Reject Inference Fallacies

Daniel Tom, Ph.D.
https://orcid.org/0000-0003-4853-2498

DTom Computing
https://www.linkedin.com/in/DanielTom

November 13, 2023

Abstract

Inferring the performance of rejects with a reject inference model built on the approved provides no new information. Nevertheless, a model built on just those approved already extrapolates quite well in the reject region. To check if there are limitations, we devise a RI test to determine if a broader approved population is needed for development.

Keywords: reject inference

Introduction

Throughout his career, this author has encountered a crackpot of modeling practices and advice which turned out to be misconceived, misguided, misinformed, imprudent, or downright false.

Some of these practices and advice would be questionable at the outset. For example, when asked how to reduce the number of predictors to a manageable few before passing them to stepwise logistic regression, a senior modeler advised us to run a stepwise logistic regression and see what predictors it chooses. Needless to say we got the runaround.

Refuting other long held modeling practices may require extensive research. Prescribed segmentation is an example of such. It has been a misconception that for a model to perform well, there needs to be many segments, and a deep segmentation tree to boot. Invariably, the modelers would exhaust their resources building out the myriad prescribed model segments. This runs out the project clock, leaving no time to explore better alternatives. The one who prescribed deep segmentation goes unchallenged, so this bunk practice perpetuates. We come along with our Logistic Regression with AI Beam Search modeling process [1], which helps overcome building over a thousand such prescribed segments, and saves us enough time to build an extra unsegmented logistic regression model. Not only does the unsegmented model beat the prescribed deeply segmented models, it also happens to beat powerful machine learning models like gradient boosted decision trees.

Without resorting to trial and error, we can still get a fair sense of the validity of some of these conventional modeling advice and practices using information-based reasoning. Fundamentally, a model is limited by the information in the training data, and could also be ruined by contaminated training data. One needs to look no further than GPT (generative pre-trained transformer), a large language model. After training on web pages, the GPT application is capable of perpetuating untruths and misinformation that is ubiquitous on the internet. Information-based reasoning tells us this: building a reject inference (RI) model is pointless, because the RI model is limited or bounded by the information contained in the training data.

In new account application modeling, the ingrained practice is to infer or label the performance of the reject with the aid of a RI model. When pooled with those approved which have actual on-us performance, these form the development or training data for a new account application model. The issue is that the RI model, which is built on the approved population with actual performance, contains no more information than within the approved population. Labeling the rejects with RI model inferred performance does not increase information. Saying it another way: There is no originality. Therefore, such data for building the new account application model is still limited by the information contained in the approved subset of population. Reject inference is futile. How do we show this with empirical data? This would remain outstanding for years until we get a break to crack this case open.

Serendipity

While building an unrelated existing account management model, it dawned on us that the specifics of this model have parallels to new account application modeling. We could use the data to simulate approve/reject since we have full performance information on both. With that we could begin to find answers to age-old questions. The following is an account of the project leading up to this fortuitous find. The project generates plenty of new learnings in its own right, so we are sparing no ink.

The purpose of the existing account management custom score is to help promote activation/spend/balance build. Risk management can set up multiple risk criteria for these discretionary promotions. This custom score is typically used not alone, but often in conjunction with a generic score (an industry standard consumer credit risk score). 620+ has long been our minimum requirement for the generic score. Therefore, we build and validate our existing account management unsegmented model on the 620+ subset.

We could have called it a day after building our model. Being in risk management, we always try to anticipate and mitigate unexpected events. Say one day under favorable economic conditions we relax our generic score cut from 620+ to 600+. Suddenly our custom score can no longer be used, because it is neither built nor validated on this larger population. A solution is to perform the validation now to make sure it works before launch. That is straightforward. What if we relax further to 580+? Again, we can perform the validation on the extended population. By induction, we would end up having to validate our model on the entire existing account population.

You may ask, "How does the above compare to a model built and validated on the entire existing account population?" We have the answer, because we have another model built and validated on the entire existing account population. The answer is not what we would expect: the model built on the 620+ subset actually outperforms by half a point when both are validated on the entire existing account population. It seems advantageous to build a model with less data.

While others may dismiss the difference as a fluke, we have reasons to treat this as significant. As previously reported, our Logistic Regression with AI Beam Search modeling process is very capable of finding the highest separation/highest likelihood model [2]. So we are quite confident that the difference is tangible. Another reason: we also have models built with various generic score cuts, including 600+, 580+, 640+, and 660+, and under the same validation they too outperform the one without truncation.

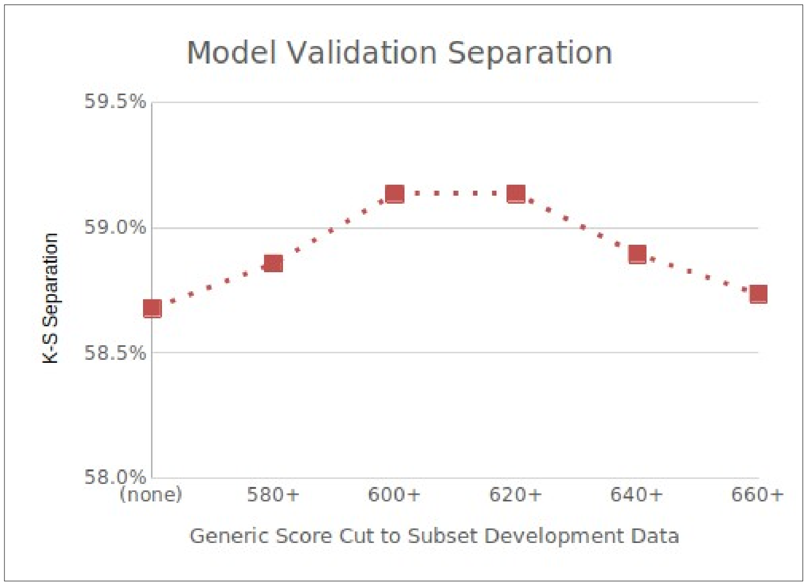

Figure 1 – Separation power of Logistic Regression Model with AI Search under various development data truncation by generic score cut. Compared with no truncation, using a generic score cut to subset the development data improves validation performance.

This piqued our curiosity. Less data is better than more? Simply use a score cut to subset data? Half a point improvement? We would dig that any day. To understand and ascertain what we observe, we begin to ask a whole slew of questions. Anticipating plenty of model builds to answer these questions, we switch to a manageable sample of the existing accounts, and compare separation with the Gini coefficient which is more sensitive [3].

Our first question is: Instead of a generic score, can we use another score to subset training data and still gain an advantage? To answer, we first build a custom score (as a first generation model) and then apply a score cut to subset the development data for other (second generation) models. Our custom score is not aligned to the generic score, nor is it required. This gives us an opportunity to choose our own custom score cuts with purpose. Observing the first generation custom score ROC chart, we have CDF curves of the Bad (default or unfavorable outcome) and the Good (non-default or favorable). A minimum requirement score would cut out more Bad than Good, so we have to be judicious. Thus, we pick a set of scores that roughly coincide with cumulative deciles of the bad. Sure enough, models built on development data subsets have higher separation, except when the score cut is as high as 650+ (90th percentile of the CDF of the Bad). We have thus reproduced the aforementioned phenomenon without using a generic score.

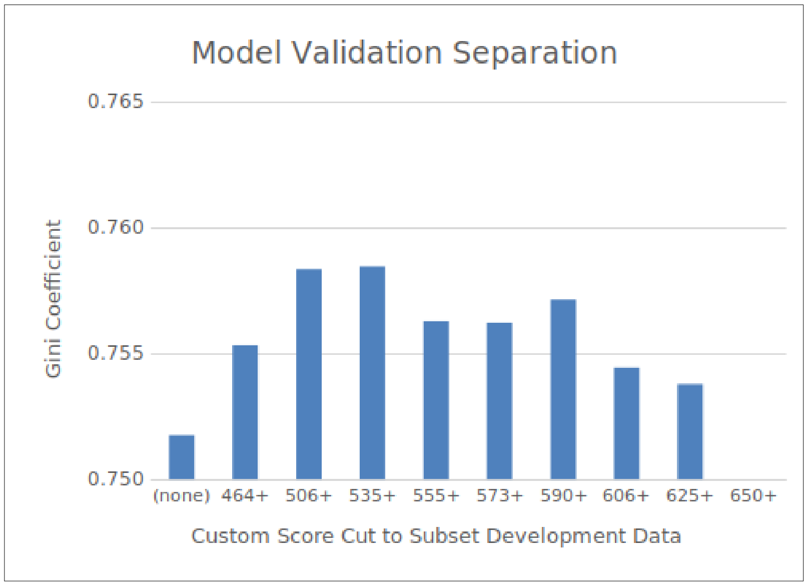

Figure 2 – Separation power of Logistic Regression Model with AI Search under various development data truncation by custom score cut. Compared with no truncation, using a custom score cut to subset the development data improves validation performance. The separation improvement is similarly observed to using a generic score cut to truncate data.

Using a lower score bound to subset the development data improves the model. How about an upper score bound? This would be our next question. To answer, we start with 535 as the lower bound as it shows the highest improvement in separation over not having a lower bound. With the help of the ROC, we select various upper bound score cuts to explore. This time we use the top CDF deciles of the Good, making sure it does not go beneath the lower bound (otherwise no data left). The question is: Does using an upper bound to subset the development data further improve model separation? The answer is affirmative. We get a higher separation using an upper bound for development data, except for 679 which drops the top half of the good population.

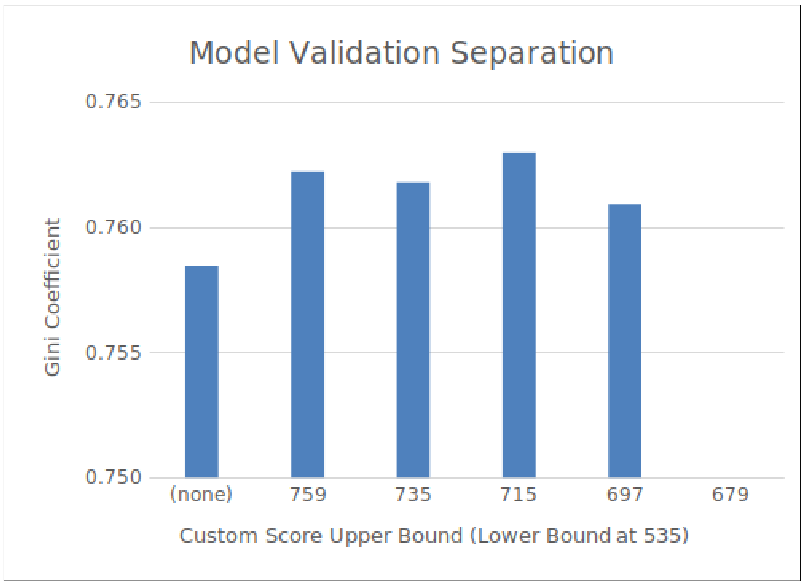

Figure 3 – Separation power of Logistic Regression Model with AI Search under various development data truncation by custom score range with lower and upper bounds. Compared with no truncation, using a custom score range to subset the development data provides additional improvement in validation performance.

Do we get an improvement in separation using some other lower bound in conjunction with an upper bound? What are model separations with various lower and upper score bounded development data? Which is the largest? To find answers to these follow-up questions, we spend the time to build all these models, and lay out the separations in a grid. We use this to generate a 3D plot, which shows a rough landscape of separation in the space of upper and lower score bounds. This landscape of separation resembles a high plateau. The peak separation (in the interior) is more than a point above the separation of the model built on development data without score bounds (lower front corner of the 3D plot Figure 4a).

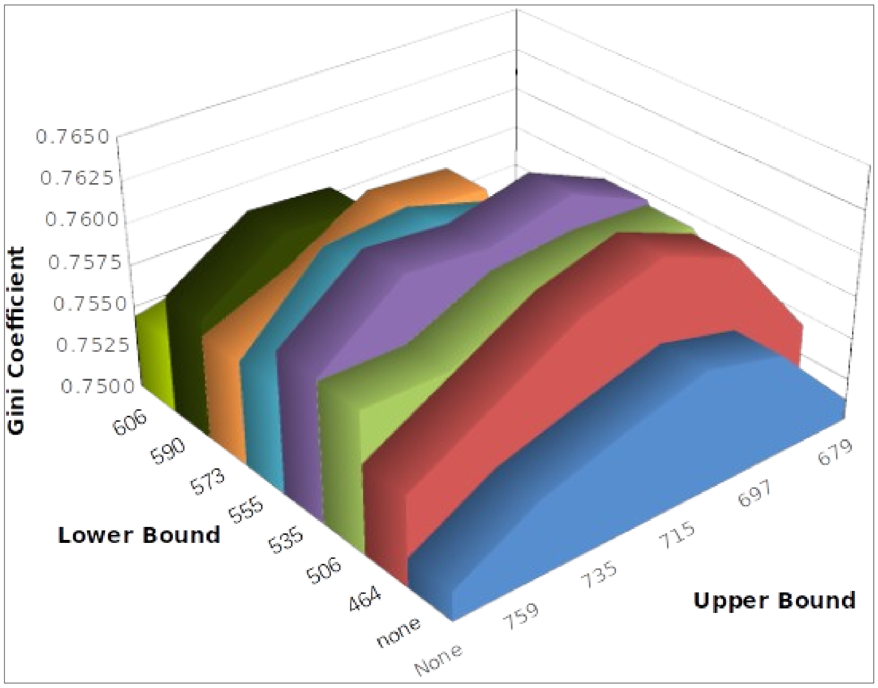

Figure 4a – Separation landscape in the space of custom score ranges with lower and upper bounds. There is a high plateau of separation when having development data bounded by custom score ranges, with a peak (in the interior) that is more than a point higher than using development data without bounds (lower front corner).

The plateau of separation drops off when either score bound is tight. When both bounds are tight, the separation drops like a cliff (far right of the 3D plot Figure 4b).

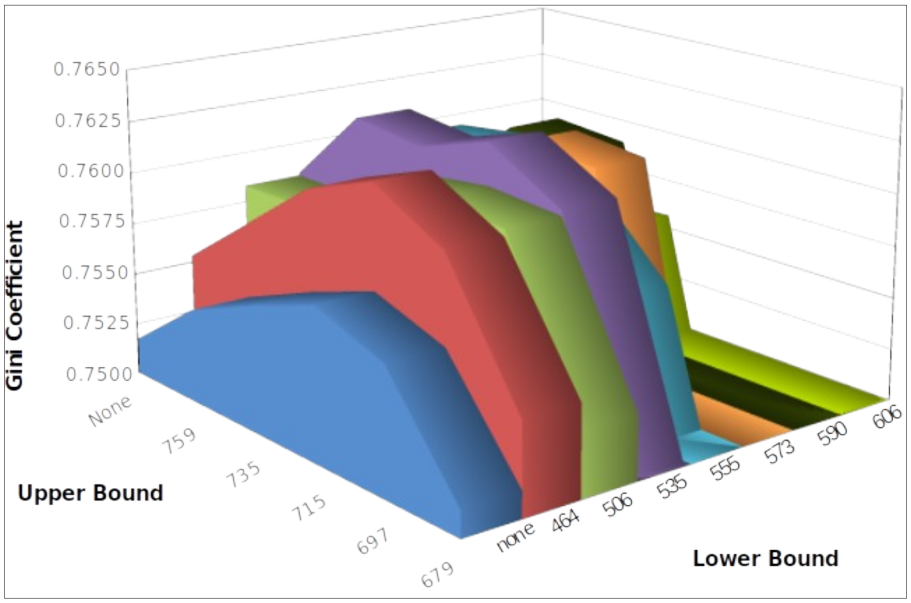

Figure 4b – Rotated view of the separation landscape in the space of custom score ranges with lower and upper bounds. This view shows the high plateau of separation drops off significantly with tight lower and upper bounds, shown on the right side of the figure. The no score bounds point is the corner on the far left.

Just to be sure, we perform extra validations of this grid of models using more hold out data. The above observations are confirmed. We also tracked this improvement in separation all the way to a recessionary vintage more than a decade back. Using score bounded data for model development improves separation power.

In summary, we can attain meaningful improvement in model separation power by using a subset of the development samples bounded by a score range. We may use a generic risk score, or build a first generation custom risk score to subset the development data. A lower bound, upper bound, or a combination of both, can generate benefit, provided the truncation is not excessive.

This is how we make sense of this phenomenon: The two ends of the risk spectrum should be slam dunk for any respectable model. To improve model performance we should redouble our effort to separate bad from good around the decision boundary. A generic score or a first generation risk score could help with targeting the decision boundary. Development data subsetting or truncation cannot be overdone though. Overlapping upper and lower bounds of the risk score obviously leaves no data. Deep truncation leaves less than sufficient data to substantiate significant predictors entering the model, thereby undermining model performance.

Data truncation or subsetting; insufficient or the lack of data – these are the hallmarks of reject inference. Once we comprehend these parallels, we realize we have a handle to grapple with the reject inference problem. A new account approval minimum score cut is akin to an existing account promotion minimum score requirement. We can utilize our existing account data to simulate approve/reject decisions in new account applications. We have a broad base of existing customers as proxy. Since we have full information on our existing accounts, we have performance for simulated rejects. Studying reject inference is now possible.

As a side note: traditional reject inference builds a coincidental model, using (non-predictive) attributes at the end of the performance window. Here, we build predictive models using application attributes. We can think of the former as shortening the performance period to zero, which should make it much easier to "predict." However, the final outcome at the end of the performance period is the same – we have actual performance of the approved, but no information on those rejected. What we learn from RI research should also apply to coincidental RI models.

Reject Inference Research

Drawing from the learnings above, we already have the first takeaway for reject inference research: Using only the approved population for model development does not necessarily yield a worse model. The separation may actually be slightly better with a moderate score cut to truncate low scores. This says, after all, reject inference is unnecessary. Only when the truncation is excessive (when the approval score cut is raised) that the separation diminishes.

How do we know when the score cut is excessive? Remember in reality we do not have performance of the rejects to compute separation. Here with RI research we can devise a test. Just like what we discussed above, for any approved population we can build models using subsets of the approved, using tighter and tighter criteria. Then we check the separation of each model on the approved population validation data. We look for parallels to having the entire population approved. If there are, we can form our conjectures around the test results.

For the first trial we use custom score cut 464+ as approved, which rejects about 10% with the lowest scores (highest risk). Models are built on subsets with tighter score cuts (around the deciles of the Bad), then validated on the 464+ population. The results draw a remarkable resemblance to those with no rejection: The separation falls off with the 650+ truncation in either case, albeit the plateaus are at different height levels. Therefore, as we find 650+ truncation excessive for the 464+ approved population, we can infer this truncation is also excessive for the entire (no reject, all approved) population.

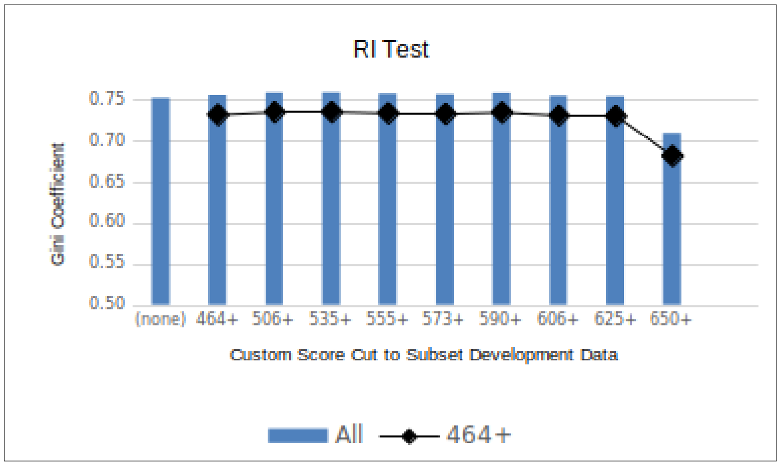

Figure 5a – Reject Inference Test for a custom score 464+ approval scenario (line chart). Models are built on subsets of the approved population with tighter score cuts, and separation performance of each is measured on the approved universe. The bar chart is the same as Figure 2 (different scale) and provides a backdrop for comparison.

Let us check our conjecture with various scenarios of score cuts to see if it holds. We will repeat with different deciles of the Bad as custom score cuts. Then we check if a separation plateau exists, and where it drops off. If they drop off at the same point, then we are confident that data truncation at this score cut is excessive.

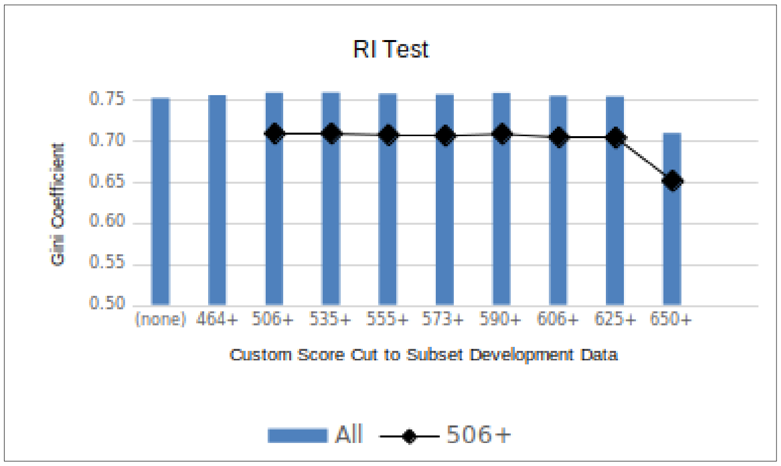

Figure 5b.

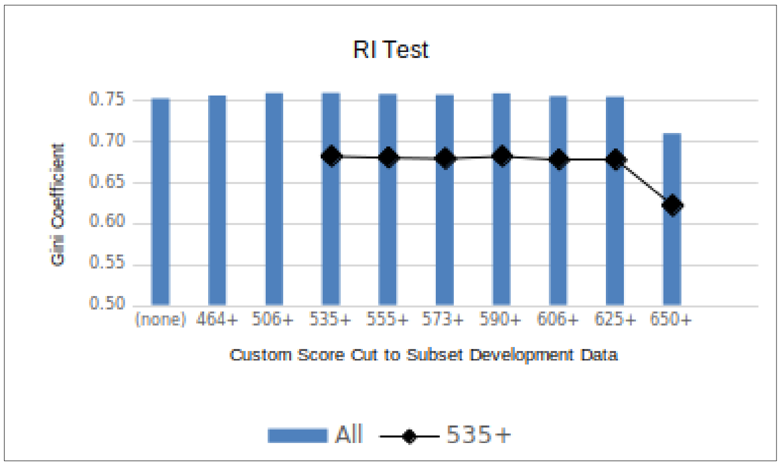

Figure 5c.

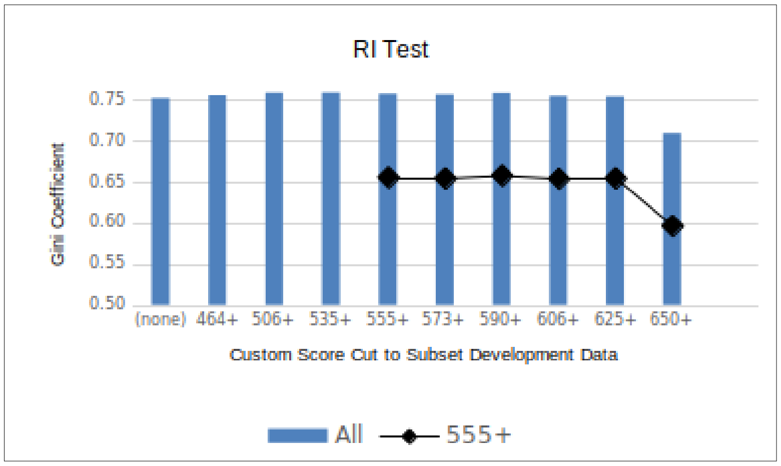

Figure 5d.

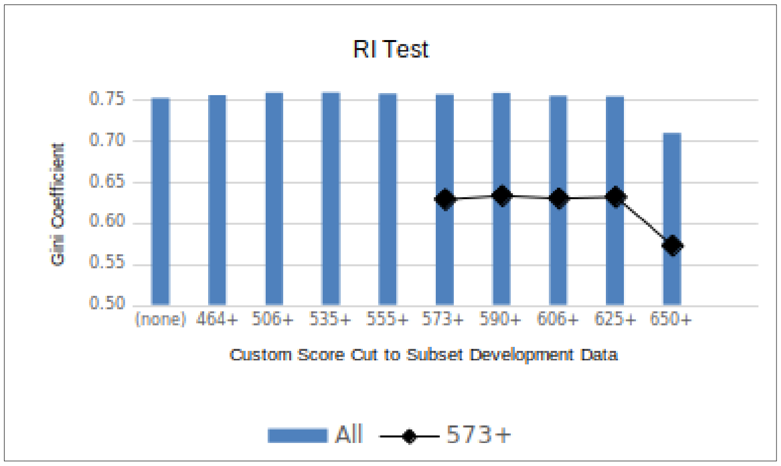

Figure 5e.

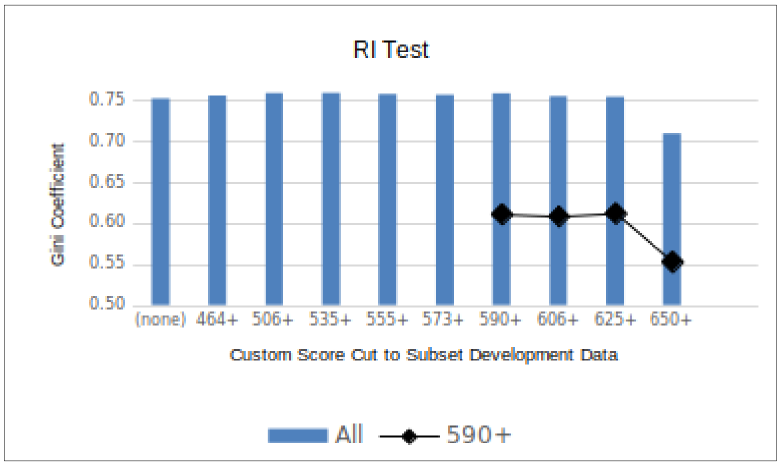

Figure 5f.

Figures 5b to 5f – Reject Inference Test for various custom score approval scenarios (5b: 506+; 5c: 535+; 5d: 555+; 5e: 573+; and 5f: 590+ line charts). Models are built on subsets of the approved population with tighter score cuts, and separation performance of each is measured on the approved universe. The bar chart is the same as Figure 2 (different scale) and provides a backdrop for comparison. All these scenarios show they all have separation plateaus, and the plateaus all drop off at 650+ which indicates excessive data truncation.

Indeed, through our RI test scenarios, we observe the separation plateaus all drop off at the same score cut, indicating the point where data truncation is excessive. Our RI test upholds the notion that, up to a point, reject inference is superfluous. That is, the model performs well without information on rejects. The point of excessive score cut can be found by the RI test.

We can easily observe the separation plateaus are at different levels for different approval score cut scenarios. The tighter the approval score cut, the lower the plateau. Although the heights of the plateaus do not change the drop-off point, the differences between levels are quite a bit larger than the modest improvement brought on by truncating low scores. This calls for an explanation, and fortunately we can address this through our prior research.

Our research on the effect of segmentation on model separation has yielded the Segmentation-Separation Formulas [4]. These formulas describe the separation power of the segments in relation to the overall scoring model. They help explain the reduction of separation power associated with tightening of the approval cutoff score.

We could apply the general Segmentation-Separation Formula for the RI research. The approval cutoff is based on a first generation score. The separation plateaus are those of second generation scores. Strictly speaking, the first generation score is an external segmentation variable, which may theoretically inject extra separation power. However, this effect, if any, may be relatively limited: The first and second generation scores share the same data ancestry, where a cutoff of the first generation score defines the subset for developing the second generation scores. There is no extra information with the same set of predictors and outcome. With this understanding we can instead apply the special Segmentation-Separation Formula.

This prior research indicates creating a score subrange segment out of an unsegmented score decreases its separation. Having an approval score cut versus none decreases score separation. Tightening the approval score cut suppresses score separation even more. This explains the difference in heights of the separation plateaus, making that a necessary result of RI test. This does not detract us from the main observation, which is finding a common drop-off point of the plateaus. This point indicates where the approval score cut would be excessive to uphold the original takeaway that reject inference is unnecessary.

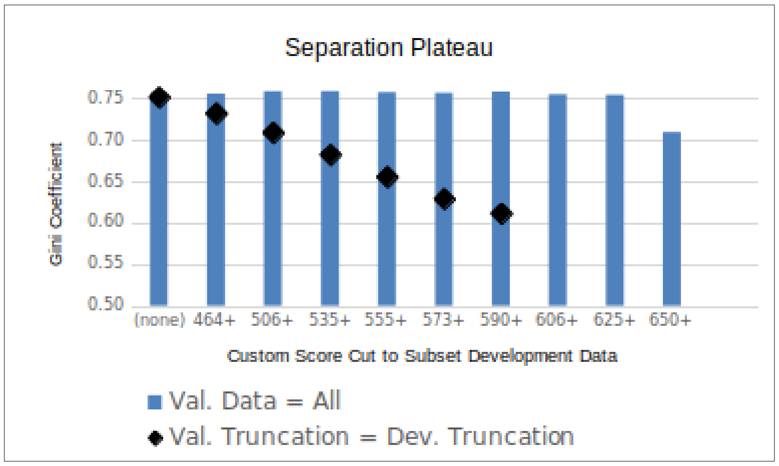

Figure 6 – Starting (leftmost) points of the separation plateaus from Figures 5a to 5f (black markers). Models are built on subsets of the approved population with various custom score cuts, and separation performance of each is measured on its own approved universe. The bar chart is for the entire (no reject, all approved) population, and provides a backdrop for comparison. The markers show the separation plateaus start out with different separations, which can be explained with the Segmentation-Separation Formula.

Next, we will check using another score for approval. We mentioned above that a separation plateau is also observed with an industry standard consumer credit risk score (generic score). So let us run the RI test with a generic score.

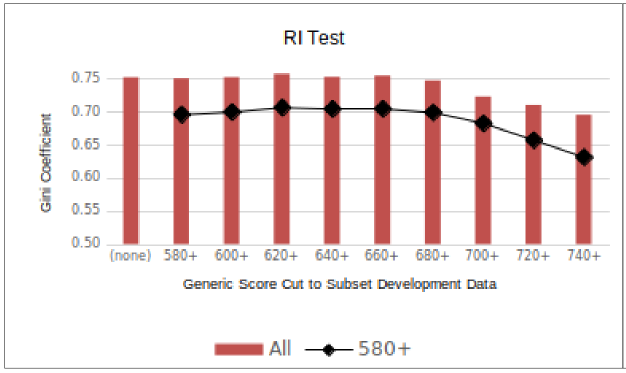

Figure 7a.

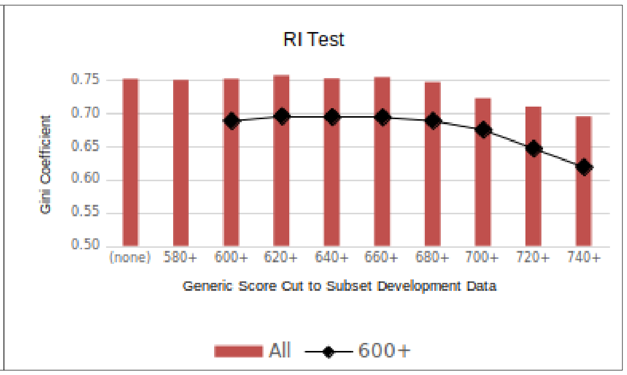

Figure 7b.

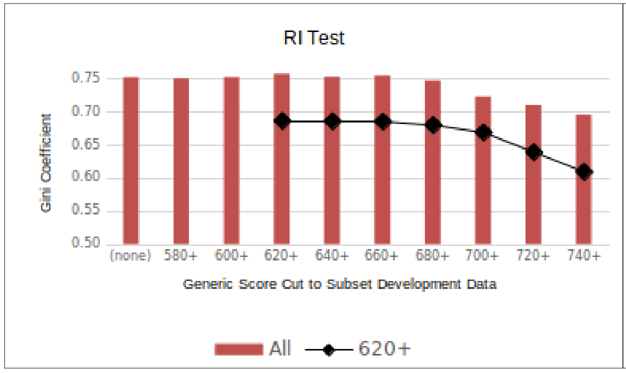

Figure 7c.

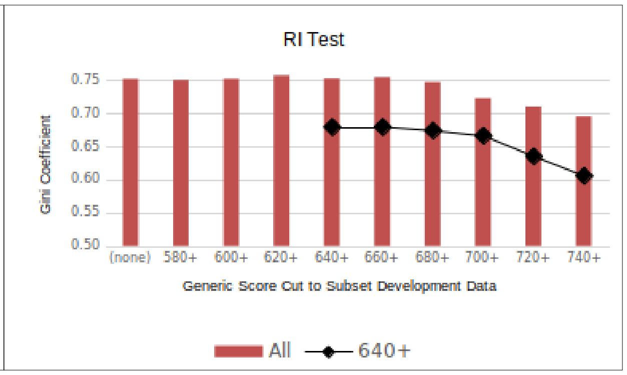

Figure 7d.

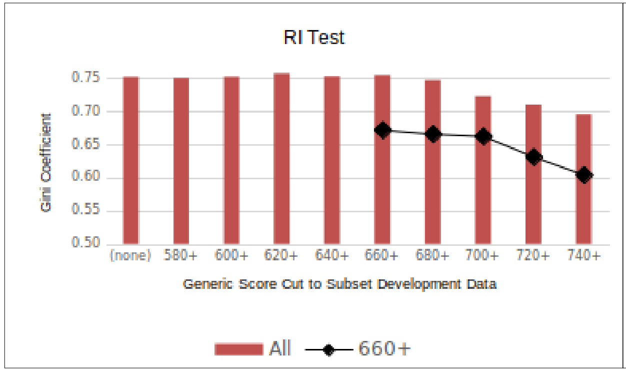

Figure 7e.

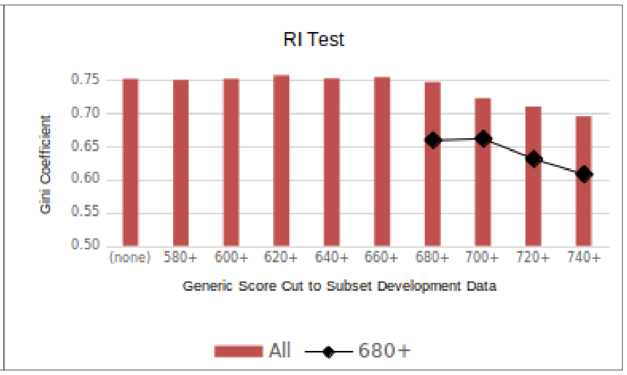

Figure 7f.

Figures 7a to 7f – Reject Inference Test for various generic score approval scenarios (7a: 580+; 7b: 600+; 7c: 620+; 7d: 640+; 7e: 660+; and 7f: 680+ line charts). Models are built on subsets of the approved population with tighter score cuts, and separation performance of each is measured on the approved universe. The bar chart is for the entire (no reject, all approved) population, and provides a backdrop for comparison. All these scenarios show they all have separation plateaus, and the plateaus all begin to drop off at generic score 700+ which indicates excessive data truncation.

Again, for each scenario of the generic score, we have similar observations from the RI tests on custom scores: existence of a separation plateau, and a common drop-off point indicating excessive approval score cut truncation.

We alluded to excessive truncation leaving insufficient data to support model separation. What if we have more data? What if we collect more application vintages? Could this alleviate or solve the excessive truncation problem? We can find out with RI research as we do have a larger data set. We just need to spend the time in performing our RI tests which involve quite a number of model builds.

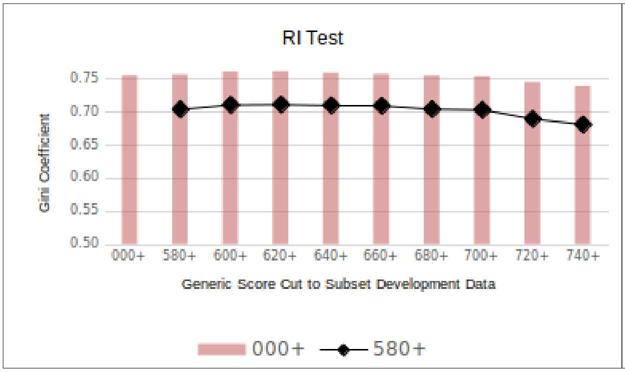

Figure 8a.

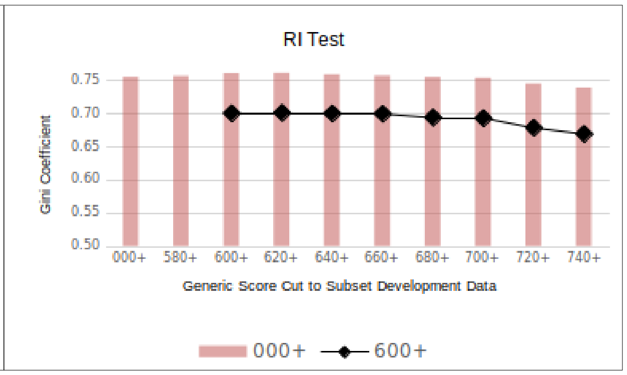

Figure 8b.

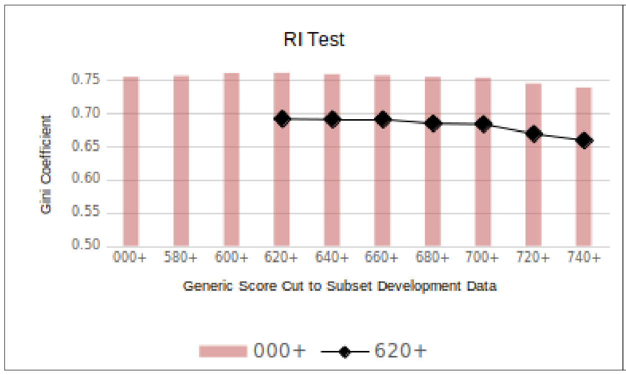

Figure 8c.

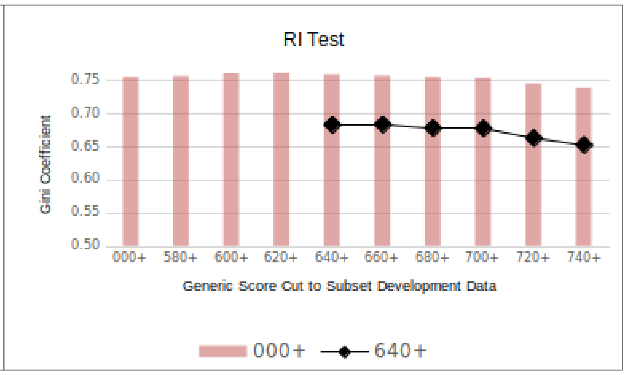

Figure 8d.

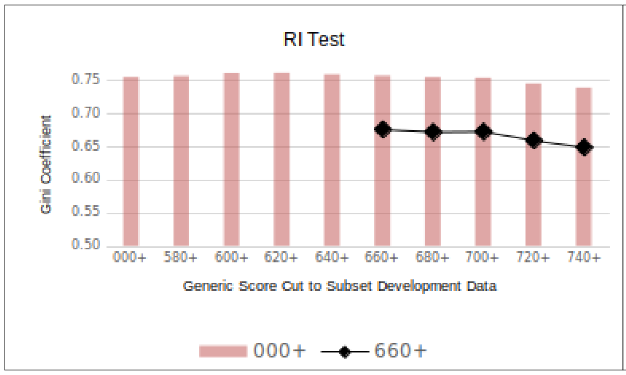

Figure 8e.

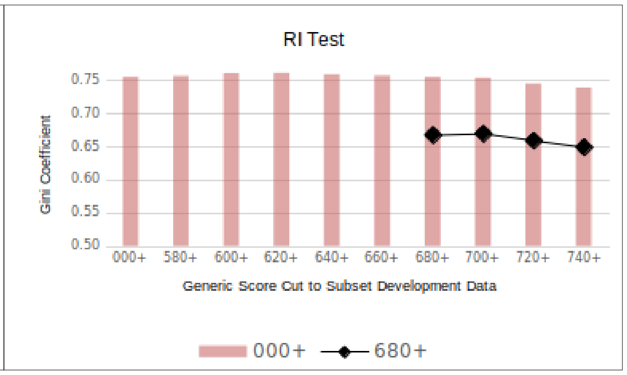

Figure 8f.

Figures 8a to 8f – Larger sample Reject Inference Test for various generic score approval scenarios (8a: 580+; 8b: 600+; 8c: 620+; 8d: 640+; 8e: 660+; and 8f: 680+ line charts). Models are built on subsets of the approved population with tighter score cuts, and separation performance of each is measured on the approved universe. The bar chart is for the entire (no reject, all approved) population, and provides a backdrop for comparison. All these scenarios show they have separation plateaus, and steep drop-offs previously seen with smaller samples in Figures 7a to 7f have largely been abated.

The result does not disappoint: Having more data can mitigate the impact of tight score cuts. A larger sample can help substantiate the significance of predictors entering the model. With more predictors available, the model can perform well. This helps extend the separation plateau to cover tighter score cuts where they may have generated excessive truncation with smaller samples.

Summary

Information-based reasoning tells us inferring the performance of rejects based on a model of known performance of the approved population provides no new information. This says reject inference is futile. Fortunately, our reject inference research concludes that reject inference is inessential. We perform the research by simulating approval score cutoff for new account application using a lower bound score criterion on existing accounts where we have complete performance information. The approved-only model performs just as well or slightly better, unless extra tight approval rejects the majority of the applications. This is true whether approval is based on a generic score cut or a (first generation) custom score cut. Not only that, adding an upper bound on the score range helps create a better (second generation) custom score. We reckon narrowing the score range helps focus the development samples on the decision boundary.

A reject inference test is devised to determine when an approval cutoff score is too tight, such that the performance of the approved population alone is insufficient for model development. The RI test entails building a number of models on subsets of the approved having tighter and tighter cutoffs. Separation of these models should be comparable (on a separation plateau) when validated on the unsubsetted population, except when the cutoff starts to be excessive which sends the separation plummeting. The point of excessive cutoff is the same, regardless of the original approval score criterion. With the RI test we can confidently build an application scoring model on the approved, and expect it to work well on the rejects without knowing their performance.

References

1. Tom, Daniel, Ph.D. (2021). Logistic Regression Collaborating with AI Beam Search, MPRA Paper.
https://EconPapers.repec.org/RePEc:pra:mprapa:116592

2. Tom, Daniel, Ph.D. (2023). Eliminating Disparate Treatment in Modeling.
https://doi.org/10.31219/osf.io/cfyzv

3. Tom, Daniel, Ph.D. (2022). Measures Of Population Stability and Instability.
https://doi.org/10.31219/osf.io/z9vd3

4. Tom, Daniel, Ph.D. (2023). Special and General Segmentation-Separation Formulas.
https://doi.org/10.31219/osf.io/mtucd