The Instability of WoE Encoding in PD Modeling

Dr. Alan Forrest
Andrija Djurovic

Standard Approaches to PD Modeling

- Probability of Default (PD) rating modeling is widely considered a cornerstone of credit risk.
- The most commonly used method for modeling PD is binomial logistic regression with categorized risk factors.
- Within this framework, two standard approaches for encoding categorical risk factors are dummy encoding and Weights of Evidence (WoE) encoding. These are the simplest kinds of pre-processing and factor engineering available to modelers.
- Practitioners often debate using one encoding method over another, while some advocate combining both within the final model. This is usually seen as a trade-off between degrees of freedom, explainability and accuracy.
- This presentation investigates factor encoding in a different way: as a possible source of model instability. It does this by examining the replicability of a model build using perfect outcome data. In other words we follow the following steps:

    1. we fix a factor encoding method;
    2. we take a training dataset on which we build a PD rating model using that adopted factor encoding method;
    3. that PD model predicts an expected outcome on the training dataset, and we use that expected outcome data as a second training dataset for a second model, built again using the adopted factor encoding method;
    4. we ask: does the second model equal the first one?

- This presentation observes that dummy variable encoding gives perfect replication (in theory and in practice), and WoE encoding does not (by example). It further explores what this means for the model risk management and potential instability of the WoE encoding method.

Dummy Encoding - Does the Model Replicate?

The following steps outline replicating the model estimates on the observed and predicted target. As can be seen, the dummy encoding replicates the exact estimates.

The simulation dataset can be found here.

1. Estimate the model of the form `Creditability ~ Maturity + Amount + Age`:

```
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -1.55       0.38   -4.08     0.00
## Maturity02 [8,12)           0.67       0.44    1.51     0.13
## Maturity03 [12,16)          0.97       0.38    2.53     0.01
## Maturity04 [16,36)          1.30       0.37    3.48     0.00
## Maturity05 [36,45)          1.57       0.43    3.68     0.00
## Maturity06 [45,Inf)         2.06       0.46    4.50     0.00
## Amount02 [3914,6758)        0.19       0.21    0.90     0.37
## Amount03 [6758,Inf)         0.58       0.25    2.29     0.02
## Age02 [26,35)              -0.54       0.19   -2.79     0.01
## Age03 [35,Inf)             -0.90       0.19   -4.68     0.00
```

2. Using the model from step 1, generate within-sample predictions `pred`.

3. Estimate the model of the form `pred ~ Maturity + Amount + Age` and compare the estimates with those from step 1:

```
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -1.55       0.38   -4.08     0.00
## Maturity02 [8,12)           0.67       0.44    1.51     0.13
## Maturity03 [12,16)          0.97       0.38    2.53     0.01
## Maturity04 [16,36)          1.30       0.37    3.48     0.00
## Maturity05 [36,45)          1.57       0.43    3.68     0.00
## Maturity06 [45,Inf)         2.06       0.46    4.50     0.00
## Amount02 [3914,6758)        0.19       0.21    0.90     0.37
## Amount03 [6758,Inf)         0.58       0.25    2.29     0.02
## Age02 [26,35)              -0.54       0.19   -2.79     0.01
## Age03 [35,Inf)             -0.90       0.19   -4.68     0.00
```

WoE Encoding - Does the Model Replicate?

Using the same dataset as in the previous slide, the following steps outline replicating the model estimates on the observed and predicted target using WoE encoding. Unlike dummy encoding, this process involves recalculating the risk factor WoE on the predicted sample. As can be seen, WoE encoding does not replicate the estimates of the perfect model.

1. Replace each modality of the risk factors with the observed WoE values.
2. Estimate the model of the form `Creditability ~ Maturity + Amount + Age`:

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -0.85       0.07  -11.69     0.00
## Maturity       -0.83       0.16   -5.12     0.00
## Amount         -0.49       0.22   -2.20     0.03
## Age            -1.05       0.23   -4.67     0.00
```

3. Using the model from step 2, generate within-sample predictions `pred`.
4. Recalculate the WoE values for the risk factor `Maturity` and compare them with the values from step 1:

WoE Values on the Observed Target:

```
##          rf          bin   woe
## 8  Maturity 01 (-Inf,8)  1.31
## 2  Maturity   02 [8,12)  0.58
## 3  Maturity  03 [12,16)  0.27
## 1  Maturity  04 [16,36) -0.11
## 19 Maturity  05 [36,45) -0.52
## 14 Maturity 06 [45,Inf) -1.13
```

WoE Values on the Predicted Target:

```
##          rf          bin   woe
## 1 Maturity 01 (-Inf,8)  1.23
## 2 Maturity   02 [8,12)  0.55
## 3 Maturity  03 [12,16)  0.29
## 4 Maturity  04 [16,36) -0.09
## 5 Maturity  05 [36,45) -0.56
## 6 Maturity 06 [45,Inf) -1.17
```

Since the WoE values calculated on the observed and predicted targets differ, this implies that the perfect model does not replicate.

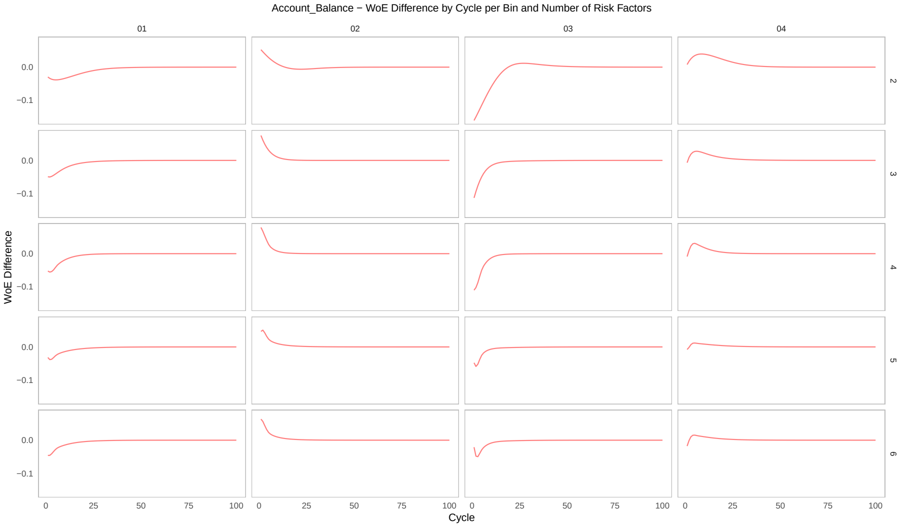
Simulation Study

The following slides present simulations visualizing the WoE difference per model recycle iteration and the number of risk factors in the model. In other words, it calculates the difference in WoE values for the same modalities of a risk factor between the initial WoE value and the one recalculated based on the predicted target (perfect model), given the number of risk factors in the final model.
The number of risk factors in the final model starts at two and increases to 13, following the order of columns in the simulation dataset.
The target variable used is `Creditability`.

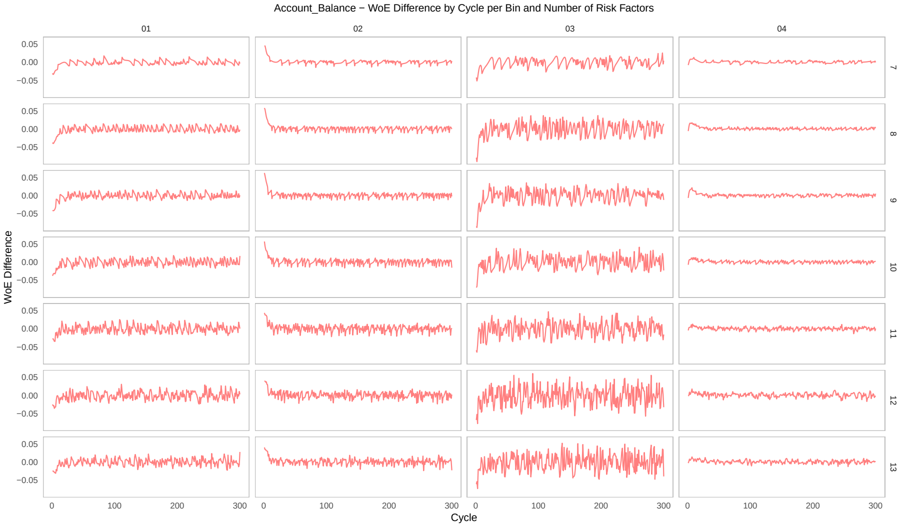
The simulation dataset is available here.

For the sake of simplicity, only the WoE differences for the `Account_Balance` risk factor are presented.
Practitioners are also encouraged to test and examine this phenomenon for other risk factors.

Simulation Results



Simulation Results cont.



Discussion Points

- We have found that WoE factor encoding shows a kind of instability or non-replicability: even on the best possible model outcome - perfect prediction - we find that we would rebuild the model differently. This is unsettling, but it is not clear how we should respond, and we open this up for further discussion and development:
  - Precisely what model risks are present in this WoE encoding instability? How far should model validators and model risk managers be concerned about this?
  - How should this concern be expressed, the risk managed? What practices should we change or adopt?
  - What performance outcomes would justify keeping the model unchanged? In some cases it seems that a perfect performance is not good enough: do we seek new kinds of model monitoring?
- We think this instability can be found in all methods that pre-process factors to reduce degrees of freedom, and more generally where upstream model outputs feed models downstream. How large can such instability become? What new insights does it give, or new practices to adopt?
- It is known that complex systems, such as neural networks for image recognition, degrade when they are trained on data that has been labelled by themselves or by other such systems. We think the unstable model replication phenomenon in this note is a similar phenomenon on a simpler scale. If this is so, these problems of machine learning degradation emerge very early, at any complexity above the simplest regression method.