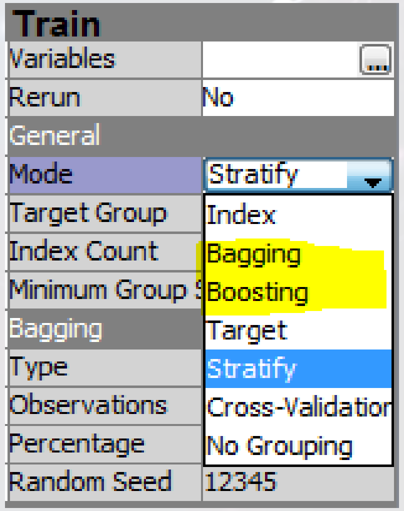
- Sometimes, a lot of not-too-shabby models
- **Ensemble modeling**: learning algorithms to combine classifiers by weighting their predictions
- Lots of models; diverse algorithms; uncorrelated predictions
  - You’re more likely to be right most of the time
  - Ensembles only more accurate than individual models if the models disagree with each other

The simple ones

- Simple average, voting
  - Ensemble node in Enterprise Miner
- Bagging: sample with replacement
  - Random forests: random subset of features for many trees trained in parallel
- Boosting: iteratively reweight misclassified observations in training
  - XGBoost, AdaBoost



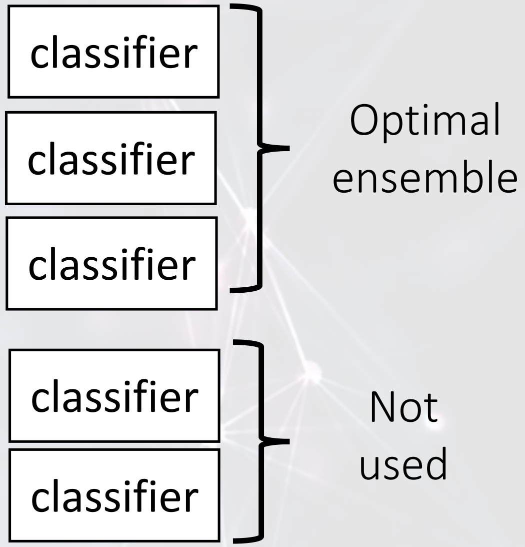
Train	
Variables	
Interval Target	
Predicted Values	Average
Class Target	
Posterior Probat	Average
Voting Posterior	
Status	
Create Time	



Train	
Variables	
Rerun	No
General	
Mode	Stratify
Target Group	
Index Count	
Minimum Group S	
Bagging	
Type	
Observations	
Percentage	
Random Seed	12345

Slightly more sophisticated

- Top-T: for N models and t <= N,
  - Take best T models according to your accuracy measure of choice
  - Use validation data to select optimal value for T
  - More is not always better!



classifier

classifier

classifier

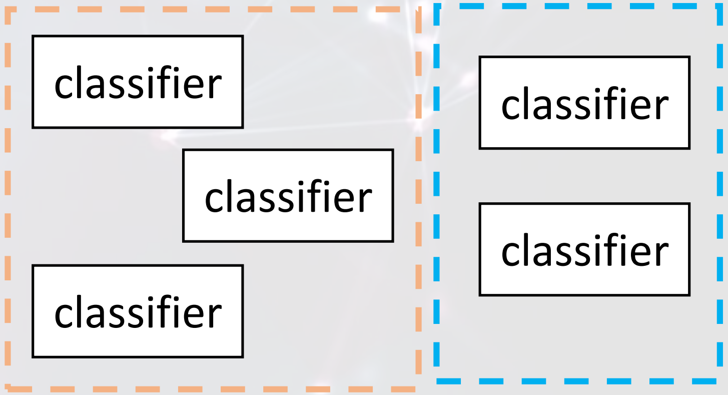
Optimal ensemble

classifier

classifier

Not used

- Unsupervised cluster ensembling
  - Use PCA to assign probabilities from N models to clusters based on original features
  - Use the models in cluster with top-T method



classifier

classifier

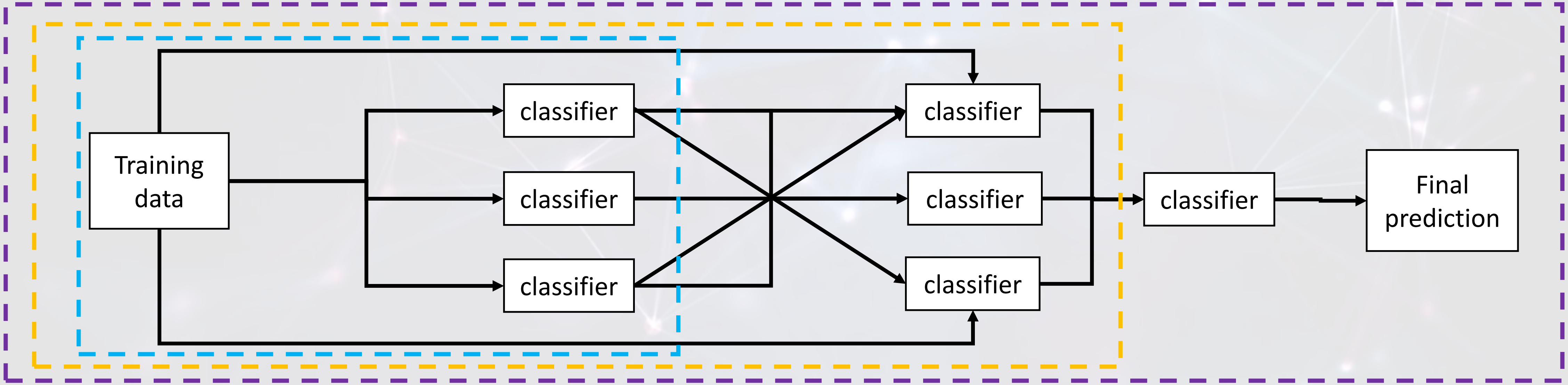
classifier

classifier

classifier

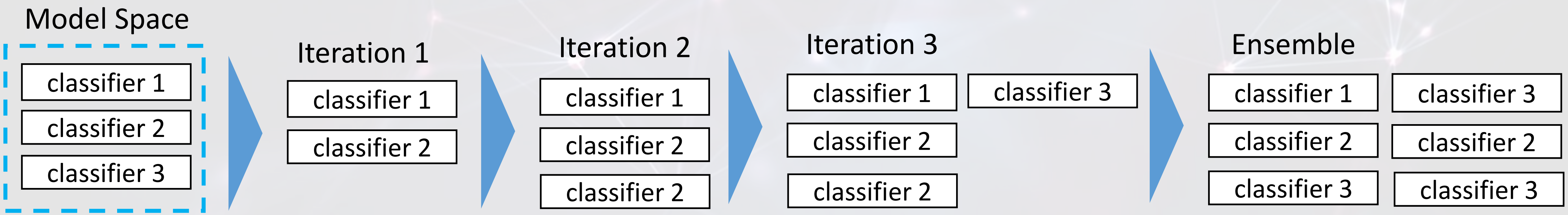
Popularized by our favourite data science competition

- Stacking & blending
  - Use posterior probabilities from trained models as numerical inputs to model original target variable
  - Can have several stacked layers
  - Stacking functions can be any supervised learning method

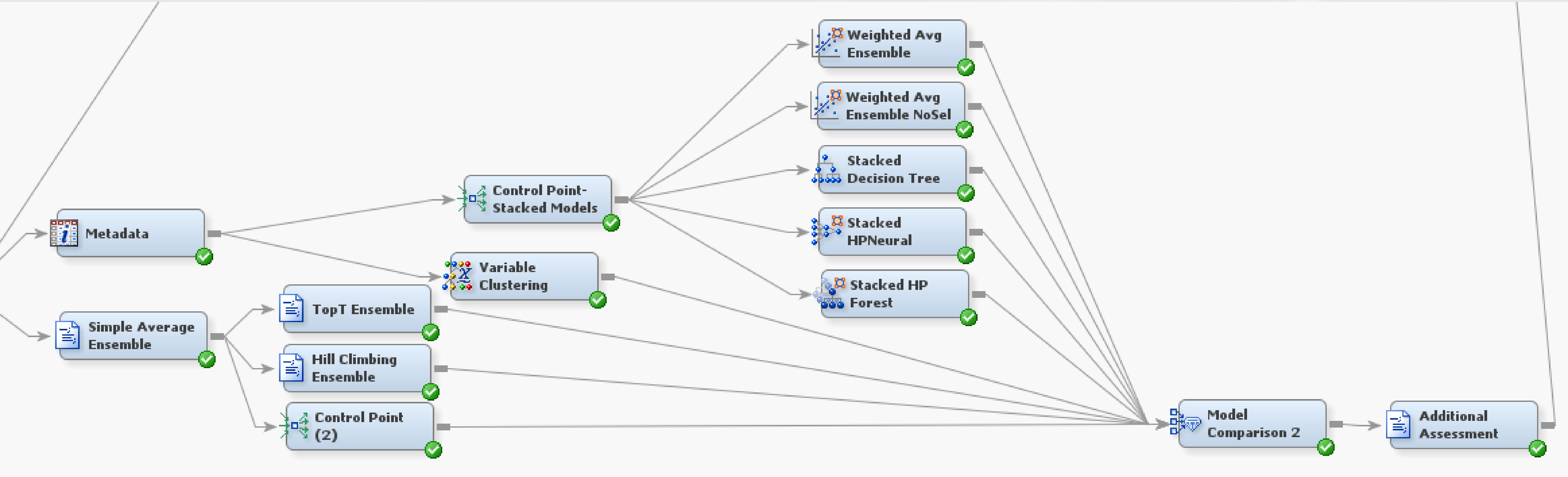


On the shoulders of giants

- Hill climbing
  - Rank models by some accuracy measure and calculate the incremental ensemble performance by adding one at a time
  - Greedy algorithm to choose next model to add
  - Final ensemble chosen based on overall accuracy metric
  - Models can be added multiple times and weighted differently; powerful models can be added many times



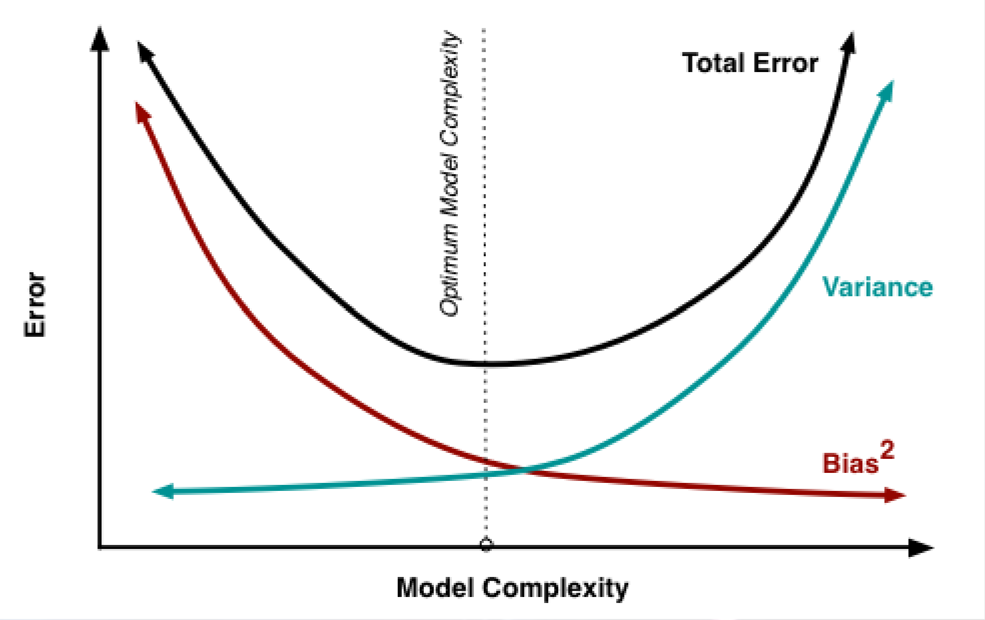
In Enterprise Miner



Source: Wendy Czika and the SAS team
https://github.com/sassoftware/dm-flow/tree/master/EnsembleModeling

The reason why ensembles work

- Bias-variance error decomposition
  - Bias (underfitting) & variance (overfitting) traded off
  - More complexity → more overfitting
- What if we have a bunch of low-bias, high-variance models?
  - Ensemble them all
    → same bias, lower variance
  - Total error is lower!
  - Less correlation is better (higher reduction in variance)



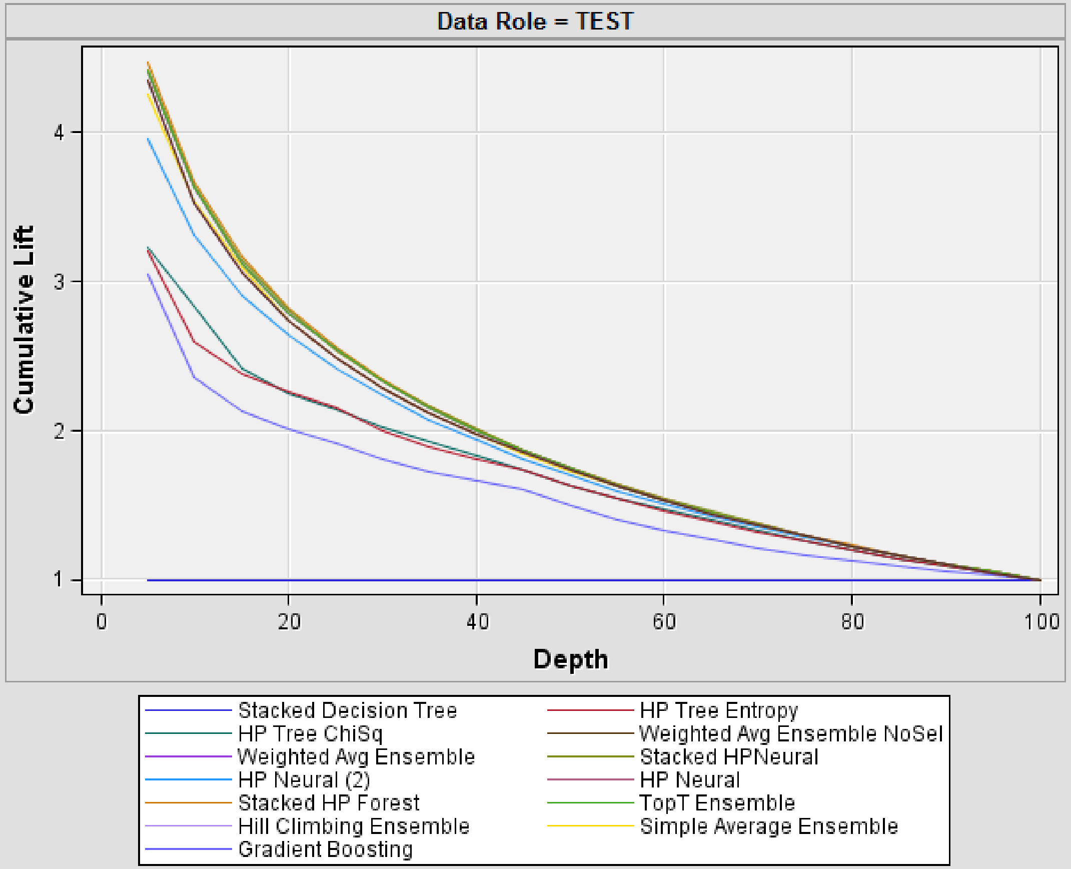
Source:
http://scott.fortmann-roe.com/docs/BiasVariance.html

Thoughts: post-real-life experimentation

- Success! Overcame “noisy” marketing data with ensembles

- Best ensembles created out of weak learners without many transformations

- Ensembles don’t necessarily win over simpler models—depends on your use case



The best ensemble models outperform the best non-ensemble significantly – 3.7x vs. 3.4x lift

Downsides of ensembles

- Search space for best ensemble is large
  - Time-consuming to train
- Feature impact on final score is unclear
- Lack of explanation for customer-facing decisions
  - Hard to sell to a business decision maker

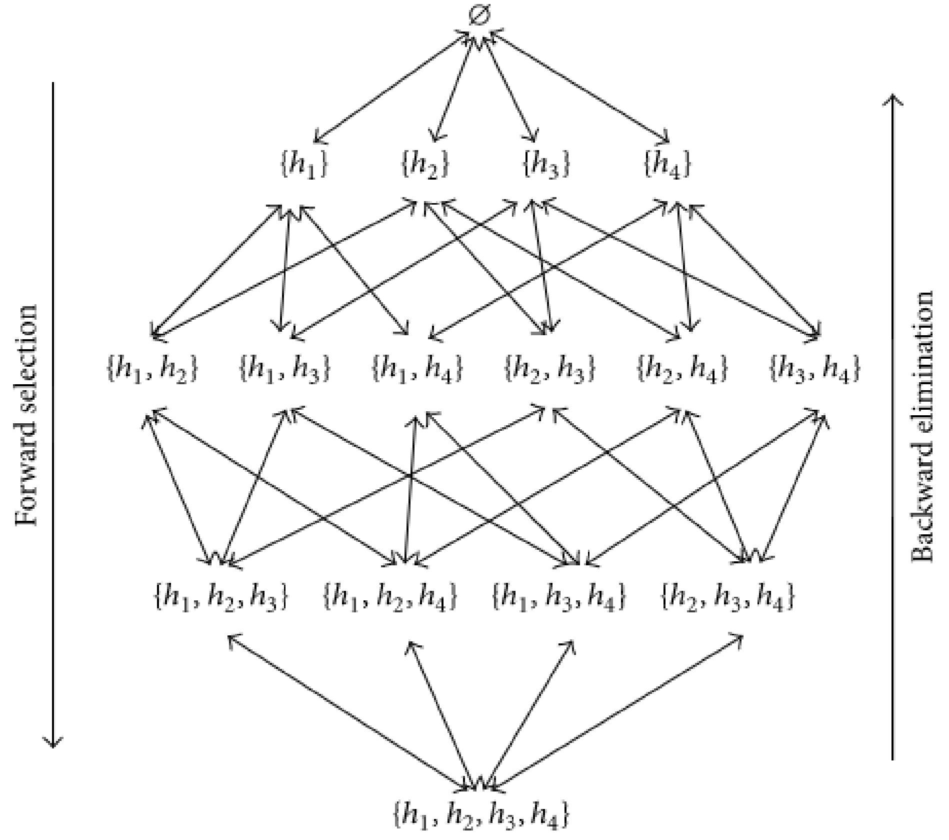


Figure 1: The search space of DHCEP methods for an ensemble of 4 models.

Source: https://www.hindawi.com/journals/mpe/2016/3845131/fig1/

Model Contributions: Peeking Under the Hood

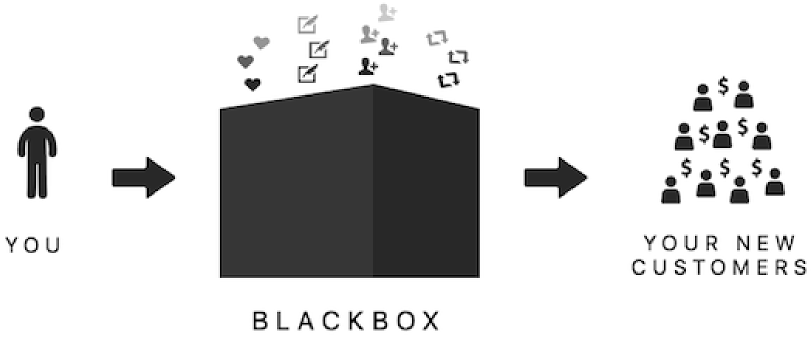
Toronto Data Mining Forum | November 2017

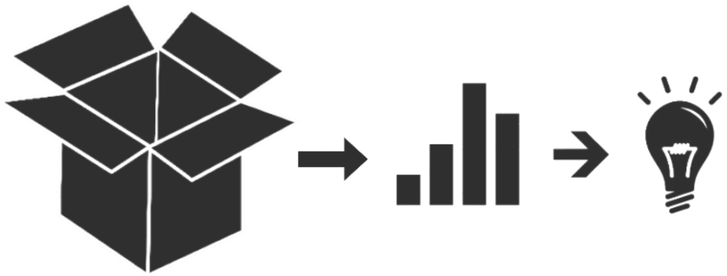
Adrian Muresan

Agenda

Adrian Muresan	
Current Role	- Senior Analyst – Business Intelligence - Started at Bell in January 2015
Education	- MSc Mathematics - MSc Computing
BI Team	- Modeling and Segmentation
Main Responsibilities	- Lead a Predictive Modeling Team

Topic	Page
Why Look Into Models	3
How To Look Into Models	4
Looking Into Models	5-11
Performance Stability	5
Variable Impact Analysis	6
Lift Drop Analysis	7
Correlation Analysis	8
Variable Level Historical Trends	9

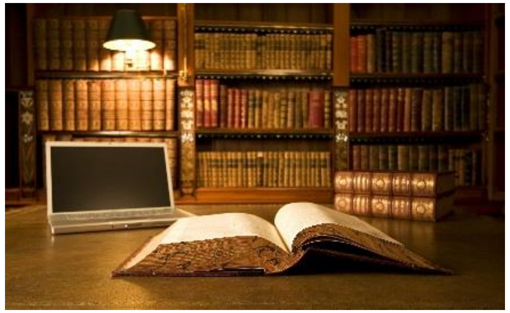




Why Look Under The Hood

Legislation:

- Under the new EU General Data Protection Regulation (GDPR) set to come into effect in 2018 all EU citizens will have a “right to explanation” from vendors using their information, even if they are based outside the EU



Promote Use:

- Cool ML products are redundant if they’re not adopted, and fear of the unknown outside of the ML/AI community keeps marketers from trusting ad using your results

Ethics:

- We must be careful in the way we allow data to influence our actions to ensure we are making decisions that are ethical. The book: Weapons of Math Destruction: The Dark Side of Big Data – O’Neil (2016) expounds upon this topic very well

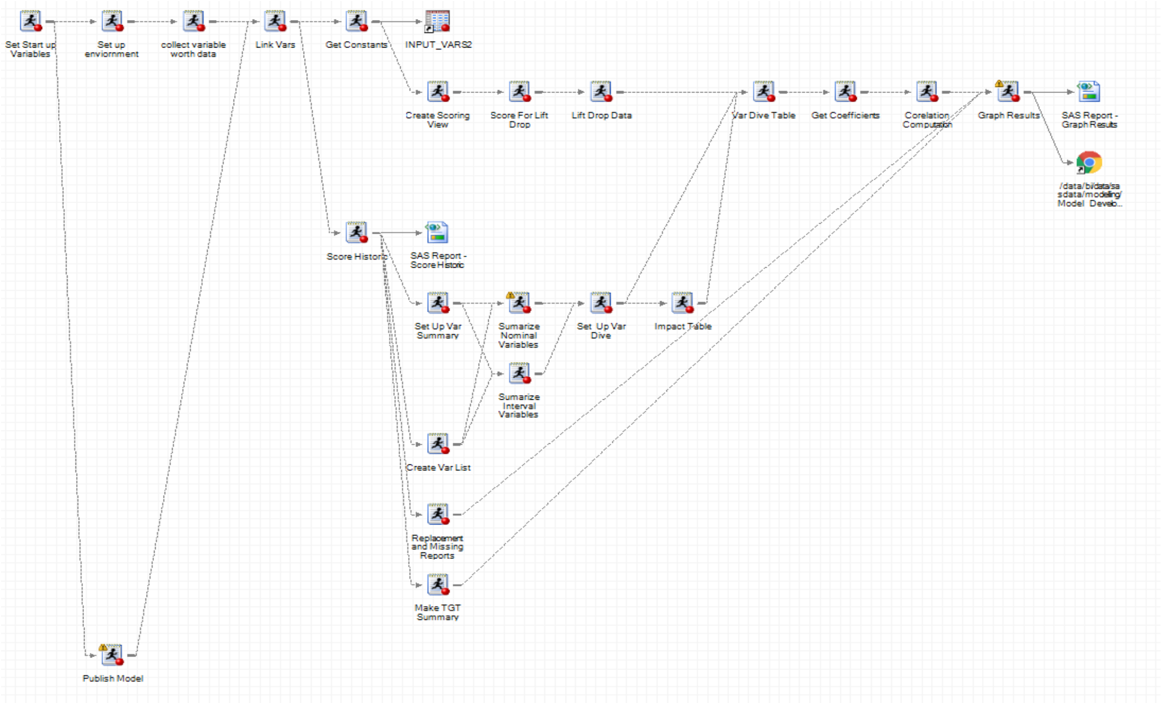
Accuracy:

- Sometimes historical performance may look stable but this does not always imply strong robust results going forward

How To Look Into Models

What Is It?

- The metrics discussed here are generated by a SAS EG program
- This program is set up to take environmental variables in it’s first step then run automatically for any given model



Prerequisites :

- Needs a method of scoring a model (preferably in data base for speed and I/O optimization)
- Needs metadata tables describing the input and output variables of the model
- For Models with coefficients needs metadata about which variables are assigned which coefficients

Pros:

- Greatly reduces model iteration time
- Provides quick and much more in depth validation that is available within EM
- The process does not rely on the model having been built in EM, only assumes that you can score it and that the output results have a certain format
- Outputs a HTML documentation file which can be shared with others, or used to compare old versions
- Largely Model Agnostic

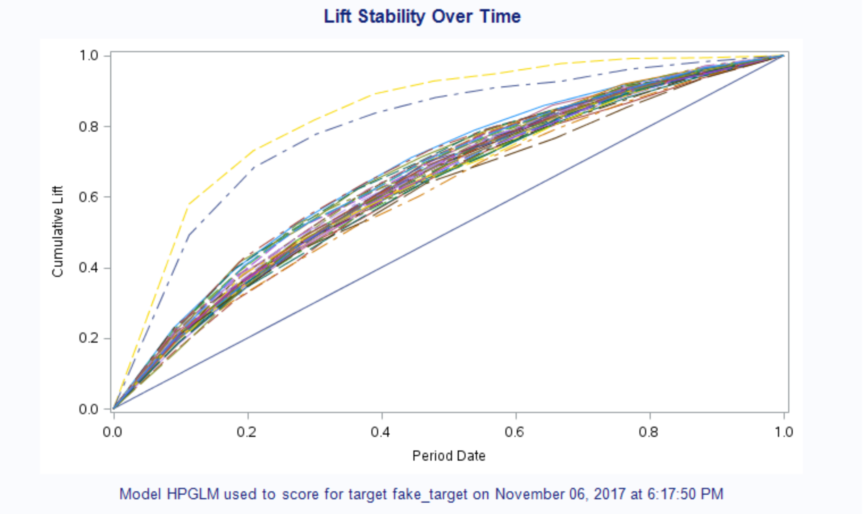
Cons:

- Some of the processes take a great deal of processing power
- As the statistical rigor of your tests increase, the time required to run also increases
- These could be alleviated by transitioning to a distributed network for scoring

Performance Stability

Lift Stability by Decile:

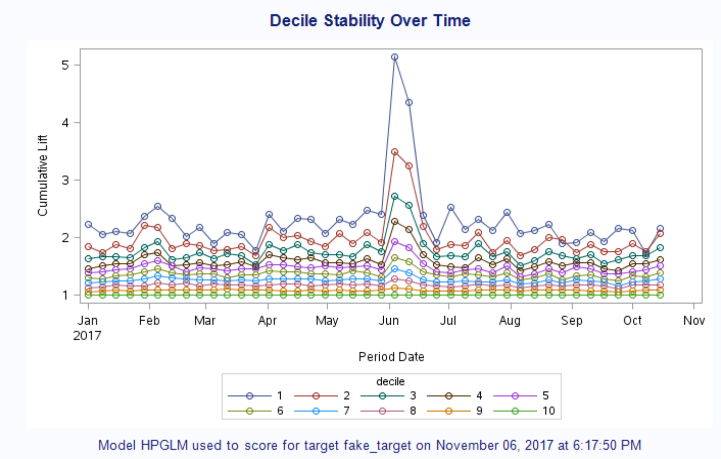
- Shows how stable (or not) our performance is.
- Ideally each line in the graph to the right would be flat.



Model HPGLM used to score for target fake_target on November 06, 2017 at 6:17:50 PM

Target Rate Over Time:

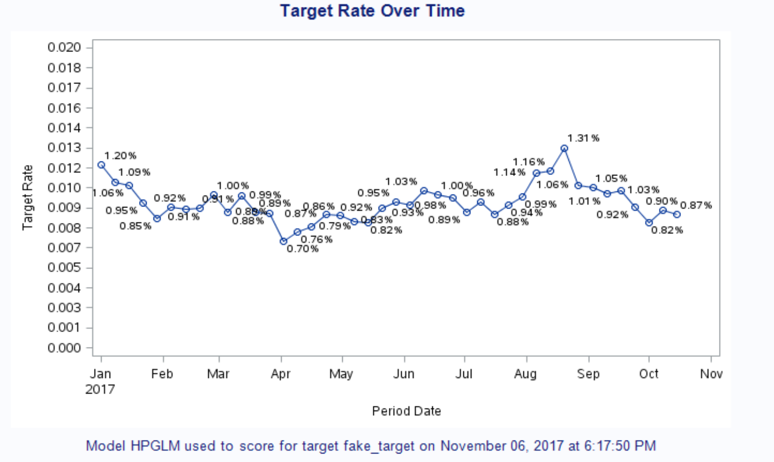
- Independent of your model results, but may explain deviations in your lift, it may also highlight points in time when there are data issues to consider.



Model HPGLM used to score for target fake_target on November 06, 2017 at 6:17:50 PM

Cumulative Response Stability:

- A different way of looking at the same stability problem.
- Ideally the various response curves to be practically indistinguishable form each other.



Model HPGLM used to score for target fake_target on November 06, 2017 at 6:17:50 PM

Variable Impact

Goal:

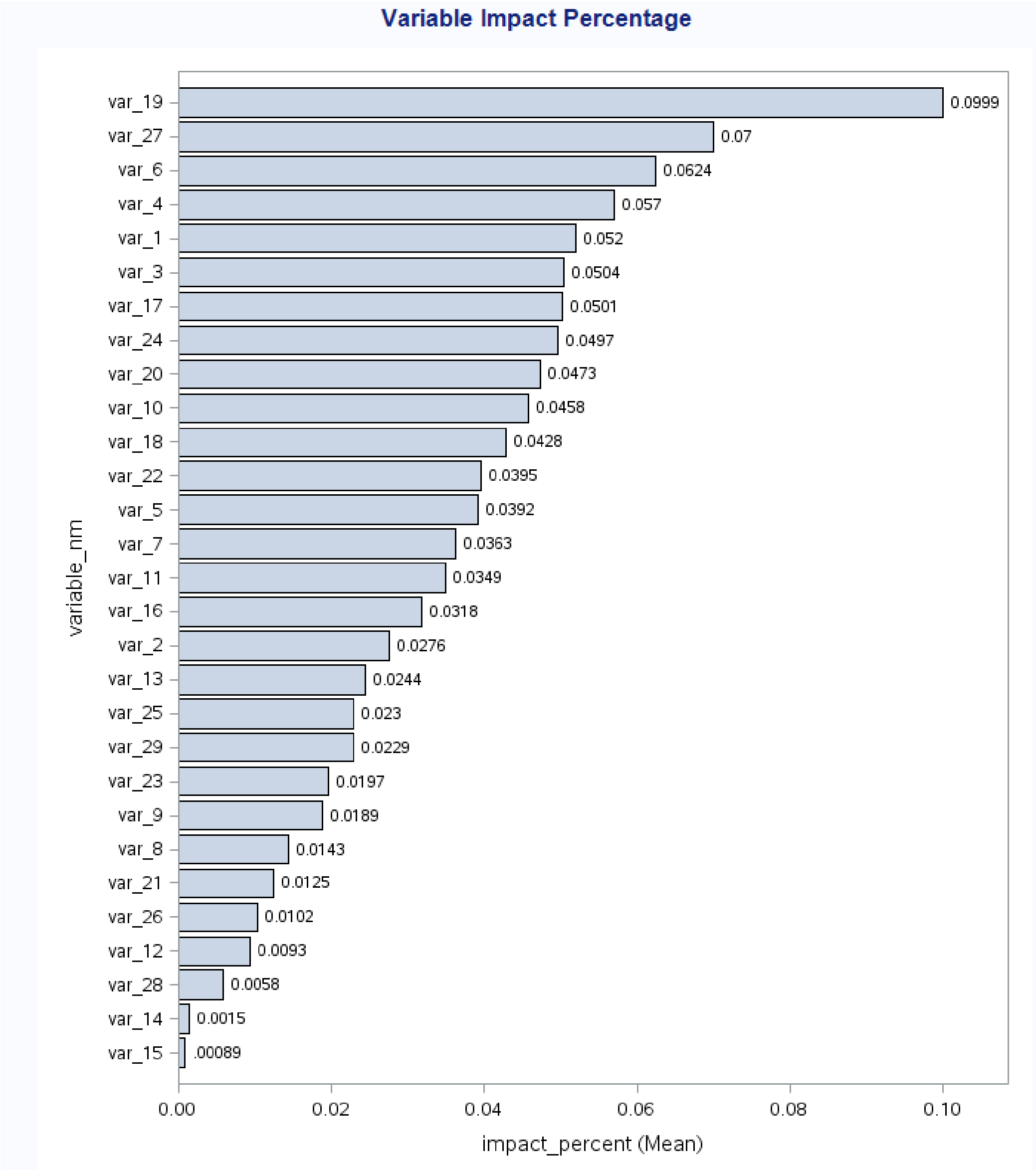
- Measure how much the variation of a variable impacts the score of the observation

What To Look For:

- Make sure that no single variable is responsible for too much of the variation in scores.
- Earmark those variables with higher impact for closer scrutiny making sure that they are not overfit.

Methodology:

- See what the average scores are based on different levels of each variable and see how much variation there is between levels
- Note that some levels will have vastly different number of observations and this needs to be accounted for when computing this metric



Lift Drop Percentage

Goal:

- Measure how much of the lift the model sees is being provided by a single variable

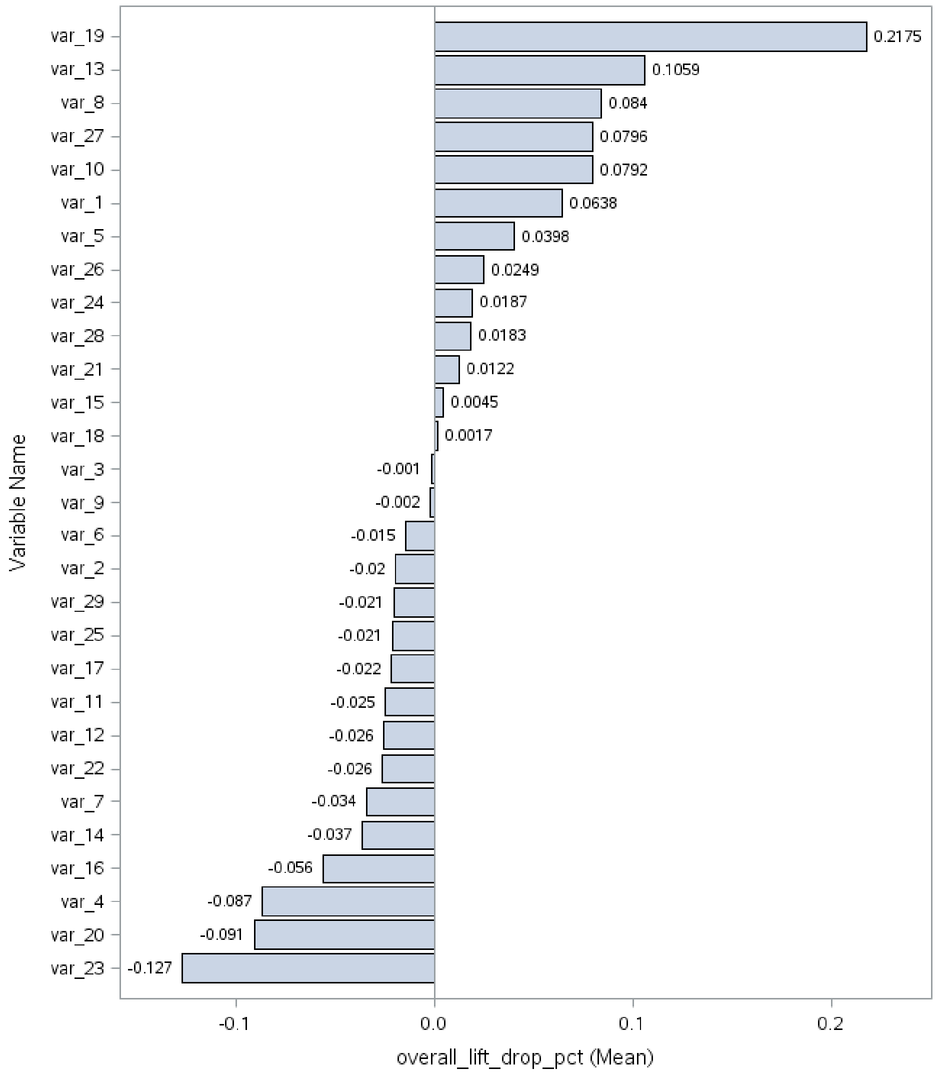
What To Look For:

- Ensure that the entire lift is not coming form a single variable (or even a small cluster)
- Ensure that the top contributors are not overly correlated or in some ways proxies for each other
- Earmark those variables with higher contributions to make sure they are not leading indicators for our target

Methodology:

- Rescore the model while fixing the variable you are testing and compare results
- Note that for statistical significance (particularly with non linear models) do this with multiple periods and multiple values

Lift Drop Percentage



Model HPGLM used to score for target fake_target on November 06, 2017 at 6:51:53 PM

Correlation to Score

Goal:

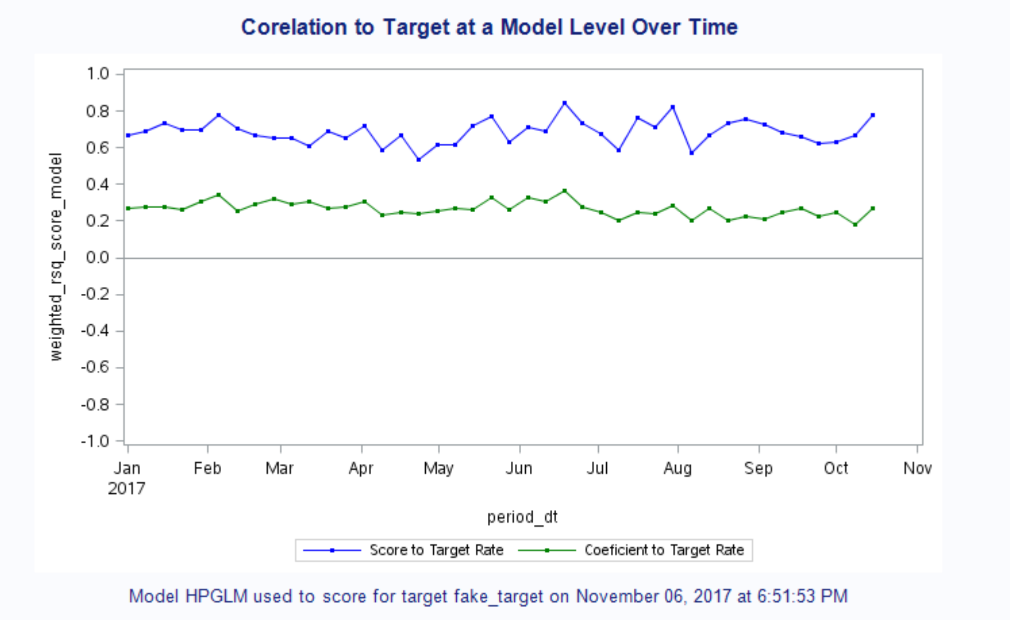
- Ensure that the average score produced at different levels of our variables correlate to the target rate on that level

What To Look For:

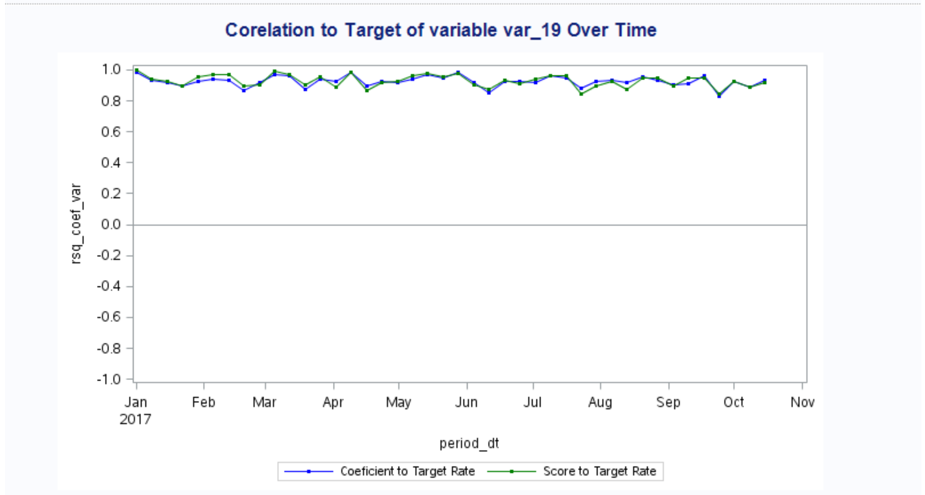
- Both that the correlation between the score and the target rate is high, and that it is stable
- For models where that have coefficients make sure the coefficients are not more correlated than the scores, this implies overfitting

Methodology:

- Proc corr works well for this
- Make sure to take into account the number of observations in each level
- Optionally weight these results by the variable impact or lift drop metrics discussed previously

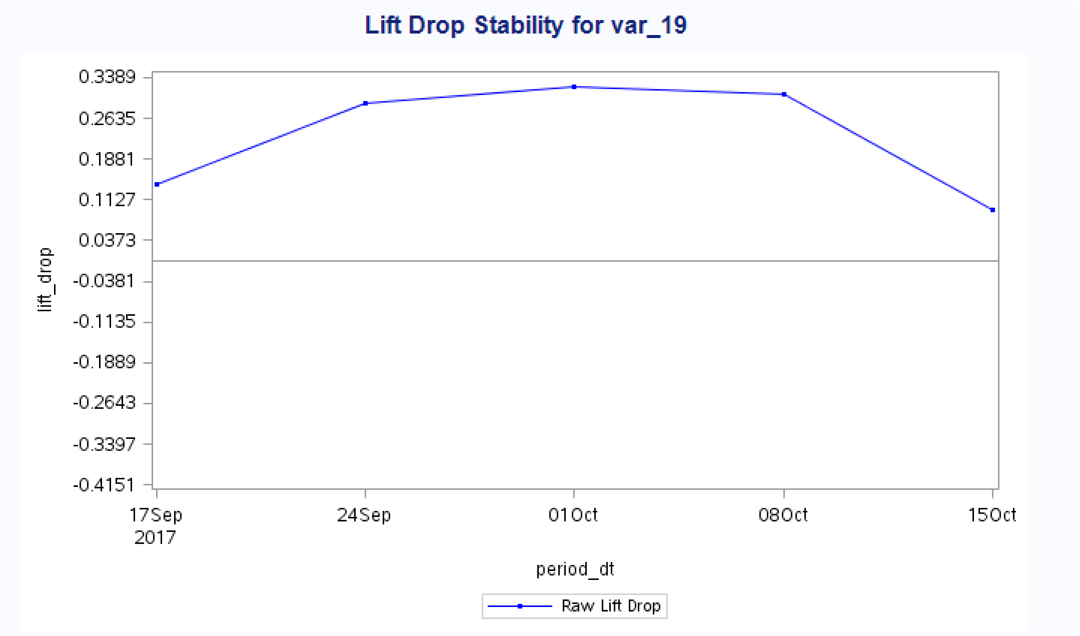


Variable Level Historical Trends



Historical Correlation:

- Very similar to the correlation analysis at the model level, but this can also help us identify which variable is most to blame for drops in correlation/lift



Historical Lift Drop:

- Ensure that the lift drop metric is stable over time and does not spike at certain points. If there is a particular time when this spiked, look deeper into that period and see what was different.

Variable Level Historical Trends

Goal:

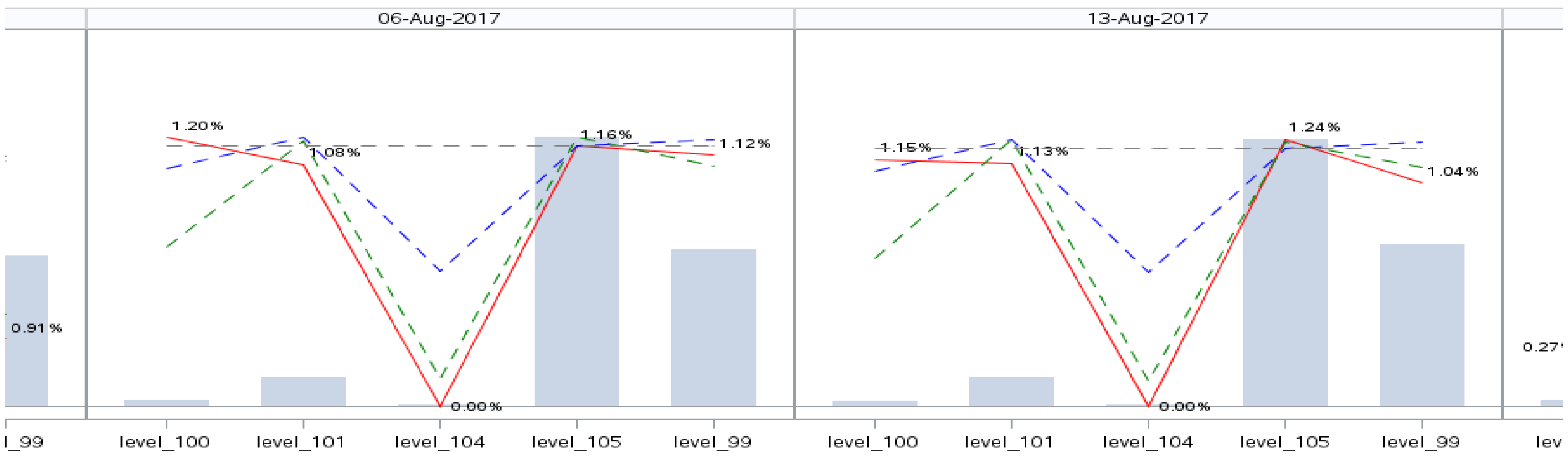
- Make sure that the scores generated are fitting the target rate, not only on an average level, but also period by period.

What To Look For:

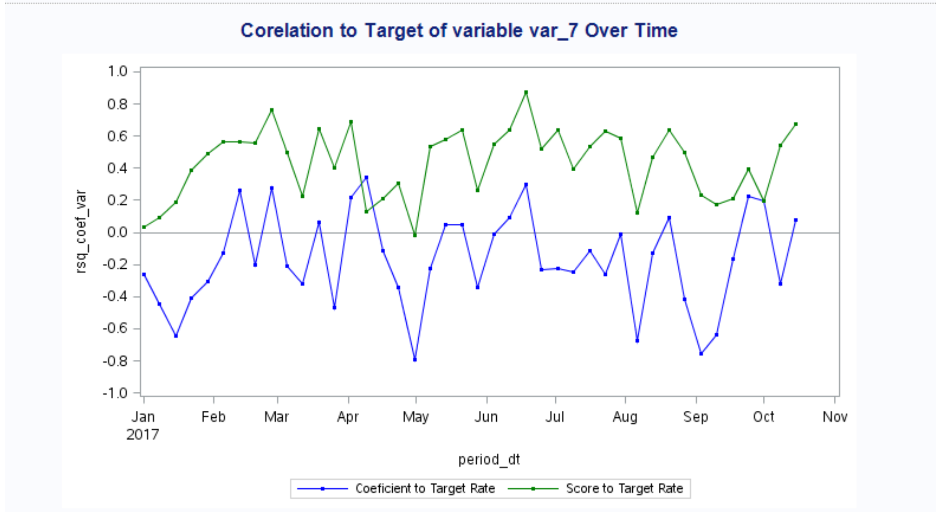
- Strong correlation between the target rate and the average score predicted
- For models that have coefficients coefficients, make sure the coefficients are also correlated, when the score is correlated but the coefficient is not, this indicates that the variable is correcting for another misattributed weight
- Check to ensure the distribution of observations by level is consistent over time

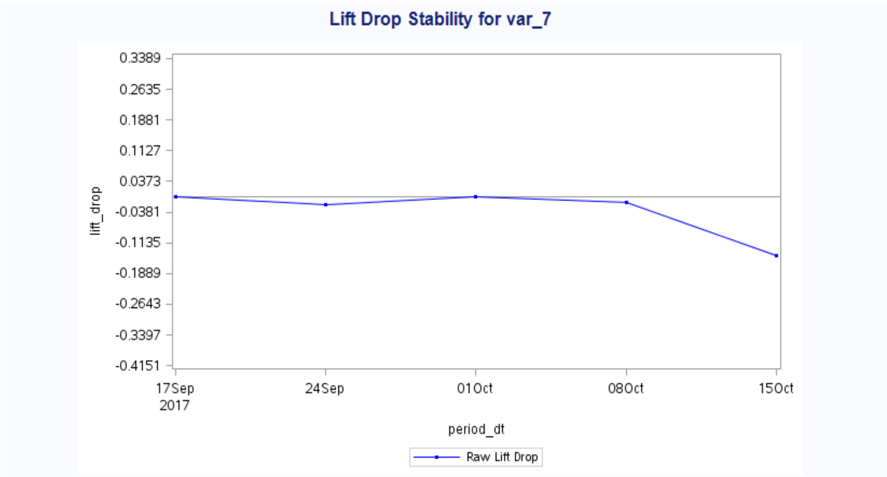
Methodology:

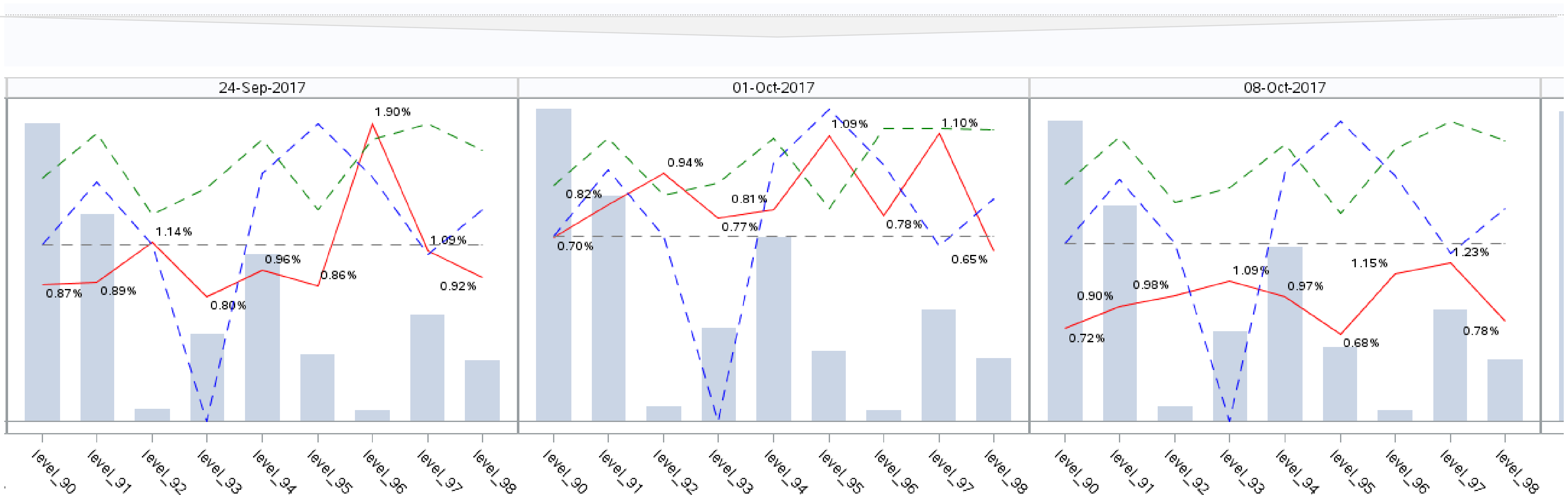
- Use the metadata tables that EM produces as well as in database scoring to produce the average score metrics.



Use Case


Corelation to Target of variable var_7 Over Time


Lift Drop Stability for var_7

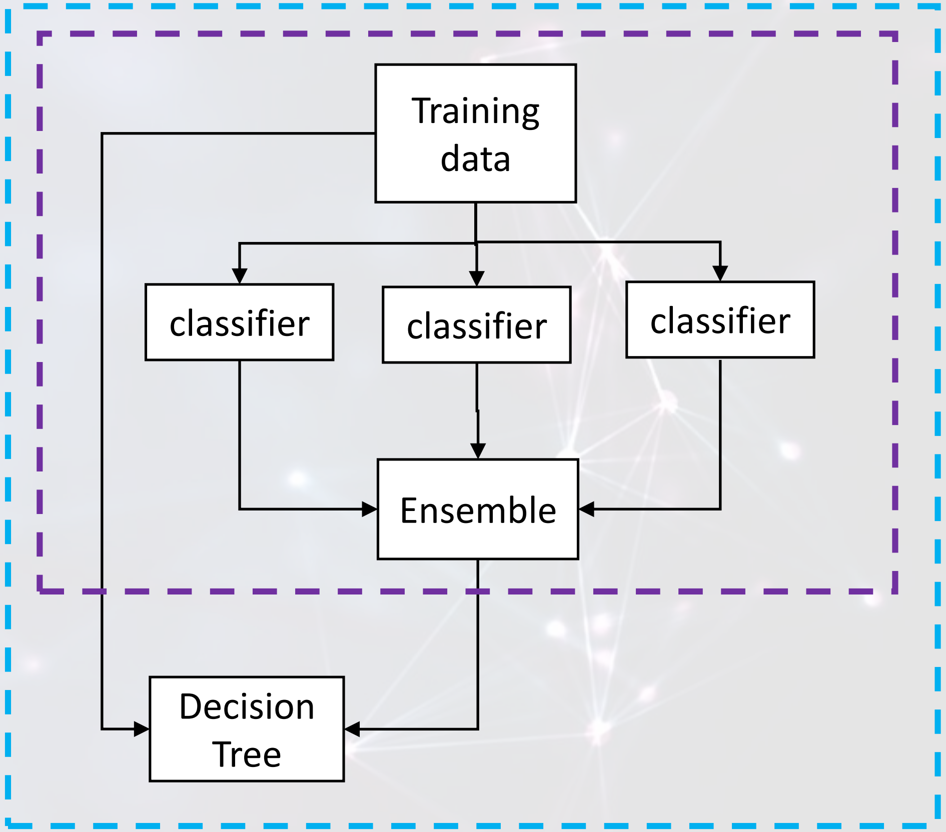


Questions?

Appendix

Metadata Node: Change the original predictors back to inputs.

- Attach a Metadata node to the winning model node
- Click “Train” under the Variables section
- Change the Role of the target_ind to be REJECTED
- Change the Role of the Event Probability to be TARGET
- This can alternatively be done via a SAS code node



Decision Tree: Use a tree node to model the scores.

- Change the Maximum Branch to be > 2 (the default is a binary tree)
- This tree should be maximally overfit as its only purpose is to describe the behaviour of the ensemble model
- The non-linear nature of the decision tree results in the best fit for the interactions as opposed to a Regression or GLM