The Unreasonable Effectiveness of Additive Models

Building high-performance, trustworthy and insightful Machine Learning Models

Michel Friesenhahn, Ondrej Slama, Kenta Yoshida

In these slides we will discuss principles and insights into building effective machine learning models for tabular datasets. This will include defining “tabular data” as well as the challenging notions of what trustworthy and insightful machine learning means. Consistent with claims from researchers like Cynthia Rudin, we show that state-of-the-art performance is usually achievable, or very nearly so, with intrinsically interpretable prediction algorithms.

What is Tabular Data?

Not precisely defined, but roughly …

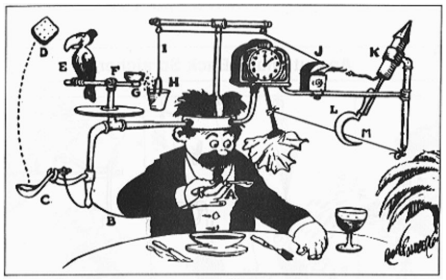
- **Ceteris Paribus (defn):** Latin phrase meaning **"holding other things constant"**
- **Tabular Data Condition:** Dataset where each feature has meaningful **ceteris paribus relationships with the outcome**
- **Examples:** disease severity features are tabular data, pixel intensities are not
- Can be assessed with **Ceteris Paribus Plots**. Tabular data models have "well behaved" ceteris paribus plots.

Tabular data is not precisely defined. But an intuitive and rough definition uses the concept of ceteris paribus: a Latin phrase meaning "holding other things constant". Then the tabular data condition is a dataset where there is inherent interest in each model input feature, and those input features all have a meaningful ceteris paribus relationships with the outcome. Later we will introduce ceteris paribus plots to further clarify this concept. Tabular data models will have "well behaved" ceteris paribus plots. Examples of tabular data features include disease severity, lab results, demographic characteristics, and imaging summary measures. Raw image files containing pixel intensities or free form text data are not usually considered tabular features.

Most of us are drawn to COMPLEXITY

When we started in machine learning (a long long time ago) we had very different ideas about predictive modeling!

- The world is complex, hence the most accurate ML models should be too
- For intrinsic interpretability, decision trees or other logical models are the way to go
- **Additive Models** are somewhat **less interpretable than Decision Trees** and generally **less accurate than complex Blackbox Models**



Since the world is complex, it is very natural to assume that the most accurate machine learning models are also complex. Furthermore, if we desire intrinsic intepretability, the best models are decision trees or other logical models. Additive models are often considered less interpretable than logical models and less accurate than complex blackbox models.

Conventional Wisdom on Interpretability: Logical Models vs Tree Ensembles

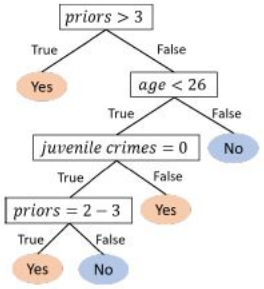
From Leo Breiman's Two Cultures Paper:

- On interpretability, trees rate an A+
- So forests are A+ predictors. But their mechanism for producing a prediction is difficult to understand. Trying to delve into the tangled web that generated a plurality vote from 100 trees is a Herculean task. **So on interpretability, they rate an F**

The specific conventional wisdom on interpretability of logical models vs tree ensembles is nicely described by Leo Breiman in his famous Two Cultures paper. Logical models such as trees rate an A+ and blackbox models such as tree forests rate an F. In this presentation we re-examine these assumptions to see if they continue to hold up. There may be a few surprises in store!

What about Logical Models?

Decision Tree



(a)

Decision List

if $(age < 26)$ **and** $(priors = 2-3)$ **then predict** yes
else if $(juvenile\ crimes = 0)$ **and** $(priors < 3)$ **then predict** no
else predict yes

(b)

Decision Set

if $(priors > 3)$ **and** $(age < 21)$ **then predict** yes
if $(juvenile\ crimes > 0)$ **and** $(prior > 3)$ **then predict** yes
if $(age < 23)$ **and** $(prior = 2-3)$ **then predict** yes
else predict no

(c)

Figure 2: Predicting which individuals are arrested within two years of release by a decision tree (a), a decision list (b), and a decision set (c). The dataset used here is the ProPublica recidivism dataset (Angwin et al., 2016)

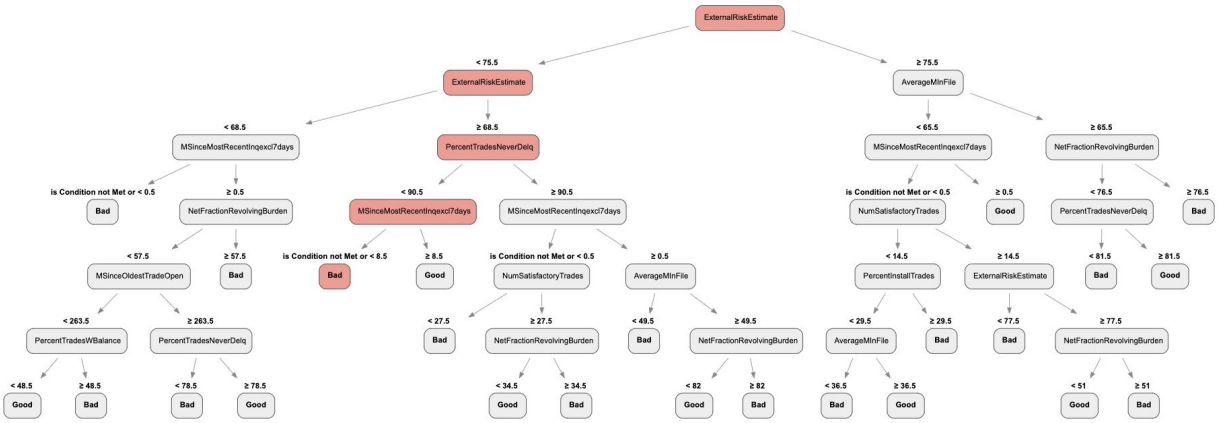
Interpretable Principles and 10 Grand Challenges, Cynthia Rudin

Logical models are widely considered highly interpetable

Let's start with logical models. There are a number of variants, including decision trees, decision lists, and decision sets. For more information see the nice summary paper: Interpretable Principles and 10 Grand Challenges, by Cynthis Rudin. These are widely considered to represent the apex of intrinsic interpretability.

Must Be Sparse or Features Become Hopelessly **Entangled**!

A Loan Default Example



https://www.interpretable.ai/solutions/loan-default-risk/

However, as many have noted, logical models these must be very sparse in the number of input features or their impacts on predicting outputs become hopelessly entangled! Here we see a number of features appearing multiple times in completely different parts of the tree. This makes it difficult to assess their net effect on model outputs.

CORELS Recidivism Decision List

Table 1 | Machine learning model from the CORELS algorithm

IF	age between 18–20 and sex is male	THEN predict arrest (within 2 years)
ELSE IF	age between 21–23 and 2–3 prior offences	THEN predict arrest
ELSE IF	more than three priors	THEN predict arrest
ELSE	predict no arrest	

This model from ref. [39] is the minimizer of a special case of equation (1) discussed later in the challenges section. CORELS' code is open source and publicly available at http://corels.eecs.harvard.edu/, along with the data from Florida needed to produce this model.

Stop explaining black box machine learning models and build interpretable models instead, Cynthia Rudin

- This rule is very sparse and easy-to-deploy
- Only slightly worse performance than the most sophisticated black box models
- Note: need to assume for the purposes of illustration that risks for this outcome measure, dataset, etc. are appropriate for its intended use

So logical models should be sparse and parsimonious in terms of the number of conditions. Here's a nice example known as the CORELS algorithm for prediction of 2 year recidivism. It's an impressively sparse algorithm using only three features–age, gender, and number of prior offenses–and it's very easy to deploy.

But is it really Trustworthy?

Table 1 | Machine learning model from the CORELS algorithm

IF	age between 18–20 and sex is male	THEN predict arrest (within 2 years)
ELSE IF	age between 21–23 and 2–3 prior offences	THEN predict arrest
ELSE IF	more than three priors	THEN predict arrest
ELSE	predict no arrest	

This model from ref. [39] is the minimizer of a special case of equation (1) discussed later in the challenges section. CORELS' code is open source and publicly available at http://corels.eecs.harvard.edu/, along with the data from Florida needed to produce this model.

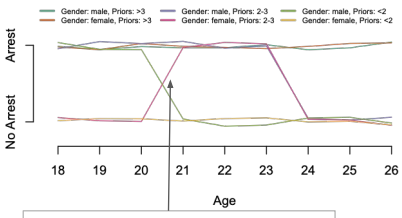
Stop explaining black box machine learning models and build interpretable models instead, Cynthia Rudin

Everything else being equal (**Ceteris Paribus**), expect to have **same or higher risk** for

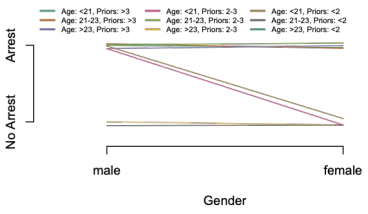
- **Lower age**
- **Male**
- **More priors**

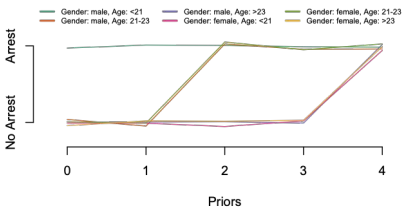
But is it really trustworthy? For sensible predictions we would expect to have the same or higher risk, everything else being equal (i.e. ceteris paribus), with lower age, male gender, and more priors. It is possible that this intuition for trustworthiness is not correct, but at the very least we should easily be able to assess if such constraints hold and if not we would want to investigate that further before deploying such a decision rule.

Need Different Representation to Evaluate Trustworthiness: E.g. Ceteris Paribus Plots



Females with 2-3 priors



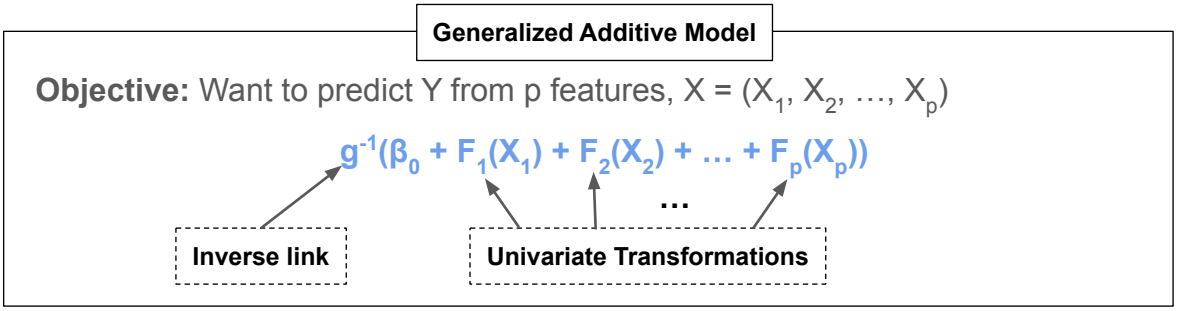


Surprisingly, does NOT appear TRUSTWORTHY!

1. **Females with 2-3 priors** predicted to have less risk when younger. Unless explained, not trustworthy for this group
2. Sparse rules necessarily also exhibit **Dichotomania**. Not trustworthy "around" such cutoffs

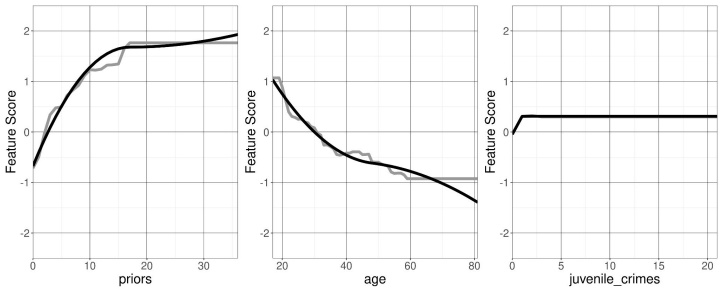
To assess trustworthiness we actually need a different representation of the prediction algorithm. Listing the rules in the logical model is not insightful. Instead we need ceteris paribus plots that show the impact of changing a single feature (shown on the x axes) while keeping the other features fixed. The different lines in the plots are for different combinations of features not being varied along the x axes. Surprisingly, the CORELS algorithm does NOT appear trustworthy! Females with 2-3 priors are predicted to have less risk when younger than older females who have the same number of priors. Unless this is explained, predictions for this group should not be considered trustworthy and such behavior is not transparent from the simple formulation of the model used to make the predictions. Furthermore, sparse rules necessarily also exhibit "dichotomania" in that small changes in a feature value can lead to an unreasonably large changes in predictions. For example, there are abrupt changes in predictions between ages 20 to 21 and 23 to 24. Subjects with birthdays a mere few months of these transition ages would have vastly different predictions depending on slight changes in age. These arbitrary prediction instabilities would have undesirable big consequences if such a rule were actually used in a setting such as parole determination.

Additive Models for Tabular Data



So as an alternative to logical models, let's consider additive models. These can be quite general since each feature value can be transformed independently of the other feature values. Furthermore, the additive sum of the transformed features, i.e. the total additive scores, are input to a univariate inverse link function to obtain the final predicted output. For continuous outcomes the link function is usually the identity and for binary outcomes the link function is the logit transformed probability. While additive models can accommodate arbitrary numbers of interactions, this adds to model complexity and unless stated otherwise, we will assume there are no interactions and the model is directly additive in the features.

Additive Models Set **The Gold Standard** for Interpretability



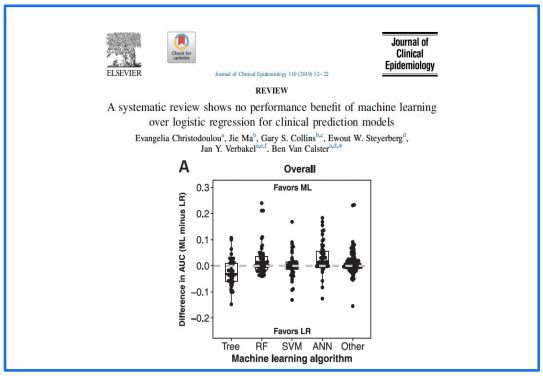
Total Score = Intercept + Prior Score + Age Score + Juvenile Score
Probability = g^{-1} (Total Score) = $e^{\text{Total Score}}$ / (1 + $e^{\text{Total Score}}$)

- **Effects are fully disentangled** and can be **visualized**
- **Ceteris paribus** relationships are **clear**
- **"Interpretable" is not the same as "Trustworthy":**
  It allows decision makers to *assess trustworthiness*

This shows an example of an additive model for the same Compass dataset. It uses three features: age, the number of priors and the number of juvenile crimes committed. Each of these features has a corresponding component function that maps the input feature into a score. The feature scores are then summed and an overall intercept is added to obtain a Total score. This total score is then transformed via the inverse link, here the anti-logit, to obtain the predicted probability of recidivism in 2 years. One can immediately assess properties such as monotonicity in the input features. Here we see the model predicted probabilities increase as a function of number of priors and juvenile crimes with the increase stabilizing at approximately 15 priors and one juvenile crime. The function also increases for younger subjects. Also, the model does not exhibit dichotomania in the influence of features. Feature scores change gradually with changes in the input features. Hence we can immediately see that the model satisfies the basic expectation for a trustworthy model that we described before.

Since the effects in an additive model are fully disentangled the impact on model outputs can be meaningfully visualized and quickly assessed for trustworthiness. Hence the ceteris paribus relationships are clear. Also note that interpretable is not the same as trustworthy. Additive models are intrinsically interpretable and can be assessed for trustworthiness but they need not automatically be trustworthy: the component functions will identify potential problems with the model. Based on these considerations we consider additive models to set the gold standard for interpretability.

Additive Models Have State-of-the-Art Performance



To our knowledge, all recent review and commentary articles on this topic imply (implicitly or explicitly) that the trade-off between interpretability and accuracy generally occurs. It could be possible that there are application domains where a complete black box is required for a high stakes decision. As of yet, I have not encountered such an application, despite having worked on numerous applications in healthcare and criminal justice (for example, ref. [21]), energy reliability (for example, ref. [20]) and financial risk assessment (for example, ref. [22]).

Cynthia Rudin

Interpret ML from Microsoft Research

Dataset/AUROC	Domain	Logistic Regression	Random Forest	XGBoost	Explainable Boosting Machine
Adult Income	Finance	.907±.003	.903±.002	.927±.001	***.928±.002***
Heart Disease	Medical	.895±.030	.890±.008	.851±.018	***.898±.013***
Breast Cancer	Medical	***.995±.005***	.992±.009	.992±.010	***.995±.006***
Telecom Churn	Business	.849±.005	.824±.004	.828±.010	***.852±.006***
Credit Fraud	Security	.979±.002	.950±.007	***.981±.003***	***.981±.003***

FICO Explainability Challenge (Cynthia Rudin)

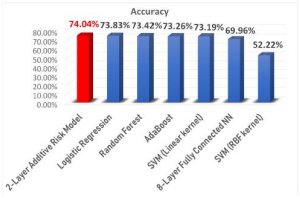
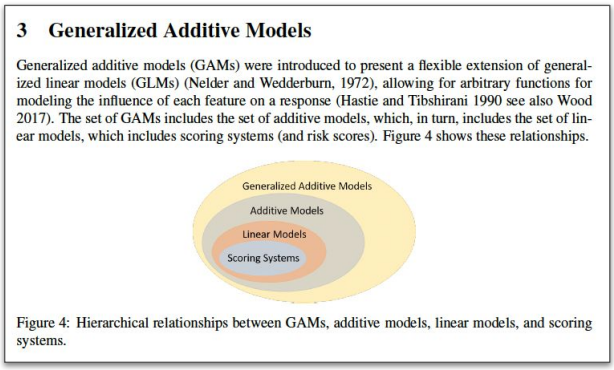


Figure 4: The accuracy of our model compared to other common machine learning models.

Is there a price to pay in performance for such clear interpretability? Surprisingly, a considerable body of empirical evidence suggests not! The top right results from Microsoft Research shows that *explainable boosting machines*, which train a type of additive model has state-of-the-art performance across a number of public datasets. In fact, on those datasets even simple logistic regression has optimal or near optimal performance. Similarly on the FICO explainability challenge, Cynthia Rudin showed that a version of logistic regression that allowed nonlinearity in the feature components via binning has near optimal performance as well. Christodoulou, in the figure shown in the top left, conducted a literature review and found that on average all machine learning methods, including logistic regression, performed similarly with the exception of decision trees which were found to have a substantially lower performance on average. In particular, additive models not only have greater interpretability, but they have better performance than trees, while maintaining comparable performance to other machine learning methods. These results are consistent with our experience on a variety of datasets.

Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges

(Cynthia Rudin)



3 Generalized Additive Models

Generalized additive models (GAMs) were introduced to present a flexible extension of generalized linear models (GLMs) (Nelder and Wedderburn, 1972), allowing for arbitrary functions for modeling the influence of each feature on a response (Hastie and Tibshirani 1990 see also Wood 2017). The set of GAMs includes the set of additive models, which, in turn, includes the set of linear models, which includes scoring systems (and risk scores). Figure 4 shows these relationships.

Figure 4: Hierarchical relationships between GAMs, additive models, linear models, and scoring systems.

- Additive Models often work well with existing standard approaches
- But the paper **identifies gaps** and **areas to improve** for even broader applicability

So additive models have gold standard interpretability and optimal or near optimal performance across a broad range of applications. But as illustrated in this paper by Cynthia Rudin, there are some remaining questions for how to build additive models that can further optimize their performance for even broader applicability. These are covered next …

Impose Flexible Constraints for **TRUST**

"Interpretable models do not necessarily create or enable trust–they could also enable distrust. They simply allow users to decide whether to trust them. In other words, they permit a decision of trust, rather than trust itself."

Interpretable Principles and 10 Grand Challenges, Cynthia Rudin

Additive models are interpretable, but not *necessarily* trustworthy

- Incorporate prior knowledge?
- Smoothness? Monotonicity?
- Minimal complexity?
  - No unnecessary "wiggliness"
  - Even no unnecessary "curvature"
- Flexibility for custom considerations?

The first challenge for additive models is how to impose flexible constraints for trust. As noted by Cynthia Rudin, interpretable models are not necessarily trustworthy. Instead they allow us to evaluate and examine whether the model should be trusted. In order to ensure the additive models are trustworthy, it is desirable to have methods of imposing constraints on the component functions. This could be based on prior knowledge and account for whether they should be smooth, monotone, avoid unnecessary inflection points (extraneous "wiggliness") or even custom considerations such as a discontinuous change point at a feature value.

Remove Unnecessary Features for **SPARSITY**

Feature sparsity facilitates model troubleshooting, ease-of-deployment, and maintenance

"Everything should be made as simple as possible, but no simpler"

Albert Einstein

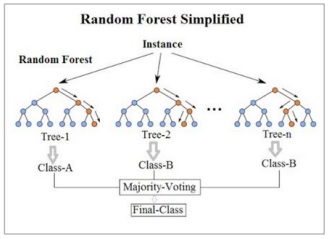
The second challenge is to remove unnecessary features from the fitted prediction model. Having a sparse set of features facilitates model troubleshooting, ease-of-deployment, and model maintenance.

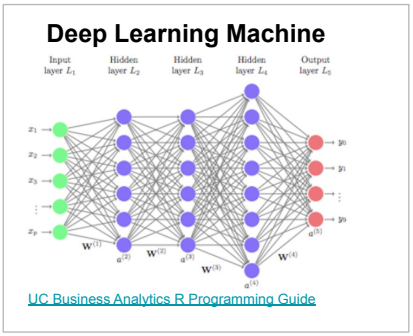
Exploit Strong Learning Methods

"We remark that GAMs have the advantage that they are very powerful, particularly if they are trained as boosted stumps or trees, which are reliable out-of-the-box machine learning techniques … **However, sparsity and smoothness are hard to control**." (emphasis added)

Interpretable Principles and 10 Grand Challenges, Cynthia Rudin

dmlc
XGBoost eXtreme Gradient Boosting





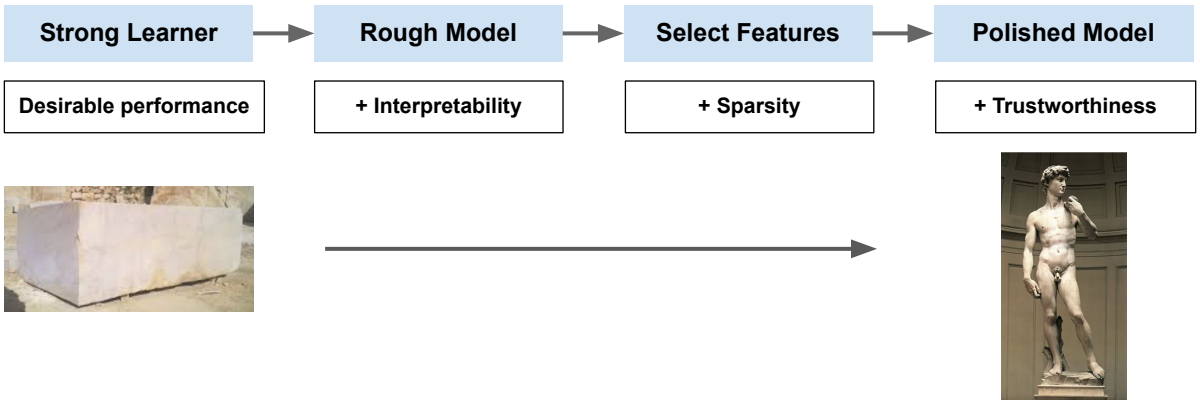
UC Business Analytics R Programming Guide

To make it even more challenging:
Can we **leverage** the strength of ***any* strong learner?**

Finally, can we exploit the strength of machine learning approaches such as boosting for extra performance while having flexible approaches to imposing constraints for trust and removing unnecessary features for model sparsity? Furthermore, can we leverage the approach using any strong learner without building a whole new algorithm? As noted by Cynthia Rudin, this can be challenging to do while maintaining the desire for sparsity and flexibility to impose constraints for trustworthiness.

Model Sculpting

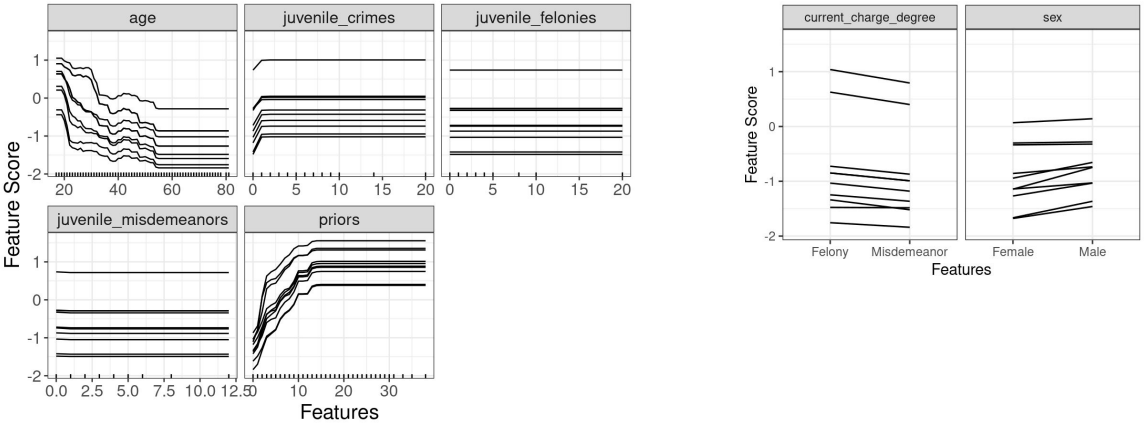
Purpose: Model sculpting is a machine learning pipeline that takes as **input a strong learner** and **outputs an interpretable, trustworthy, and sparse model (usually) with the same performance**. The method is based on easy-to-implement mathematical approximation to the base learner and is extremely flexible, fast, and visually insightful.



We propose to solve this challenge using a method called *model sculpting*. This is a machine learning pipeline that takes as input a strong learner, such as an XGBoost trained prediction model with desirable performance. Then an initial additive model that most closely approximates the input strong learner is extracted and called a rough (additive) model. Features are then selected using a directly interpretable variable importance measure. Finally, the additive model components for the selected features are potentially adjusted, e.g. via smoothing and/or shape constraints to obtain the final *polished model*. Note that the initial strong learner need not be a blackbox model; it can be any model that has the desired performance.

Ceteris Paribus Plots for an XGBoost Model: Compass Dataset

- **Also known as Individual Conditional Expectation (ICE)** plots
- **Strong Learner:** XGBoost was used to create a model
- **Use log odds scale for model sculpting** since the outcome is binary

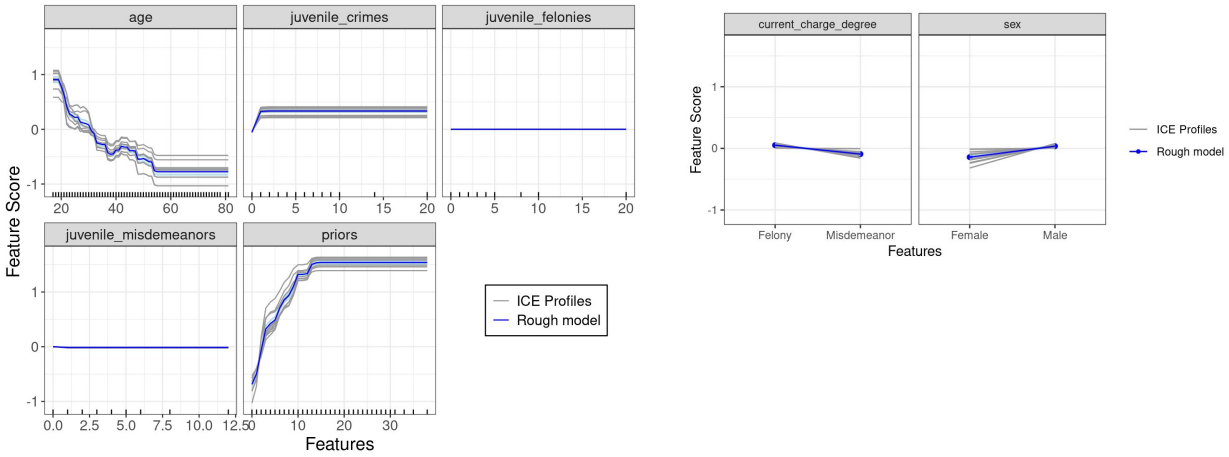


To extract the rough model, we start with ceteris paribus plots. These plots show the strong learner predictions varying one feature at a time, labeled on the x axis, while holding the other features constant. Various combinations of values for the other features lead to different lines in these plots. For the purposes of model sculpting we usually select these combinations of feature values randomly from the values seen in the training set and doing this independently for the various features. We also refer to this as sampling from the product marginal dataset, which has the same marginal distributions for the features as the original training dataset with input features made to be independent of each other. A common alternative is to keep the between feature correlations and sample at random from the training dataset.

When the outcome predictions are probabilities, as is the case here with the Compass dataset, the predictions are plotted on log odds scale when extracting the rough model. Finally, we note that these ceteris paribus plots are also frequently referred to as Individual Conditional Expectations (ICE) plots.

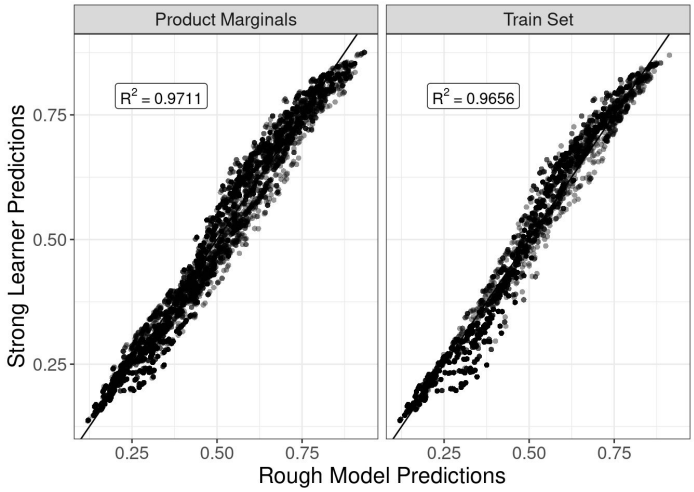
Extract A Rough Model from the Strong Learner

- **Rough Model:** Simple average of the **centered** individual ICE profiles



Next, we center the ceteris paribus profiles by subtracting off their mean values and take the mean across the sampled centered lines. These form the component functions for the additive rough model. To complete that model, an average prediction across a sample from the product marginal (or training if using the non default alternative) dataset to obtain the overall intercept. Note the consistency between the centered ICE profiles, suggesting that the original strong learner is nearly additive.

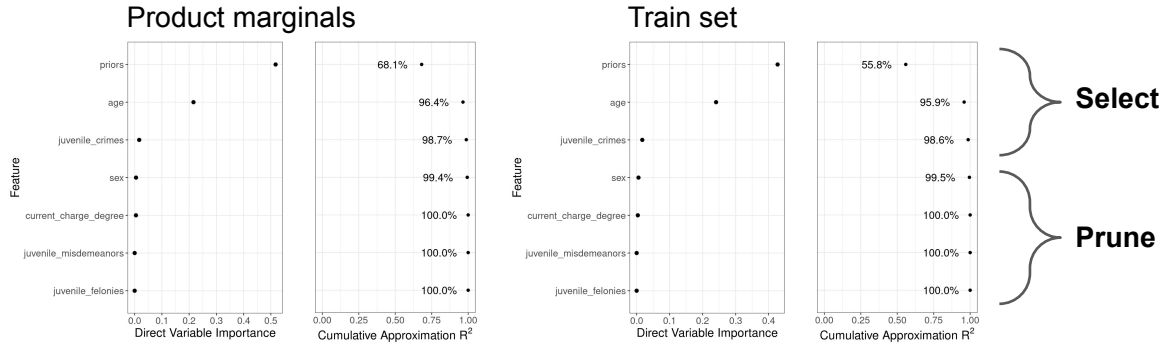
Rough Model Approximation to the Strong Learner



Additive model predictions are excellent throughout the feature space

An alternative way to assess additivity is to plot the strong learner probability predictions vs the extracted rough model probability predictions. As expected from the centered ICE plots, the additive rough model is an excellent approximation to the original strong learner, with R^2s of 0.97 on the product marginal and training datasets. Even though tree ensembles like XGBoost trained models can be extremely complex, in principle, we see here that this need not be the case. In fact, it is our experience that in most cases well trained blackbox models actually are approximately additive models at the individual prediction level.

Select Features for the Polished Model (Log odds scale)

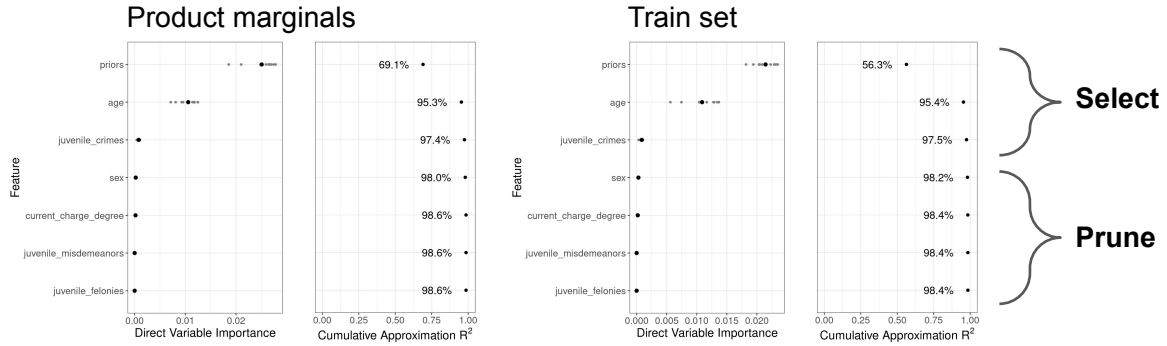


Left panel: Direct Variable Importance = Variance of feature terms

Right panel: Cumulative R^2 for prediction of full rough model using top features

Next we examine what features drive predictions from the rough model. The left panels show the variance of the rough model component function scores, which we call the *direct variable importanc*e. These represent the variances of the individual component values. The features are then sorted by their direct variable importance and for the right panel we show cumulative R^2 for how well the sequence of submodels approximates the full rough model. The top number of 68.1% represents the R^2 for the model containing only the most important feature, in this case *priors*. The predictions for these submodels "zero out" features that are not selected by setting the component functions for those features to 0. For the Compass dataset, by the time three features are included–priors, age, and juvenile crimes–the submodel approximates the full rough model with an R^2 of over 98% on the product marginal dataset. There were similar results on the training dataset, with an identical ordering of features and nearly identical approximation R^2 with the same top 3 features.

Select Features for the Polished Model (Probability Scale)



Left panel: Direct Variable Importance = Variance of feature terms

Right panel: Cumulative R^2 for prediction of full rough model using top features

The previous slide based direct variable importance on the component functions, which is equivalent to calculating variability using the individual ceteris paribus plots on the log odds scale. Since original scale prediction probabilities are generally more interpretable, we can also obtain a version of direct variable importance from these probabilities instead of log odds. To do that we obtain individual ceteris paribus plots for a sample of feature combinations and obtain the variance of the probability predictions of each line in the sample. Since the model is additive on the log odds scale and generally not additive on original probability scale, these variances are no longer the same across the different ceteris paribus lines. Therefore we now get multiple points for each feature in the left panel. An overall mean across those different variances per feature is also plotted and used to sort the features. Then cumulative approximation R^2s are shown again in the right panels. Even though the left panels are now different from before, we end up with the same ranking of features and corresponding cumulative R^2s. Again, the same top 3 features are selected.

This version of the direct variable importance plot generalizes the previous version since the model on the original scale is no longer additive–it is additive on the log odds scale only. Interestingly, this also easily generalizes to any blackbox model! For that you again plot individual ceteris paribus line variances of the predictions to construct the left panel and use the simple mean summary to sort the features. To "zero out" the effect of features not selected, simply set those features to a reasonable constant value, e.g. their mean or mode in the case of discrete features. This generalized version retains the direct interpretation since it still represents a measure of how variable the ceteris paribus plots are when individual features are

varied.

Beware of Rashomon Effects!

Rashomon explored the theme of having *multiple subjectively valid viewpoints of the same events*

Rashomon Effect in Machine Learning: Leo Breiman observed there are many models, *potentially using different sets of features*, with nearly optimal generalization error

Implication: There is no unique set of drivers of the *data*, only a set of drivers for a *specific model*.

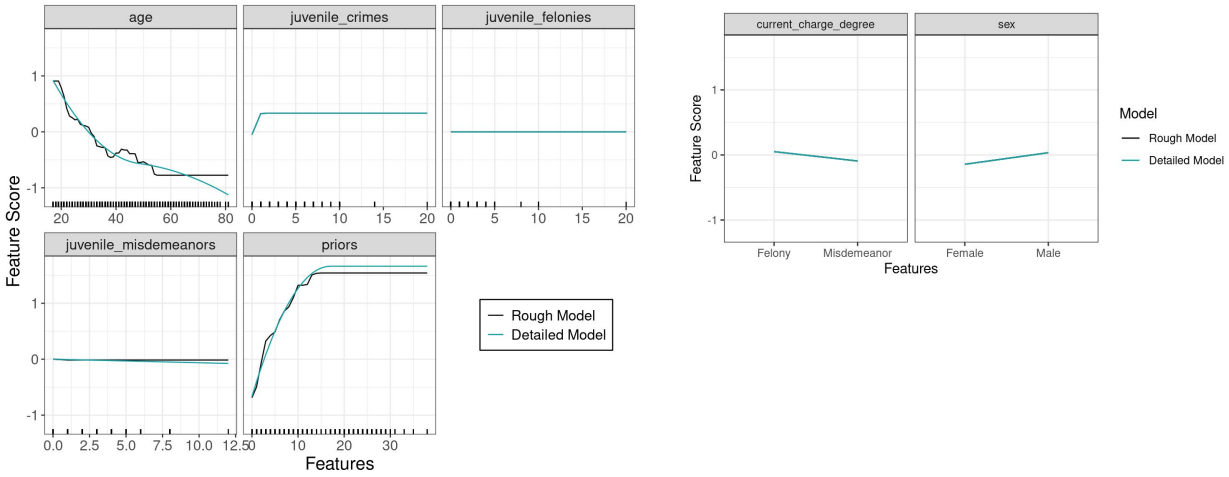
There is a **difference between explaining a *specific model*** and **explaining the *data***

- Direct variable importance explains a specific model
- There may be other combinations of variables that drive predictions for alternative models

Note that in interpreting variable importance we should be aware of Rashomon effects. This term refers to a classic movie that explored the theme of having multiple subjectively valid viewpoints of the same events. Leo Breiman coined the use of this term in machine learning where it refers to the observation that there are many models with different combinations of selected input features with nearly identical performance. This is caused by joint correlation amongst the input features and output being predicted. The implication is that there is not usually a single set of drivers of the *data*, only a set of drivers for a *specific model*. So there is a difference between explaining a specific model and explaining the data. Since the direct variable importance works with a single model and examines impacts on predictions by systematically varying individual features, one at a time for that model, it is explaining the model not the data. Explaining the data and fully understanding all possible combinations of features that drive predictions is a considerably more difficult problem and noted as one of the fundamental challenges of interpretable machine learning according to Cynthia Rudin. For practical purposes, we address explaining the data in a pragmatic ad hoc way in an additional step by deliberately including and excluding particular combinations of features and seeing what happens to overall model performance for such newly trained models. This can help explore if there are sets of features that might be masking the impact of other features as well as assessing the value added of a group of novel features.

Optionally Smooth Component Functions

- **Extraction** of the **Detailed Model** guided by three considerations:
  - **Prior knowledge**
  - **Less shape complexity is generally preferred**
- Manual extraction of individual features is OK

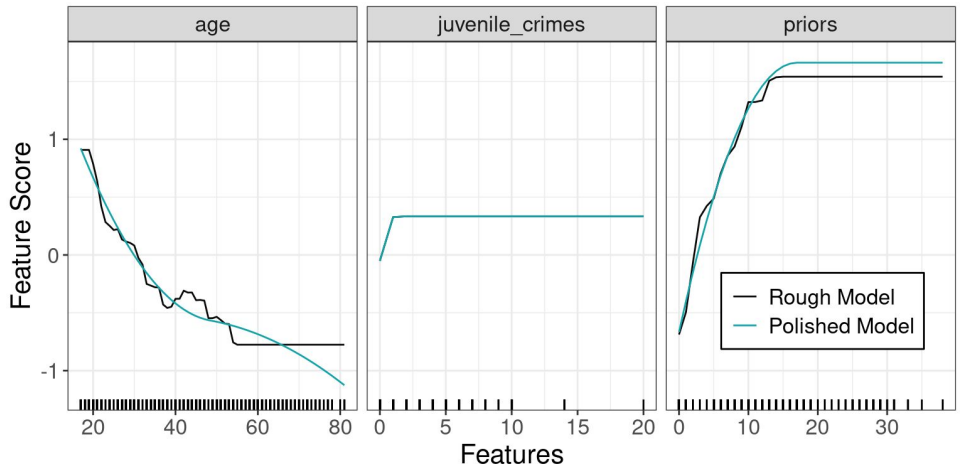


After extracting the rough additive model and selecting features we optionally modify some or all of the component functions. In this step we impose trustworthiness, as needed, based on prior knowledge. E.g. we can smooth the components and/or impose shape constraints such as monotonicity. In this case, we use a smoother and note that shape constraints are not needed since we already have desired monotonicity for the three selected features: age, priors, and juvenile crimes.

The Final Polished Model!

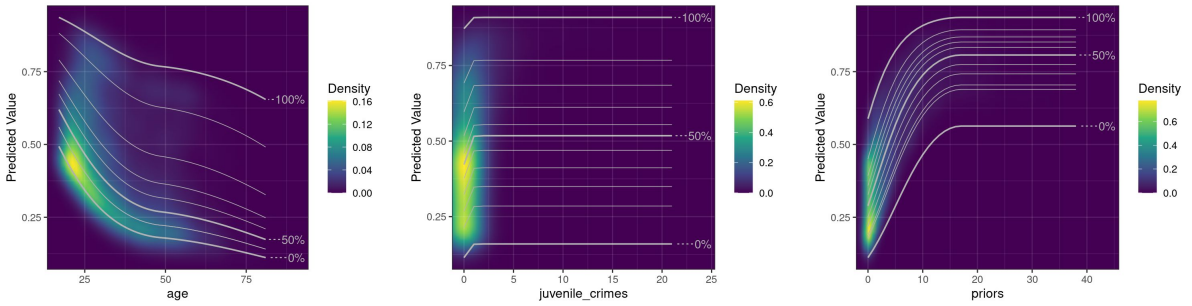
Final model components for the selected features

- Prior knowledge
- Smoothing



We now have the final polished model! The component function for age is monotonically decreasing and juvenile crimes and priors are both increasing.

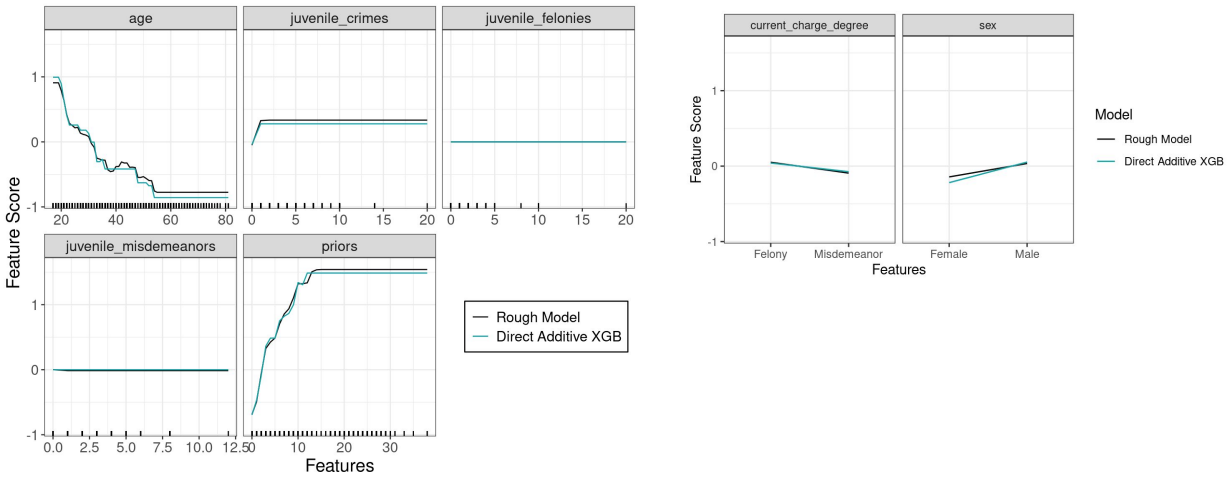
Where is the Data?



To further understand the models we can also plot bivariate densities of each feature and their corresponding predictions for the training dataset observations. Then we can superimpose percentiles of the ceteris paribus lines ranked according to their overall height. This will provide a good understanding of where the data and predictions lie in the dataset and which regions of feature space have a dearth of data. We see that most data is in an L shape for either relatively low predicted probabilities or low ages. Outside that region there is relatively little data we might wish to be cautious with predictions in that region, but this should be relatively rare. We also see that virtually all subjects have 3 or fewer juvenile crimes and therefore the predicted flat ceteris paribus lines for greater values occur very rarely. It is likely that if we had a considerably larger dataset from this population that the predictions would continue to increase for larger numbers of juvenile crimes, but since the number of such observations is relatively rare in the training dataset, XGBoost sensibly avoids wild extrapolations and limits the impact on predictions in that region of feature space. The same occurs with the number of priors which flattens at about 10.

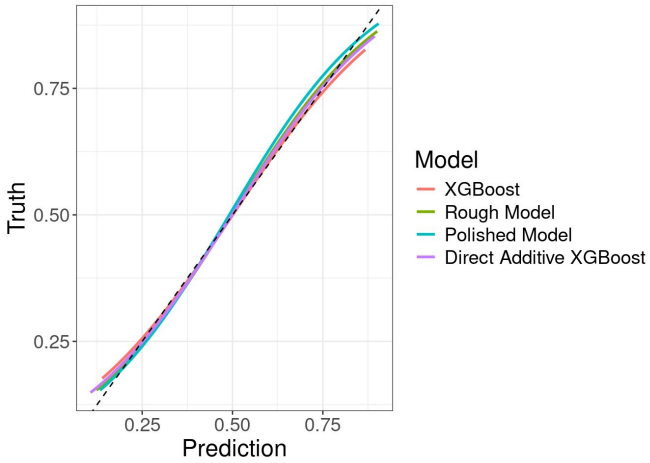
We note, again, that this plot can easily be generalized and used for blackbox models. In that case, ceteris paribus lines can cross and their ranks and percentiles are no longer uniquely defined. Instead plot a random sample of ceteris paribus lines.

Can also use XGBoost to Directly Train Additive Models



XGBoost can also be used to directly train additive models. One of the training parameters is the maximum fraction of variables per tree and when this is set so that only one feature is allowed per tree that model is necessarily additive. Here this is done for the Compass dataset and we see that the component functions are nearly identical to those obtained with model sculpting. In other examples there are larger differences between the model sculpting based component function and a direct additive model fit using XGBoost. For that reason, we routinely use both the indirect model sculpting and the direct XGBoost additive modeling approaches and compare performance.

Calibration Plots on the Test Set



Calibration plots show smooths of the outcomes vs the predictions. Identity lines at 45 degrees going through the origin represent perfect calibration

- All models well calibrated
- All models have similar R^2s

Calibration plots assess if there are systematic biases in the predictions. Ideally we want many observations with a particular prediction, say 50%, to have a true mean value of outcomes to also be 50%. These plots assess this on the test dataset for our various models. The ideal calibration line is the dashed line which is a 45 degree line through the origin. All models are well calibrated here.

Model Performance

Quadratic Loss

Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
XGBoost	0.150	0.153	0.003	0.147	0.153	0.006
Rough	0.147	0.152	0.005	0.154	0.158	0.004
Polished	0.149	0.154	0.005	0.158	0.163	0.006
Direct	0.146	0.151	0.006	0.156	0.162	0.005

Log Loss

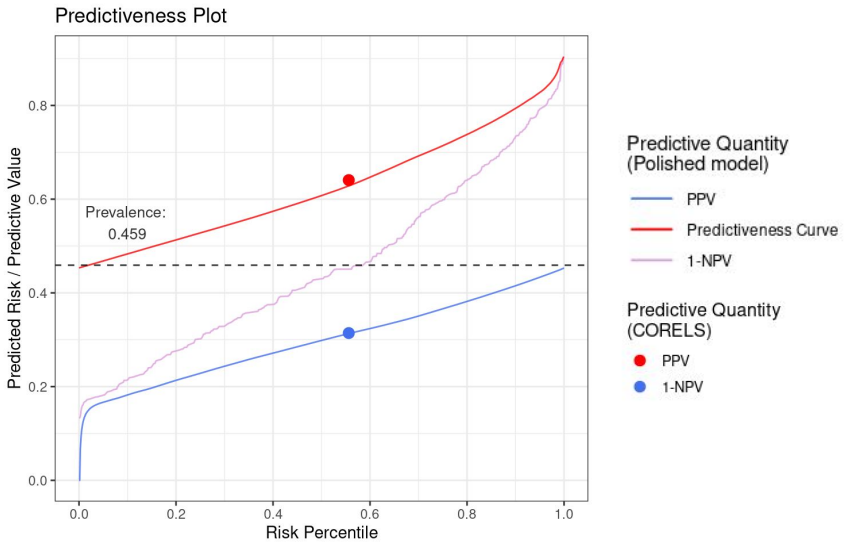
Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
XGBoost	0.113	0.116	0.003	0.112	0.117	0.006
Rough	0.112	0.116	0.004	0.118	0.121	0.004
Polished	0.113	0.117	0.004	0.120	0.125	0.006
Direct	0.111	0.115	0.005	0.119	0.124	0.005

See this blog for log loss based metrics

We look at model performance for binary outcomes using two loss functions: quadratic loss and log loss. Quadratic loss is the same as squared error and log loss is also known as cross entropy. Based on these loss functions we can look at R^2. Furthermore, after recalibrating the predictions we can also recalculate R^2 to obtain what we call a Discrimination Index (DI). Finally we subtract the Discrimination Index from R^2 to obtain a Miscalibration Index (MI). We see here that the polished model has the same or better performance as the other models, including the original XGBoost model.

See the linked blog for details on these metrics.

Test Set Predictiveness Curves for the Polished Model



See the [Stats4phc R package here]

Here we show our final performance visualization that we find useful and applies to probabilistic prediction models. The line in purple shows the estimated risk as a function of risk percentile. It is called a predictiveness curve and is estimated here using the calibration curves plotted vs the corresponding empirical risk percentiles. The flatter this line, the less prognostic information is available in the predictive model. Models with no information make a single prediction equal to the prevalence, leading to the flat dashed line. Cumulative versions of the predictiveness curve represent PPV, when cumulating down from maximum risk percentile, and 1-NPV, when cumulating up from the 0 risk percentile. The PPV and 1-NPV converge to the horizontal prevalence line on the left and right, respectively. For more information and code, see the Stats4phc package linked in the slide.

Also included on this plot are the PPV and 1-NPV for the CORELS algorithm. Interestingly we see that the CORELS algorithm has performance equivalent to the polished model at one operating point. Since the algorithm does not generate probabilistic predictions, it does not have the flexibility to choose different operating points, as might be desired for different decision makers using this algorithm.

Try Standard Linear Models as Well!

Quadratic Loss

Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
Polished	0.146	0.151	0.006	0.156	0.162	0.005
Logistic	0.138	0.146	0.008	0.145	0.168	0.023
Ridge	0.136	0.145	0.010	0.144	0.169	0.025
Lasso	0.123	0.146	0.023	0.142	0.169	0.027
Elastic Net	0.119	0.148	0.029	0.132	0.169	0.037

Log Loss

Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
Polished	0.111	0.115	0.005	0.119	0.124	0.005
Logistic	0.102	0.111	0.009	0.109	0.130	0.021
Ridge	0.101	0.111	0.009	0.110	0.132	0.022
Lasso	0.092	0.111	0.019	0.108	0.131	0.023
Elastic Net	0.089	0.113	0.024	0.100	0.131	0.031

See this blog for log loss based metrics

For a more complete analysis, it is important to try standard linear modeling approaches as well. In this case we see that the polished model appears to have slightly better performance than the various traditional and penalized regression approaches. This appears to be driven primarily by the excellent calibration of the polished model.

Two Surprising Conclusions for Logical Models

Conclusion 1:

Logical models are **fundamentally flawed for interpretability** since their building blocks are feature entanglements. They are **often also untrustworthy** and their structure inherently makes this difficult to rectify.

Conclusion 2:

Tree ensembles have an emergent property so they often behave **more like additive models than their constituent trees**. This structure is revealed by **model sculpting**, making them **more interpretable and insightful than trees**.

Summarizing our various observations, we find first, that logical models are fundamentally flawed for interpretability since their building blocks are feature entanglements. Furthermore, trained logical models often contain untrustworthy elements and their structure inherently makes this difficult to fix. Second, tree ensembles surprisingly have an emergent property that makes them behave more like additive models than their constituent trees. This structure is revealed by ceteris paribus plots and model sculpting, making them more interpretable and insightful than single trees. Therefore somewhat shockingly, at least for us, is that neither assumption of Leo Breiman's conventional wisdom on interpretability actually holds up. Of course, the state of the art performance of tree ensembles that he developed and popularized have held up well which is why we often use a tree ensemble as the strong learner to start the model sculpting process.

What if the additive model is not sufficient?

If best efforts at additive modeling still are not adequately close to a blackbox model:

- Heavily use ceteris paribus plots to understand the blackbox model
- Ensure that individual ceteris paribus lines behave reasonably by imposing constraints. E.g. XGBoost and Lightgbm both have excellent facilities for imposing monotonicity constraints. Can also ensure piecewise monotonicity such as an umbrella ordering:
  - Make copies of feature, one for each piece.
  - Zero out feature values that are not in the piece represented by the subfeature.
  - Include feature values at the split points into both adjacent parts
  - Impose the desired monotonicities separately on each subfeature.
- **Make sure good faith effort was put into additive modeling approaches! We recommend at least: standard linear modeling approaches (traditional and penalized regression), direct additive tree ensembles, and model sculpting**

If best efforts at additive modeling are still are not adequately close to blackbox model performance, we could consider the following and using a (potentially modified) blackbox model.

First, make sure to make heavy use of ceteris paribus plots to understand the blackbox model. Although popular, we do not think such methods are used enough.

Second, ensure that individual ceteris paribus lines behave reasonably. We look for similarity in shape and minimal or minor crossing of the individual lines, which is nicely automatic with models that are additive after a monotone transformation of the predictions. We also recommend imposing shape constraints such as monotonicity of the ceteris paribus lines. In some cases, a weaker version called piecewise monotonicity may be desired. This piecewise monotonicity partitions the feature values into pieces and monotonicity in potentially different directions can be forced. For example, we may want monotone decreasing predictions until a feature cutpoint, C, after which the predictions are monotone increasing. To do this we replace the feature with subfeature, one for each of the pieces. Then values outside of the piece that the subfeature represents are zeroed out. To ensure continuity we make sure to include the cutpoints defining the features appear as non zero values in both subfeatures bordering the cutpoint. Retraining the tree ensemble with appropriate subfeatures and monotonicity constraints will then ensure piecewise monotonicity.

Third we emphasize the importance of making a good faith effort at additive modeling approaches! In most instances additive models have optimal or near optimal

performance so we need to make sure non additive blackbox models are really necessary before introducing such complexity.

Summary: We ended far from where we started!

Additive Models

- Complete feature disentanglement makes additive models the gold standard for interpretability
- Usually have state-of-the-art performance

Model Sculpting

- Flexible, simple, and highly visual machine learning pipeline
- Turns strong learners into powerful, interpretable, trustworthy, and sparse models

Logical Models

- Fundamentally flawed for interpretability due to inherent feature entanglement in their building blocks
- Tree ensembles behave more like additive models than their constituent trees

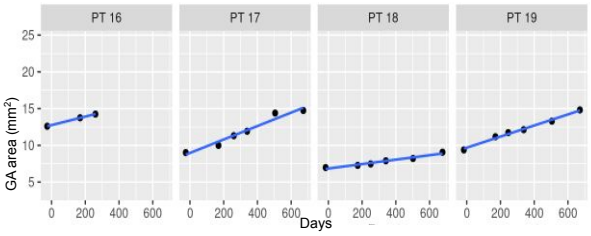
Model sculpting R package and Model sculpting workflow example

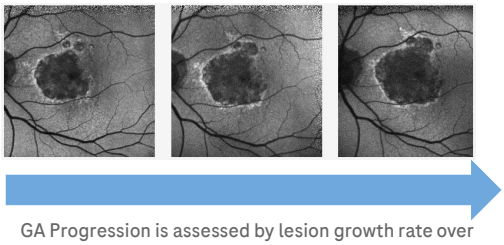
We end up with quite a different perspective from where we started!

We showed that additive models provide excellent gold standard interpretability, often with state-of-the-art performance. Direct and indirect ways of using the power of tree ensembles to construct additive models were introduced. Direct models use simple options in modern tree ensembles to ensure a single feature per tree is used and the indirect method of model sculpting extracts closest additive model approximations to strong learner blackbox models. Model sculpting techniques can be used to flexibly select features to ensure parsimony and constraints on shape for trustworthiness. Use of ceteris paribus plots showed, surprisingly, that tree ensembles are often simple additive models in disguise. Finally we also note that logical models are fundamentally flawed for interpretability due to inherent feature entanglement in their building blocks. With ceteris paribus plots and model sculpting we see that tree ensembles are often more interpretable than individual trees.

Backup

Geographic Atrophy (GA) Progression Modeling Objective





GA Progression is assessed by lesion growth rate over time in **Fundus Autofluorescence (FAF)** images as the primary endpoint in a clinical trial

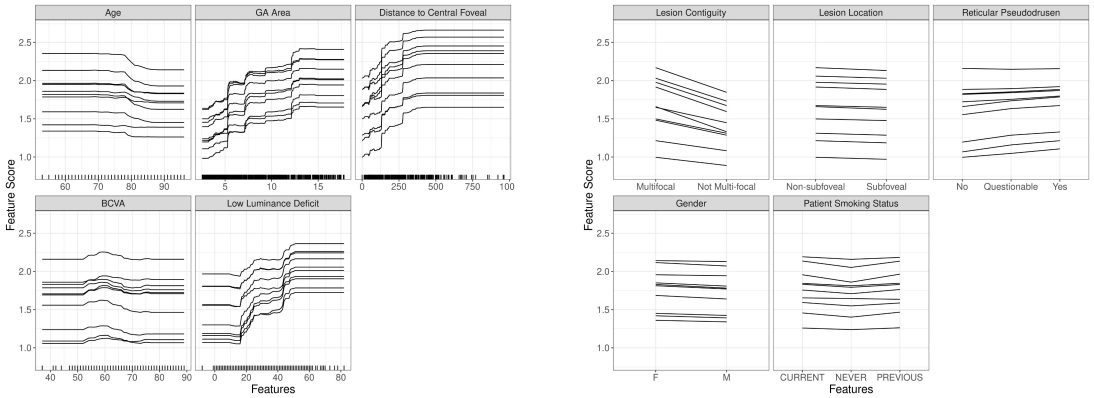
- **Endpoint in GA trials:** GA progression as defined by a **growth rate in GA area**
- **Objective:** Develop model(s) that **predict the primary endpoint** using baseline information. Use the model(s) to improve trial power via covariate adjustment

Code to generate these examples is here

The backup slides show an example where we build a model to predict progression of Geographic Atrophy.

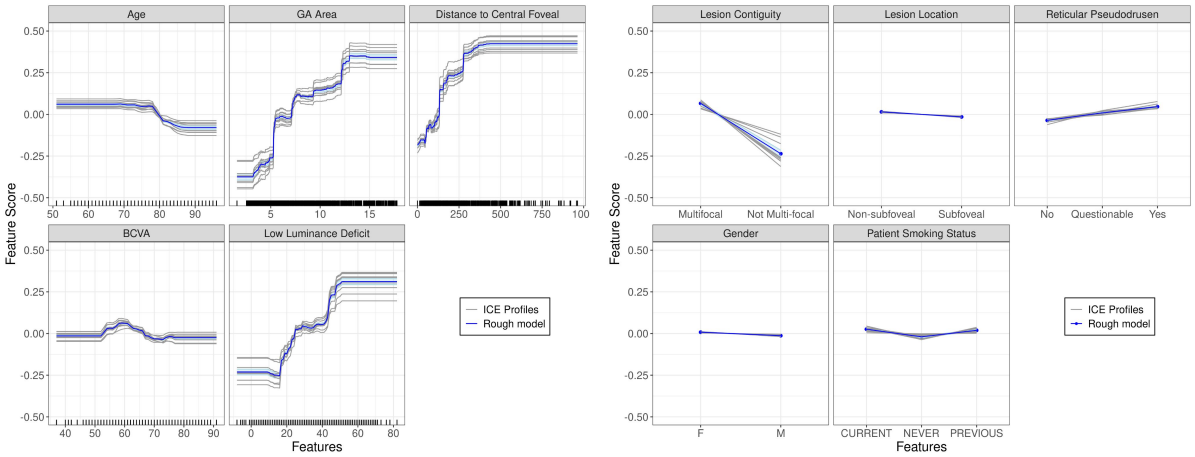
Ceteris Paribus Plots for an XGBoost Model for GA

- Also known as **Individual Conditional Expectation** (**ICE**) plots
  - Vary one feature holding all others constant
  - (In some versions they are then centered)
  - Other features can be independently selected from the product marginal distribution

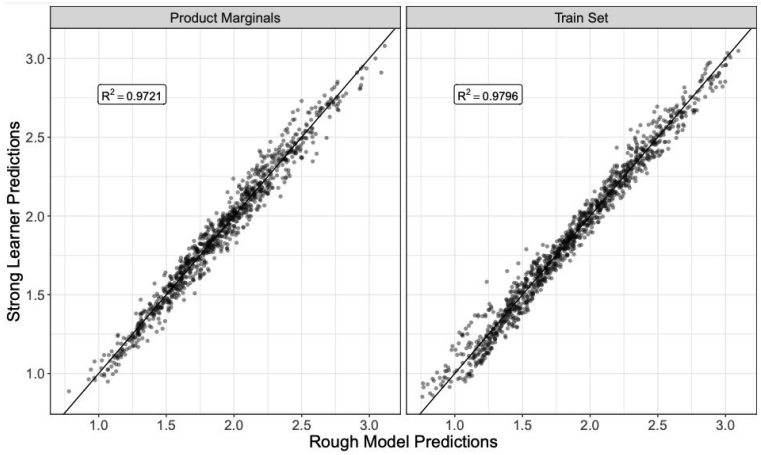


Extract A Rough Model from the Strong Learner

- **Rough Model:** Simple average of the **centered** individual ICE profiles

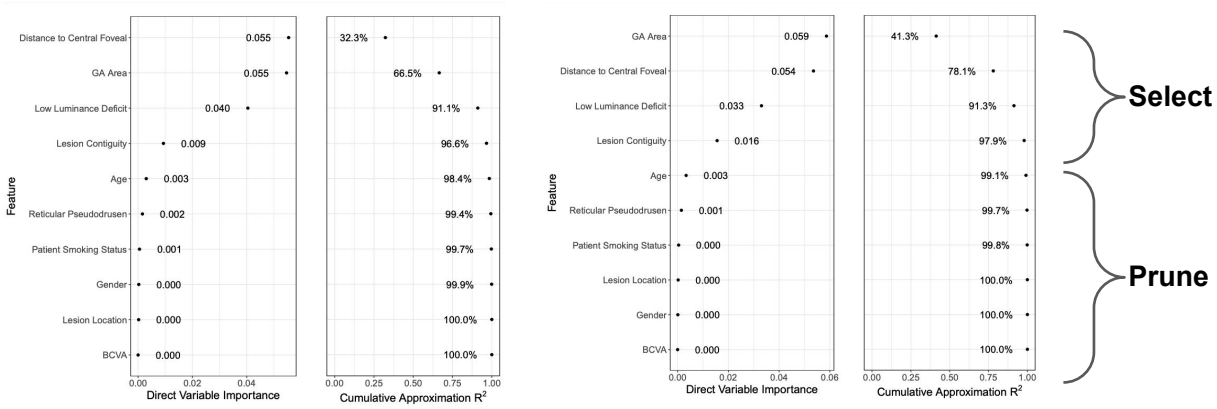


Rough Model Approximation to the Strong Learner



Additive model predictions are excellent throughout the feature space!!!

Select Features using Direct Variable Importance



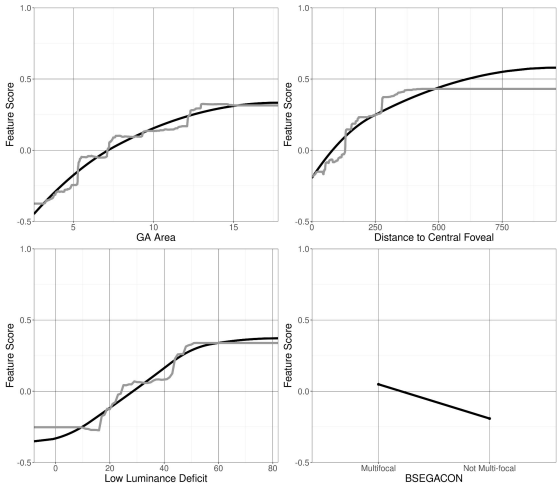
Left panel: Direct Variable Importance = Variance of feature terms

Right panel: Cumulative R^2 for prediction of full rough model using top features

Obtain Final Polished Model

Final model components for the selected features

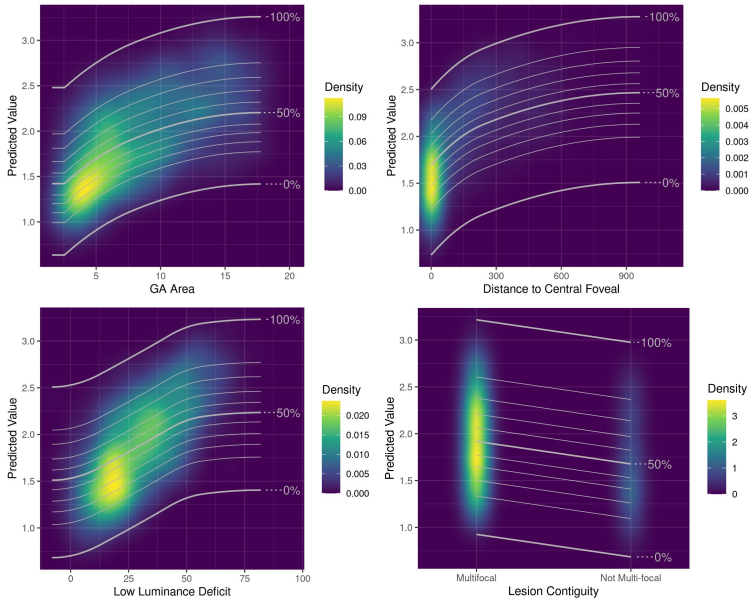
- Prior knowledge
- Smoothing



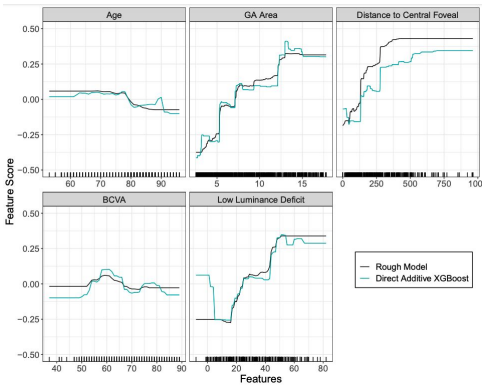
"The sculpture is already complete within the marble block, before I start my work. It is already there, I just have to chisel away the superfluous material."

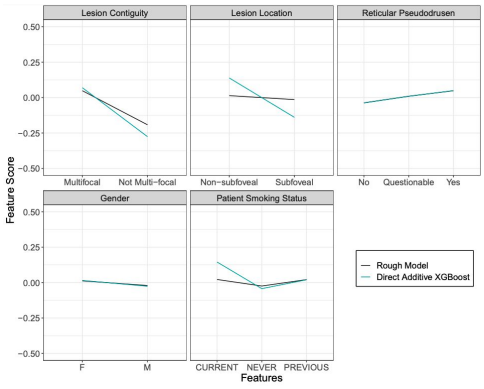
–Michelangelo

Where is the data?



Can also use XGBoost to Directly Train Additive Models



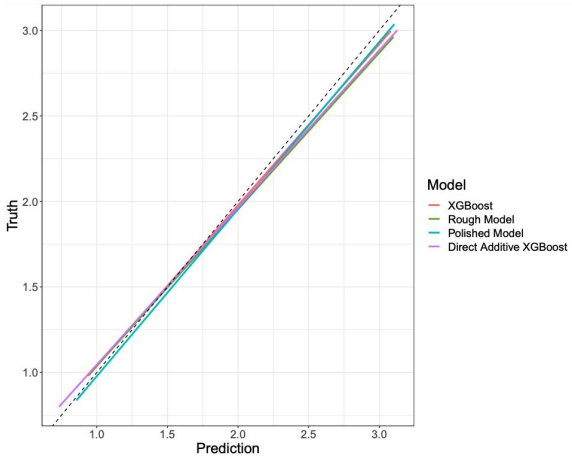


Model Performance

Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
XGBoost	0.251	0.263	0.012	0.212	0.232	0.021
Rough	0.245	0.263	0.018	0.197	0.218	0.020
Polished	0.245	0.262	0.017	0.216	0.218	0.002
Direct	0.244	0.258	0.015	0.207	0.223	0.016

For more on these performance metrics, see "Everything you wanted to know about R^2 but were afraid to ask"

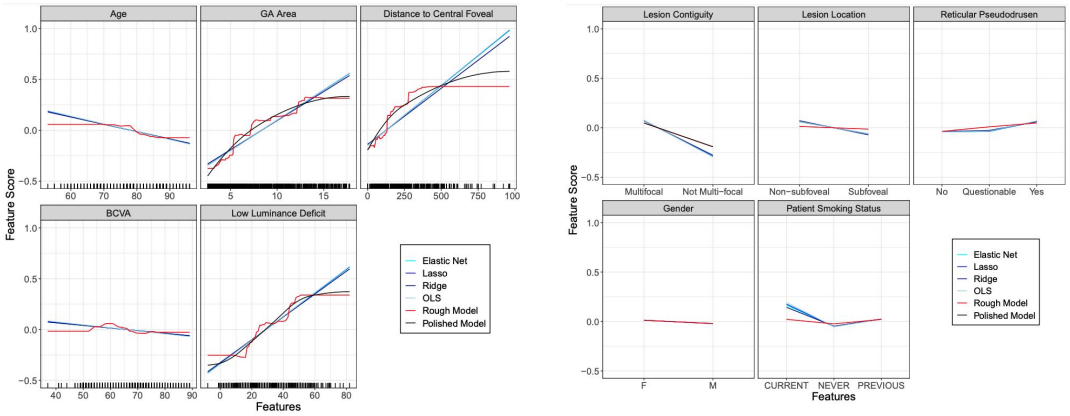
Calibration Plots on the Test Set



Calibration plots show smooths of the outcomes vs the predictions. Identity lines at 45 degrees going through the origin represent perfect calibration

- All models well calibrated

Do not forget simpler benchmark models!



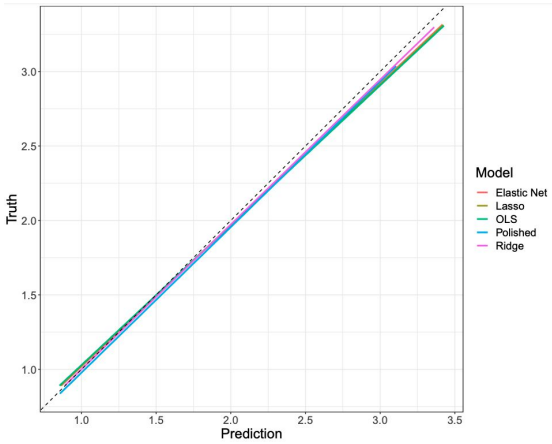
Unselected polished model components are identically 0 and not shown

Benchmark Model Performance

Model	Training Set (Resampled)			Test Set		
	R^2	DI	MI	R^2	DI	MI
Polished	0.245	0.262	0.017	0.216	0.218	0.002
Linear	0.250	0.268	0.018	0.221	0.223	0.002
Lasso	0.248	0.263	0.014	0.222	0.223	0.001
Ridge	0.250	0.264	0.014	0.222	0.222	0.001
Elasticnet	0.248	0.267	0.019	0.222	0.223	0.001

Note how well the linear models work here!

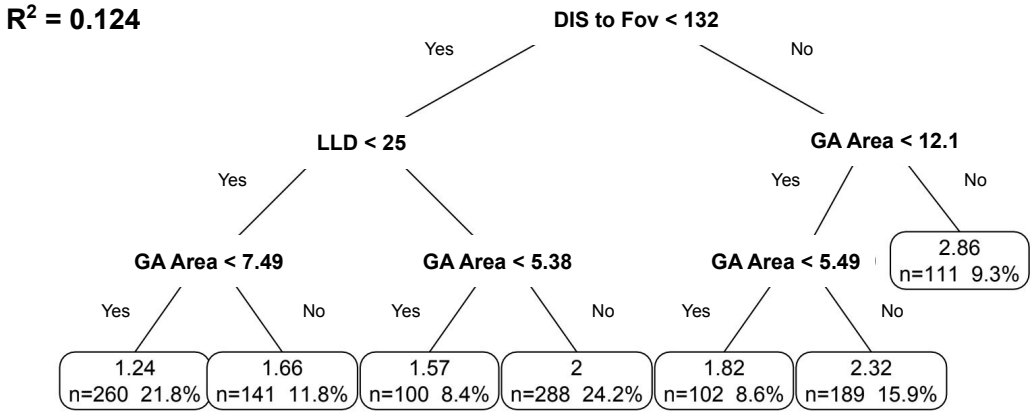
Calibration Plots Test Set



Calibration plots show smooths of the outcomes vs the predictions. Identity lines at 45 degrees going through the origin represent perfect calibration

- All models well calibrated
- All models have similar R^2s

GA Progression Decision Tree



- **Ceteris Paribus relationships are not clear** directly from the tree
- **Severe Dichotomania**: the more plausible smooth relationships not reflected here
- **Poor performance**