FRE 9733 Big Data in Finance Week 4 Homework

Mengheng Xue

March 6, 2019

1 Clean vs Dirty Datasets

1.1 Age

From Fig. 1 and Fig. 2, we can see that in terms of ages, the clean dataset distributes more smoothly and mainly concentrates in the range between 0 and 40 months. However, for the dirty dataset, the age distribution is fluctuated, and not as concentrated as the clean dataset, which has a longer range of loan age.

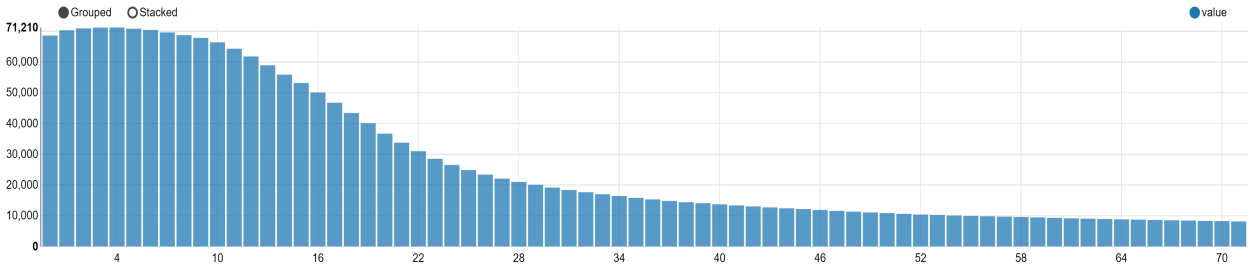

Fig. 1: df_clean1 vs age

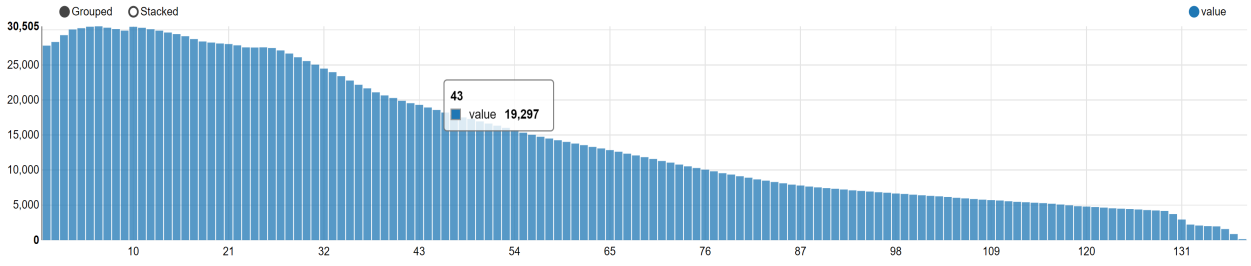

Fig. 2: df_dirty1 vs age

1.2 Term

From Fig. 3 and Fig. 4, we can find that in the clean dataset, almost all loan terms are 360 months. However, for dirty dataset, loan terms may concentrate on distinct values, such as 180, 240 and 360 months.

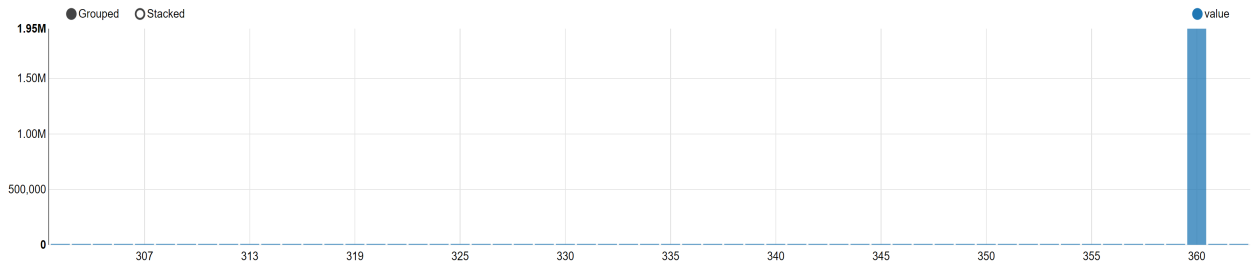

Fig. 3: df_clean1 vs term

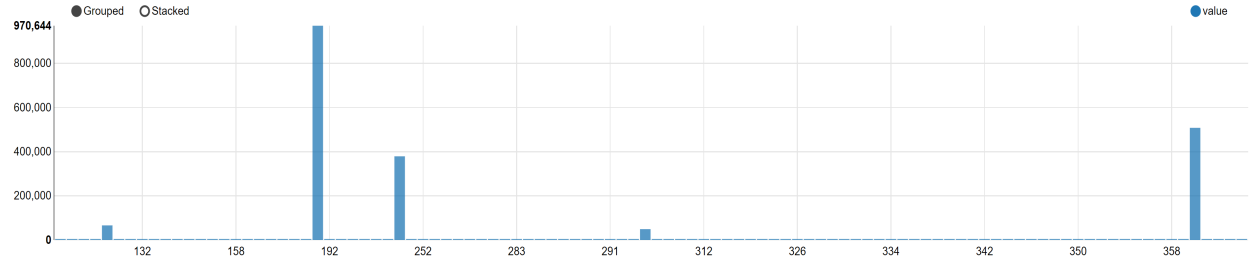

Fig. 4: df_dirty1 vs term

1.3 Credit Score

From Fig. 5 and Fig. 6, we can see that it is unexpected that the clean dataset contains more credit score outliers (300 and 9999) than dirty dataset.

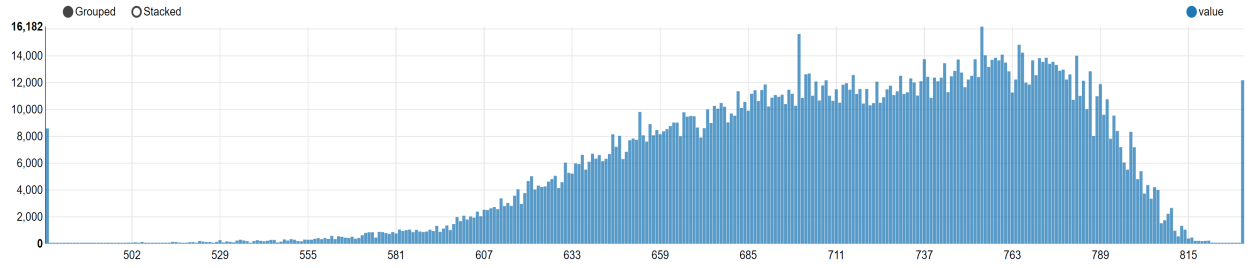

Fig. 5: df_clean1 vs credit score

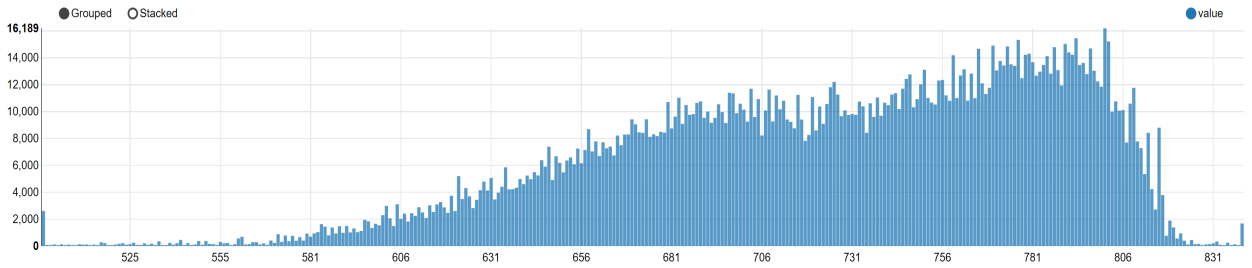

Fig. 6: df_dirty1 vs credit score

2 Model Fitting

We use df_clean1 as the fitting data, df_clean2, df_dirty1 and df_dirty2 as testing data to test our LR and RF performance. Firstly, we just use the default settings:

Features	"AGE","MTM_LTV","INCENTIVE","CREDIT_SCORE"
RF	NumTrees = 12, MaxDepth = 10
LR	MaxIter = 100, RegParam = 0.001, ElasticNetParam = 0.001

We could obtain that

	rho	distEntropy
rf_pred	0.1115	0.0653
rf_pred2	Infinity	0.0688
rf_pred_dirty	Infinity	0.0954
rf_pred2_dirty	Infinity	0.095
lr_pred	0.1912	0.0798
lr_pred2	0.2067	0.0801
lr_pred_dirty	0.1372	0.0699
lr_pred2_dirty	0.1397	0.0693

We can see that

- When we use df_clean1 as the fitting data, due to RF does not satisfies White's Regularity Conditions, we obtain infinite rhos for the test set, which does not provide useful information.
- It is kind of unexpected that for LR rho is smaller on the dirty sets than clean set. The same observation for distEntropy, which means model is smoother on the dirty set than clean set.

3 Model Improving

3.1 Random Forest

3.1.1 Hyper-parameters Tuning for RF

For RF, since we always have the infinity rho issue on test set, we cannot evaluate rho on the dirty dataset. Therefore, we only evaluate disEntropy instead.

There are 3 hyper-parameters which are important to the RF model, i.e., sample_size, NumTrees, and MaxDepth. Since we cannot evaluate rho on the dirty dataset, it is uncertain whether the model is overfitting or underfitting. Assume the model is underfitting, which means the bias of prediction is high. We need to increase the model complexity (increase NumTrees and MaxDepth) to reduce the bias. If it is overfitting, we should do the other way around. We choose different settings to test rf_pred2_dirty, the results are as follows:

setting	rho	distEntropy
SampleSize = 10k, NumTrees = 20, MaxDepth = 15	Infinity	0.1058
SampleSize = 10k, NumTrees = 12, MaxDepth = 7	Infinity	0.092
SampleSize = 10k, NumTrees = 12, MaxDepth = 5	0.1433	0.0833
SampleSize = 10k, NumTrees = 12, MaxDepth = 2	0.1402	0.0881
SampleSize = 10k, NumTrees = 20, MaxDepth = 2	0.1412	0.0893
SampleSize = 20k, NumTrees = 12, MaxDepth = 2	0.1407	0.0895
SampleSize = 40k, NumTrees = 12, MaxDepth = 7	0.1528	0.1155
SampleSize = 40k, NumTrees = 12, MaxDepth = 2	0.1418	0.0982

Based on my experience, we can find that:

- When max depth of random forest is too high, the model may suffer overfitting, which causes the variance is too high and rho becomes infinite. When we reduce max depth within 5, we can get rid of the infinity rho problem.
- When I try to evaluate the effect of sample size, it helps to get rid of the overfitting problem. You could see that even with max depth 7 (in red), if we use sample size 40K, we could still solve infinite rho problem. However, increasing sample size does not improve model accuracy, i.e., rho will not decrease.

- Based on my experiments, rho achieves the lowest (0.1402), when SampleSize = 10k, NumTrees = 12, MaxDepth = 2 (in blue).

3.2 Logistic Regression

3.2.1 Hyper-parameters Tuning for LR

For LR, since we use Elastic Net regularization, there are 3 hyper-parameters which are important to the LR model, i.e., MaxIter, RegParam and ElasticNetParam. RegParam controls the weight of the shrinkage term relative to the date-dependent error term. ElasticNetParam controls the mixing between Ridge and Lasso. We choose different settings to test rf_pred2_dirty, the results are as follows:

hyper-parameter setting	rho	disEntropy
MaxIter = 100, RegParam = 0.001, ElasticNetParam = 0.001	0.1397	0.0693
MaxIter = 200, RegParam = 0.002, ElasticNetParam = 0.005	0.1382	0.0733
MaxIter = 200, RegParam = 0.005, ElasticNetParam = 0.008	0.1385	0.0840
MaxIter = 200, RegParam = 0.002, ElasticNetParam = 0.008	0.1382	0.0732
MaxIter = 200, RegParam = 0.002, ElasticNetParam = 0.01	0.1382	0.0729
MaxIter = 300, RegParam = 0.002, ElasticNetParam = 0.01	0.1382	0.0733

Based on my test, it is not obvious how to choose MaxIter and RegParam to improve LR performance, but when we increase ElasticNetParam to 0.008, rho is maximized (in blue). As I mentioned before, ElasticNetParam controls the mixing between Ridge and Lasso. A good choice of ElasticNetParam do make a difference for the model.

3.2.2 Regressors Selection for LR

Then we try different mixes of regressors. We use hyper-parameter setting as MaxIter = 200, RegParam = 0.002, ElasticNetParam = 0.008, which obtains the best performance in the previous subsection. The results are shown as follows:

regressors selection	rho	disEntropy
AGE, MTM_LTV	0.2078	0.0811
AGE, MTM_LTV, CREDIT_SCORE	0.2077	0.081
AGE, MTM_LTV, ORIG_INTRATE	0.2078	0.081
AGE, MTM_LTV, UNIT_COUNT	0.2072	0.0806
AGE, MTM_LTV, UNIT_COUNT, ORIG_CLTV	0.2077	0.0811
AGE, MTM_LTV, UNIT_COUNT, PROPERTY_TYPE	0.2078	0.0811
AGE, MTM_LTV, UNIT_COUNT, LOAN_PURPOSE	0.2077	0.0812
AGE, MTM_LTV, UNIT_COUNT, INCENTIVE	0.2069	0.0809
AGE, MTM_LTV, UNIT_COUNT, INCENTIVE, ORIG_INTRATE	0.2064	0.0813

Based on my experiments, we can find that

- [AGE, MTM_LTV, UNIT_COUNT, INCENTIVE, ORIG_INTRATE] will achieve the best performance (in blue).
- Basically, I use forward selection method to add one extra regressor which can reduce rho the most, then repeat the previous step until rho stops improving.
- We can see that UNIT_COUNT (in red) have larger predictive power compared with CREDIT_SCORE and ORIG_INTRATE.

3.2.3 Regressors Comparsion between RF and LR

Based on the best performance hyper-parameter setting, we choose different regressors to improve model performance. The results are as follows:

regressors selection	rho	disEntropy
AGE, MTM_LTV	0.2012	0.083
AGE, MTM_LTV, UNIT_COUNT	0.2027	0.0833
AGE, MTM_LTV, INCENTIVE	0.2017	0.083
AGE, MTM_LTV, CREDIT_SCORE	0.2004	0.082
AGE, MTM_LTV, CREDIT_SCORE, ORIG_UPB	0.2005	0.082

We can see that regressors which show great predictive power in LR (e.g., UNIT_COUNT) may not be good in RF. So feature selection standards are different between these two models.

For RF, highly correlated variables will not cause multi-collinearity issues in RF. Feature selection based on impurity reduction is biased towards preferring regressors with more categories. It means that the features with more number of categories (unique values if its a numerical feature) would be more likely to get find splits; making it more important feature. That's why we can see that CREDIT_SCORE is a powerful regressor in RF and obtain best performance when I choose [AGE, MTM_LTV, CREDIT_SCORE] (in blue). However, CREDIT_SCORE may not be a good regressor for LR, since it has high correlation with other regressors, which causes the overfitting problem.