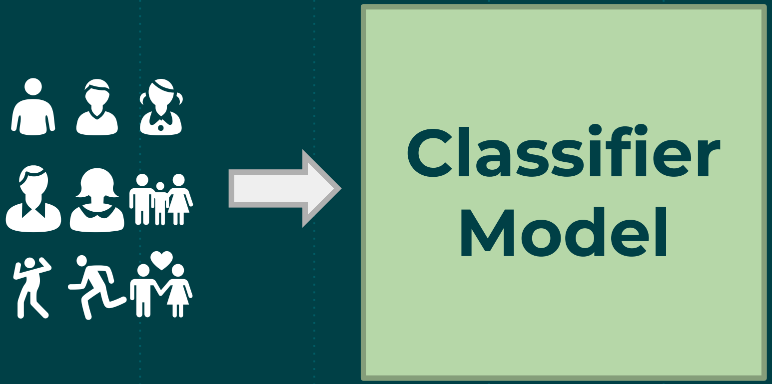
To Catch a Welcher: CLASSIFYING THE DEFAULTERS

Prepared by Seow Xian Jin

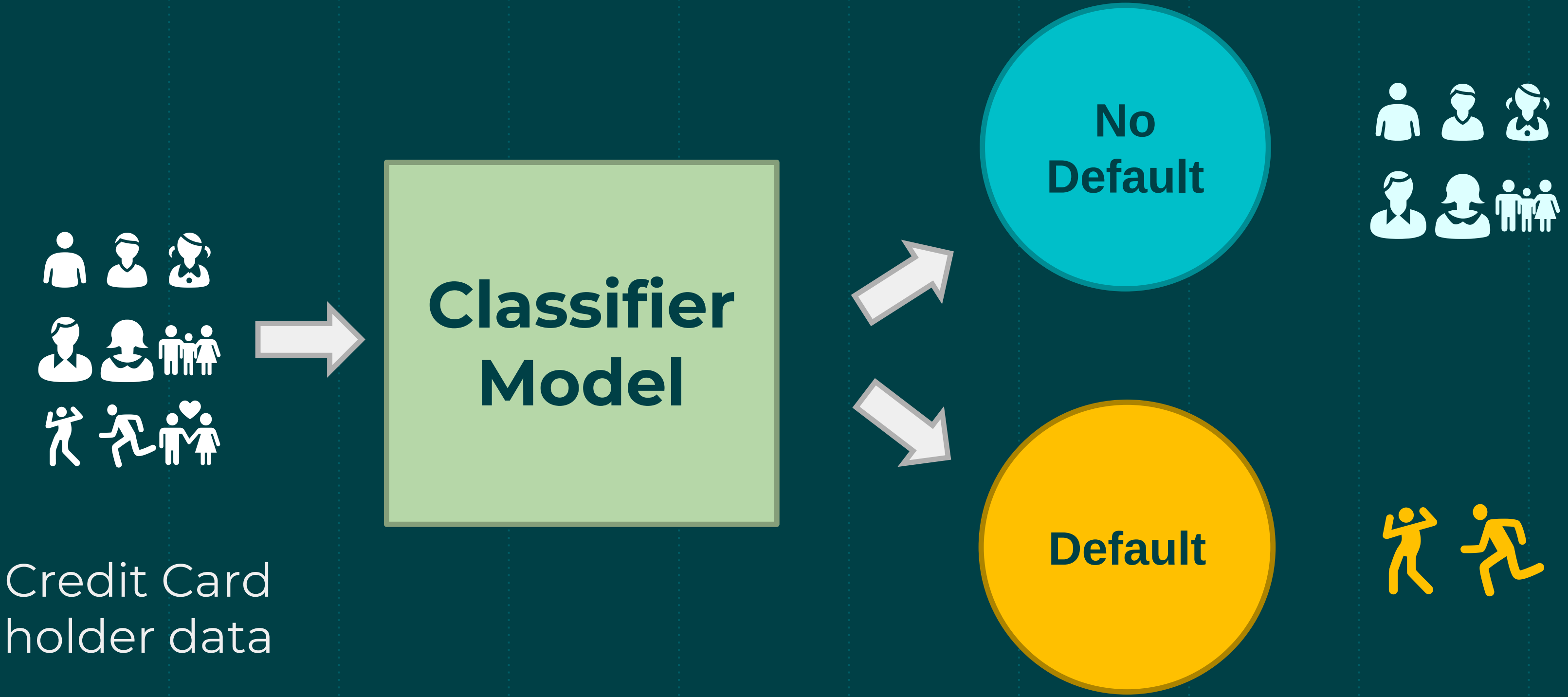
*How do we **properly** assess if someone is going to be a credit risk?*

OBJECTIVE



Credit Card
holder data

OBJECTIVE


Credit Card holder data
Classifier Model
No Default
Default

OBJECTIVE

No Default

*More **consistency** to loaning process*

Default

***Telltale signs** of a potential defaulter*

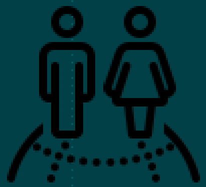
The Data

Dataset information

30,000 clients

24 variables

- Age
- Gender
- Marriage
- Education

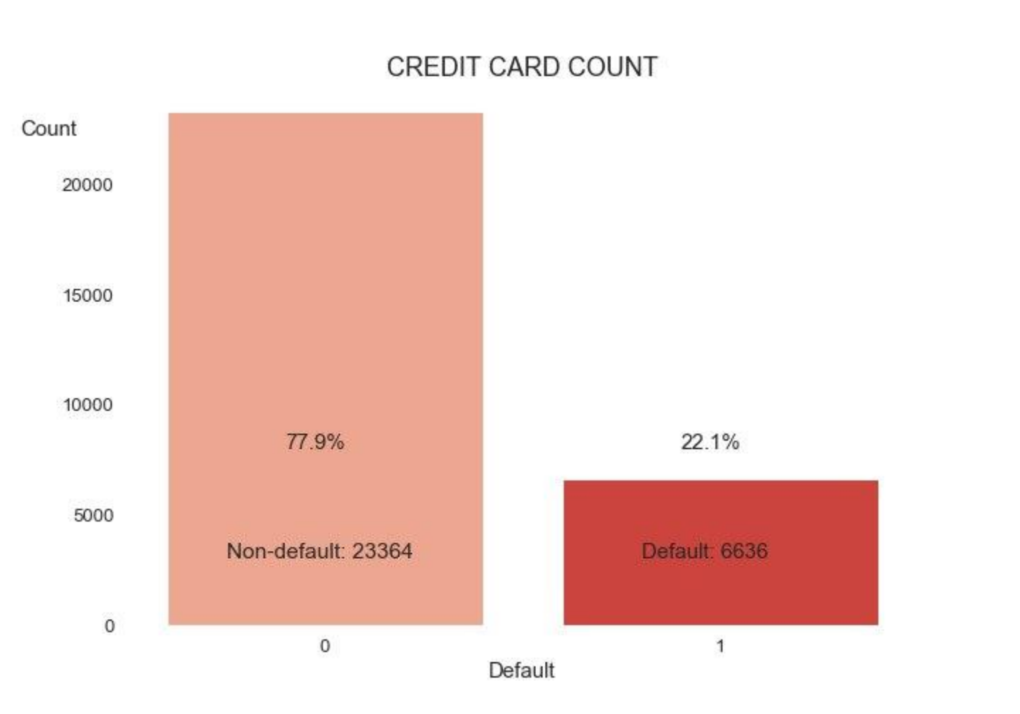


- April to September, 2005
- Bill Statements
- Payment amount
- Repayment status

Data Pre-Processing

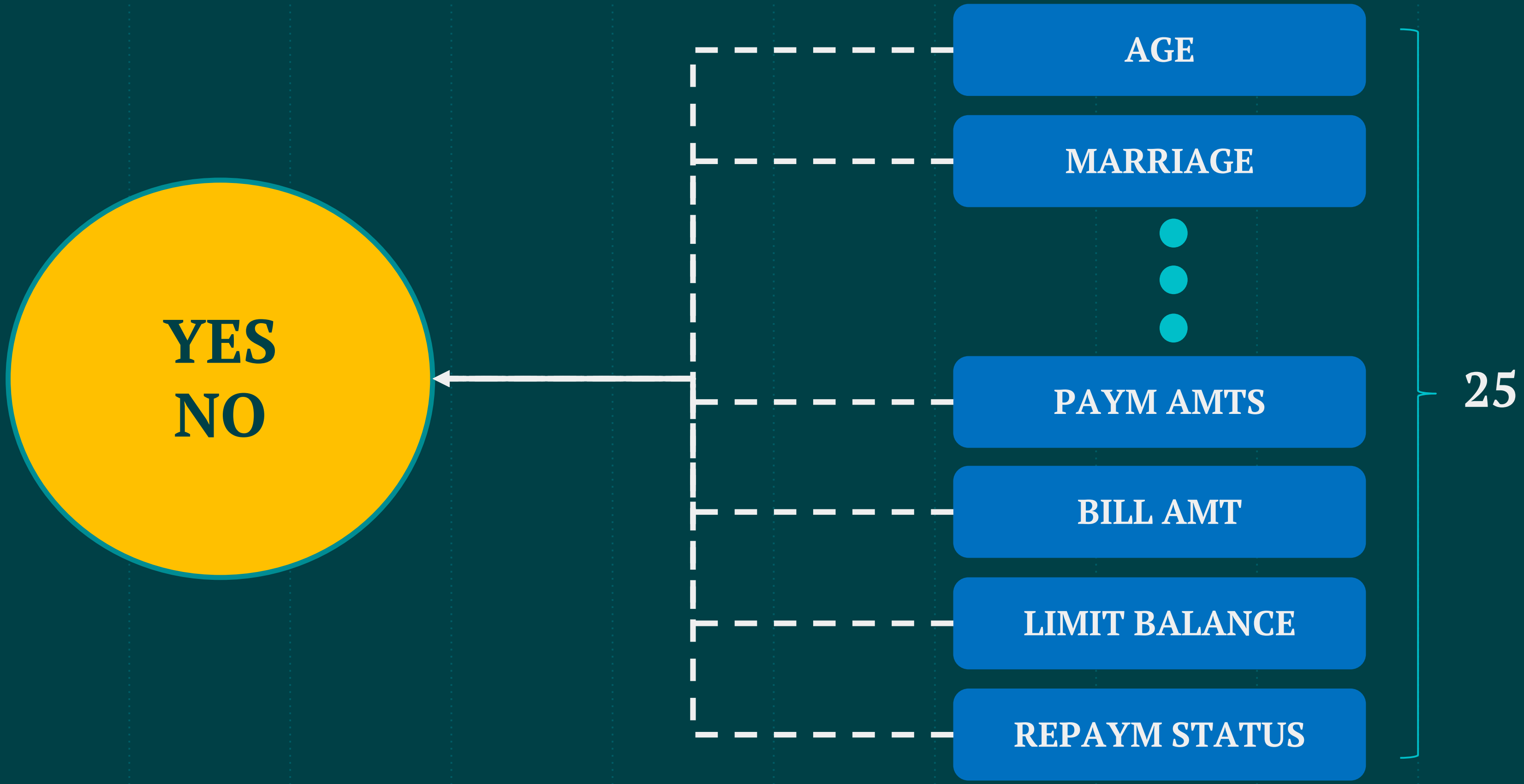
Target Skew

TARGET SKEW



- Class imbalance
- 22% Defaulters
- Taken into account

DATASET FEATURES



Methodology

Models Used

F1

Train/Cross-Validation

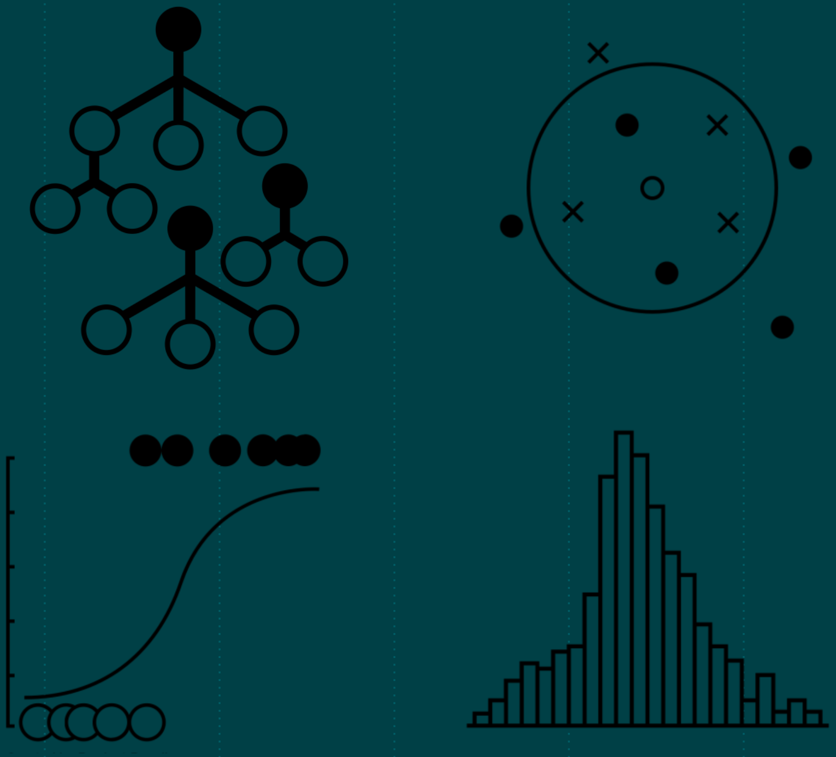
Results of best model from Train/Cross-Validation

Threshold Tuning

Final Train/Test

Models Used

- Gaussian Naïve Bayes
- Logistic Regression
- K-Nearest Neighbors
- Decision Tree
- Random Forest
- Linear SVC



Model Evaluation

- F1 Score
- Harmonic mean of *Precison* and *Recall*
- Best value at 1, worst value at 0

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Recall

Out of ***ALL*** the defaulters, how many did the model actually get right?

Precision

How accurate the model is based on it's own predictions

Model Evaluation

- F1 Score
- Harmonic mean of *Precison* and *Recall*
- Best value at 1, worst value at 0

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Recall

Out of ***ALL*** the defaulters, how many did the model actually get right?

Precision

How accurate the model is based on it's own predictions

Train/Cross-Validation

60% Train Data

20% Holdout Data

20% Validation Data

5-fold Cross-Validation

Train/Cross-Validation

Vanilla Dataset

Models	F1
GaussianNB	0.40
Logistic Reg	0.0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

60% Train | 20% Val | 20% Test

Train/Cross-Validation

Models	F1
GaussianNB	**0.40**
Logistic Reg	**0.0**
K-NN	**0.24**
Decision Tre	**0.39**
Random Forest	**0.40**
Linear SVC	**0.01**

Vanilla Dataset

60% Train | 20% Val | 20% Test

Train/Cross-Validation

Vanilla Dataset

Models	F1
GaussianNB	0.40
Logistic Reg	0.0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

Must improve F1 Score

60% Train | 20% Val | 20% Test

Train/Cross-Validation

Vanilla Dataset

Oversample Dataset

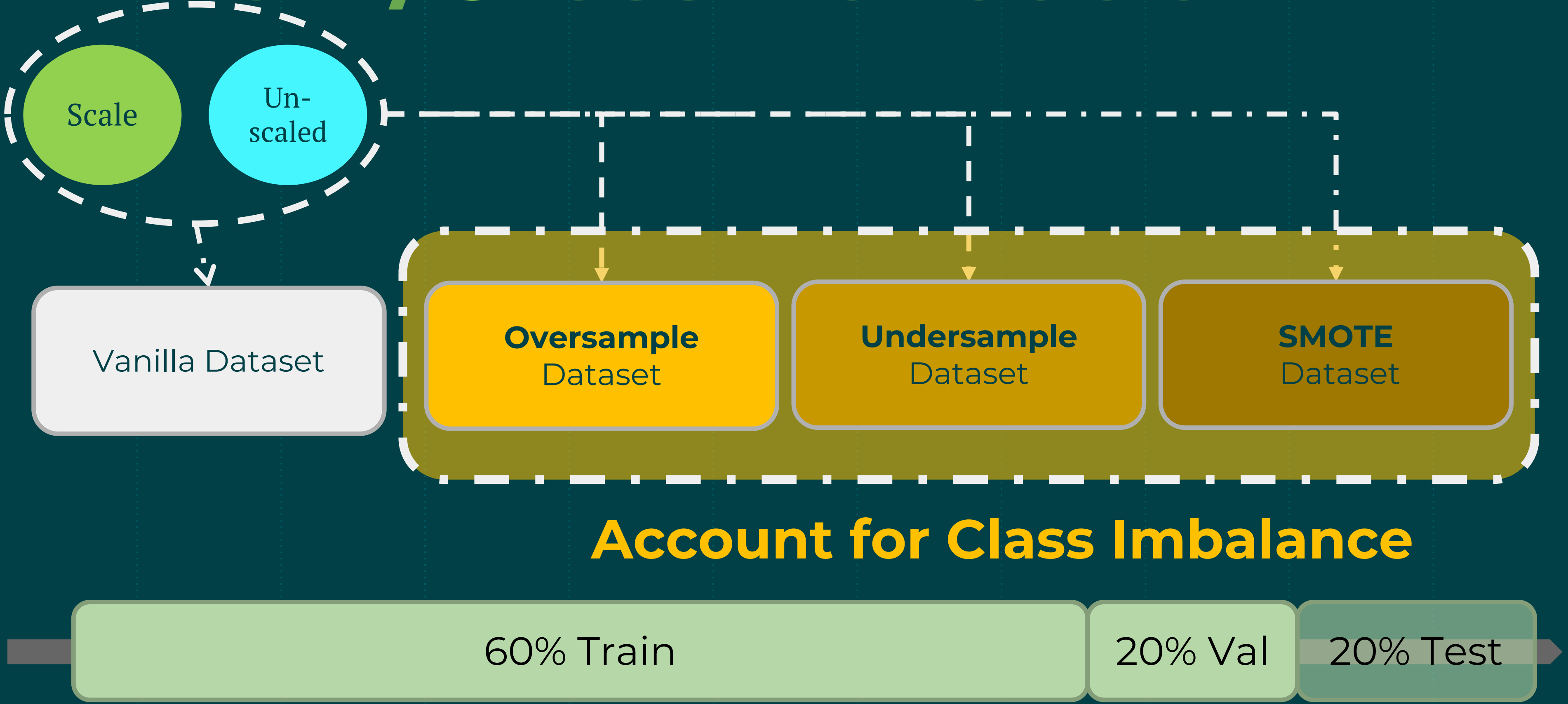
Undersample Dataset

SMOTE Dataset

Account for Class Imbalance

60% Train | 20% Val | 20% Test

Train/Cross-Validation



Preliminary Results

Models	F1
GaussianNB	0.40
Logistic Reg	0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

Vanilla Dataset

60% Train | 20% Val | 20% Test

Preliminary Results

Scale

Vanilla Dataset

Models	F1
GaussianNB	0.40
Logistic Reg	0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

→

F1 Improved
0.52
0.35
0.42
0.40
0.41
0.27

60% Train | 20% Val | 20% Test

Preliminary Results

Scale

Vanilla Dataset

Models	F1	F1 Improved
GaussianNB	**0.40**	**0.52**
Logistic Reg	0	0.35
K-NN	0.24	0.42
Decision Tre	0.39	0.40
Random Forest	0.40	0.41
Linear SVC	0.01	0.27

60% Train | 20% Val | 20% Test

Further increase F1

Scale

Vanilla Dataset

Models	F1
GaussianNB	0.40
Logistic Reg	0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

F1 Improved
0.52
0.35
0.42
0.40
0.41
0.27

Next highest two

60% Train | 20% Val | 20% Test

Further increase F1

Scale

Vanilla Dataset

Models	F1
GaussianNB	0.40
Logistic Reg	0
K-NN	0.24
Decision Tre	0.39
Random Forest	0.40
Linear SVC	0.01

F1 Improved
0.52
0.35
0.42
0.40
0.41
0.27

Hyper-Parameter Tune

60% Train | 20% Val | 20% Test

Post-Hyper Tuning

Scale

Models	F1 Improved
GaussianNB	0.52
K-NN	0.42
Random Forest	0.41

F1 H.Param
0.52
0.42
0.46

Vanilla Dataset

No significant improvement in F1 score

60% Train | 20% Val | 20% Test

Post-Hyper Tuning

Scale

Models	F1 Improved
GaussianNB	0.52
K-NN	0.42
Random Forest	0.4

F1 H.Param
0.52
0.42
0.46

Vanilla Dataset

GaussianNB Chosen
F1 = 0.52

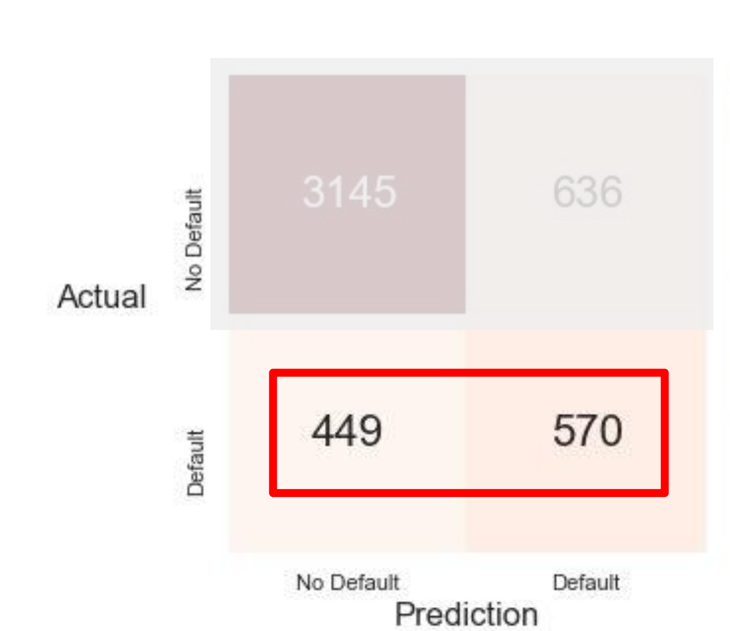
60% Train | 20% Val | 20% Test

GaussianNB Train/CV Results

Threshold = 0.5

60% Train | 20% Val | 20% Test

GaussianNB Train/CV Results



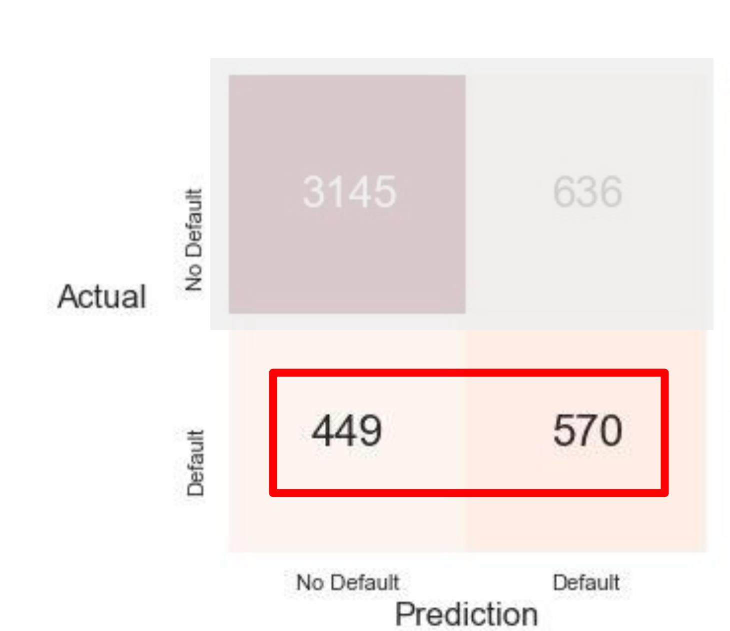
Threshold = 0.5

Test/CV Recall **= 0.56**

Only captures ***56% of all the defaulters***

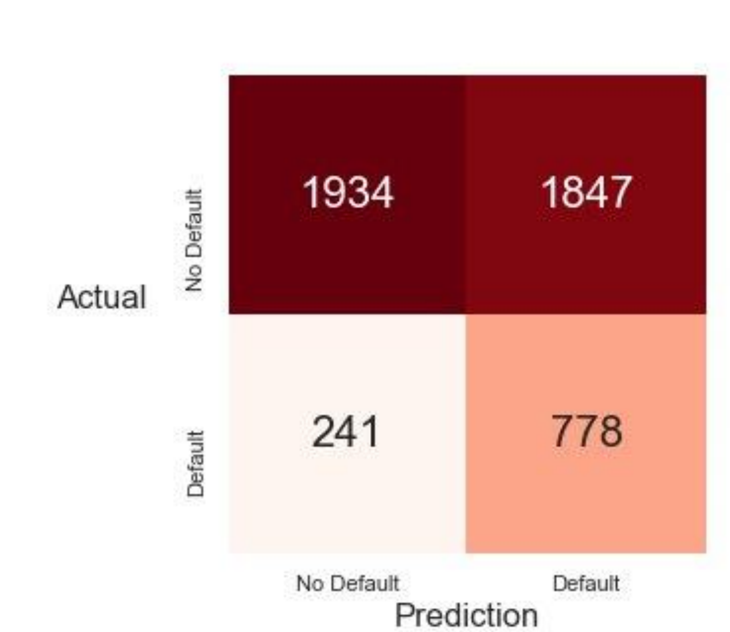
44% not captured

GaussianNB Train/CV Results



Need for further optimization

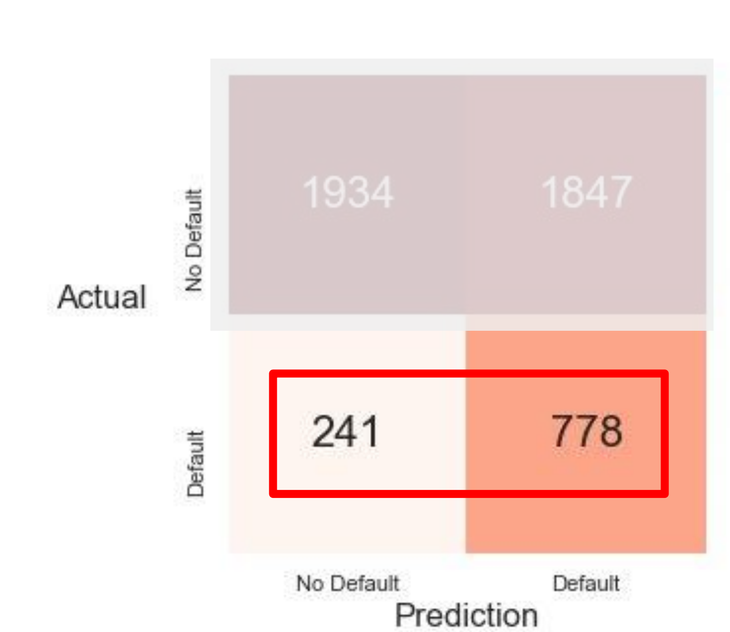
Threshold Optimization



Threshold = **0.25**

60% Train | 20% Val | 20% Test

Threshold Optimization



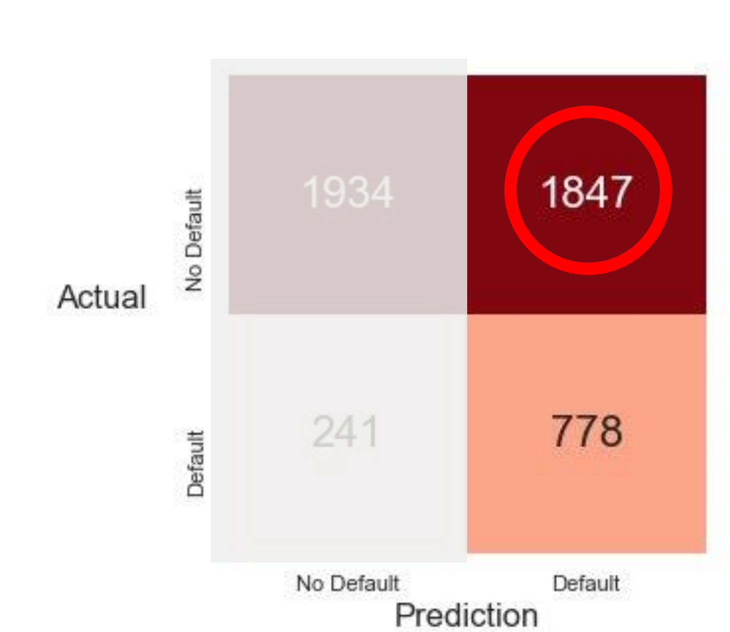
Threshold = **0.25**

Test/CV Recall = **0.76**

Improved to **76%**

60% Train | 20% Val | 20% Test

Threshold Optimization



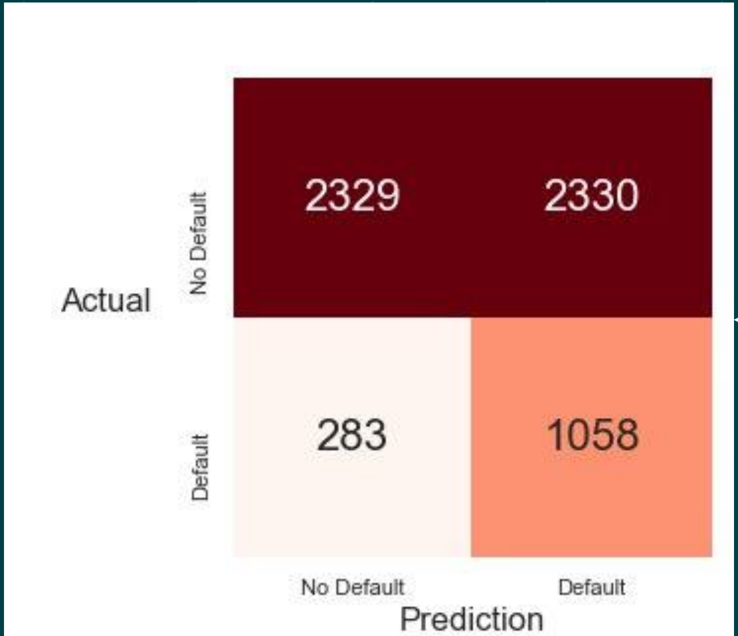
Trade-off: More False Positive Predictions

Higher recall outweighs cost of inconveniencing more people

60% Train | 20% Val | 20% Test

Final Results

Final Results

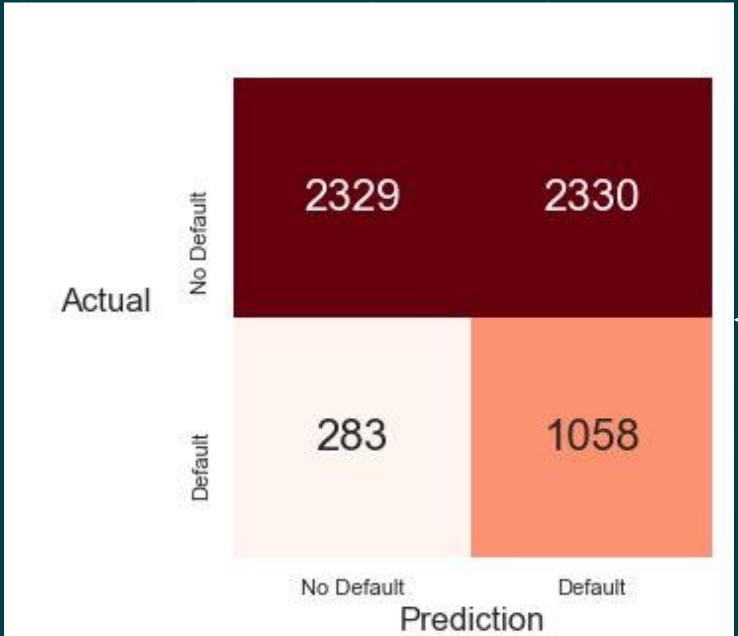


Train on 80% data

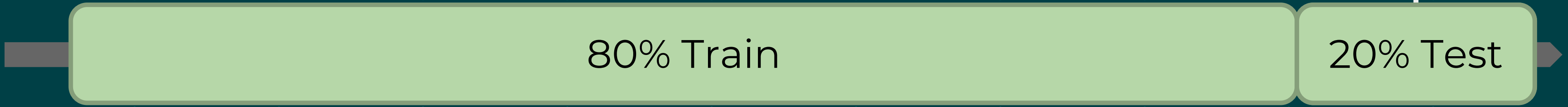
80% Train

20% Test

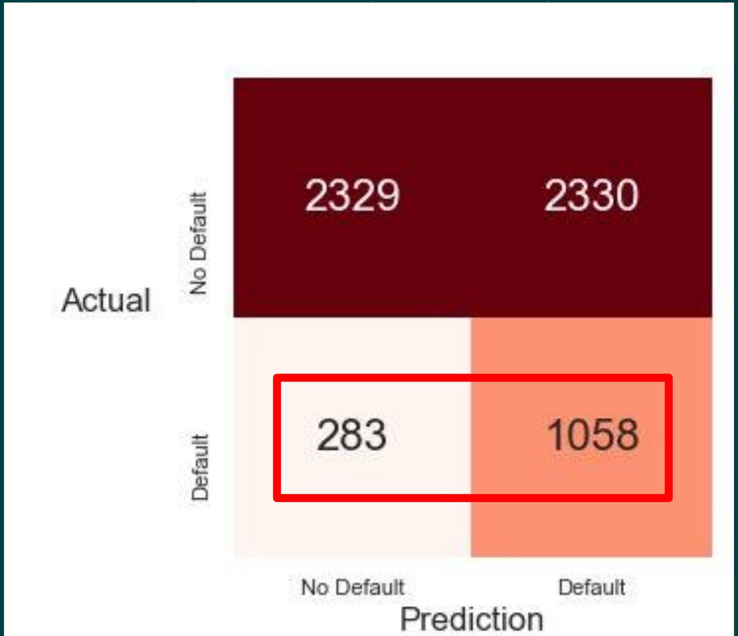
Final Results



Test on 20% **unseen** data



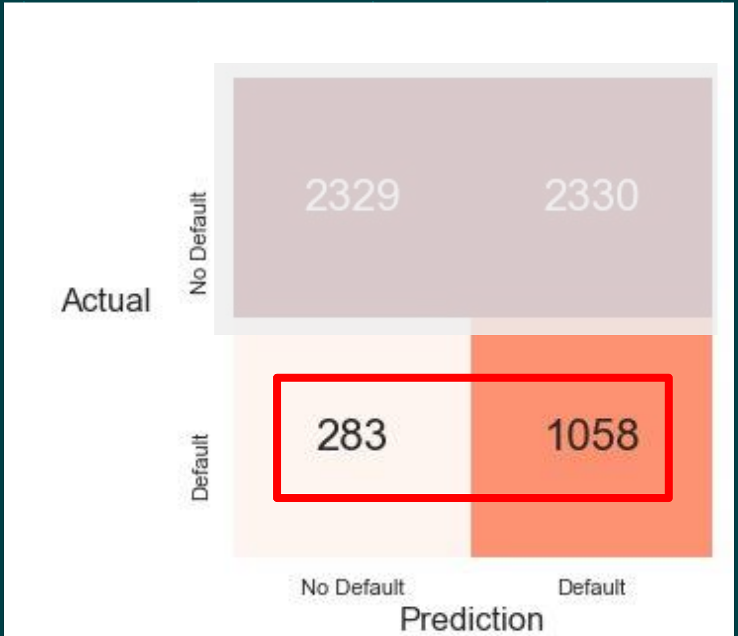
Final Results



80% Train

20% Test

Final Results

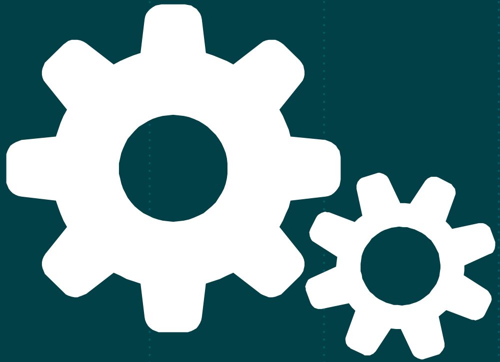


79%

80% Train

20% Test

Future Work



FEATURE ENGINEER

FEATURE SELECTION

HYPERPARAMETER TUNING

OTHER MODELS

Questions?

LINKEDIN

www.linkedin.com/in/seow-xian-jin

GITHUB

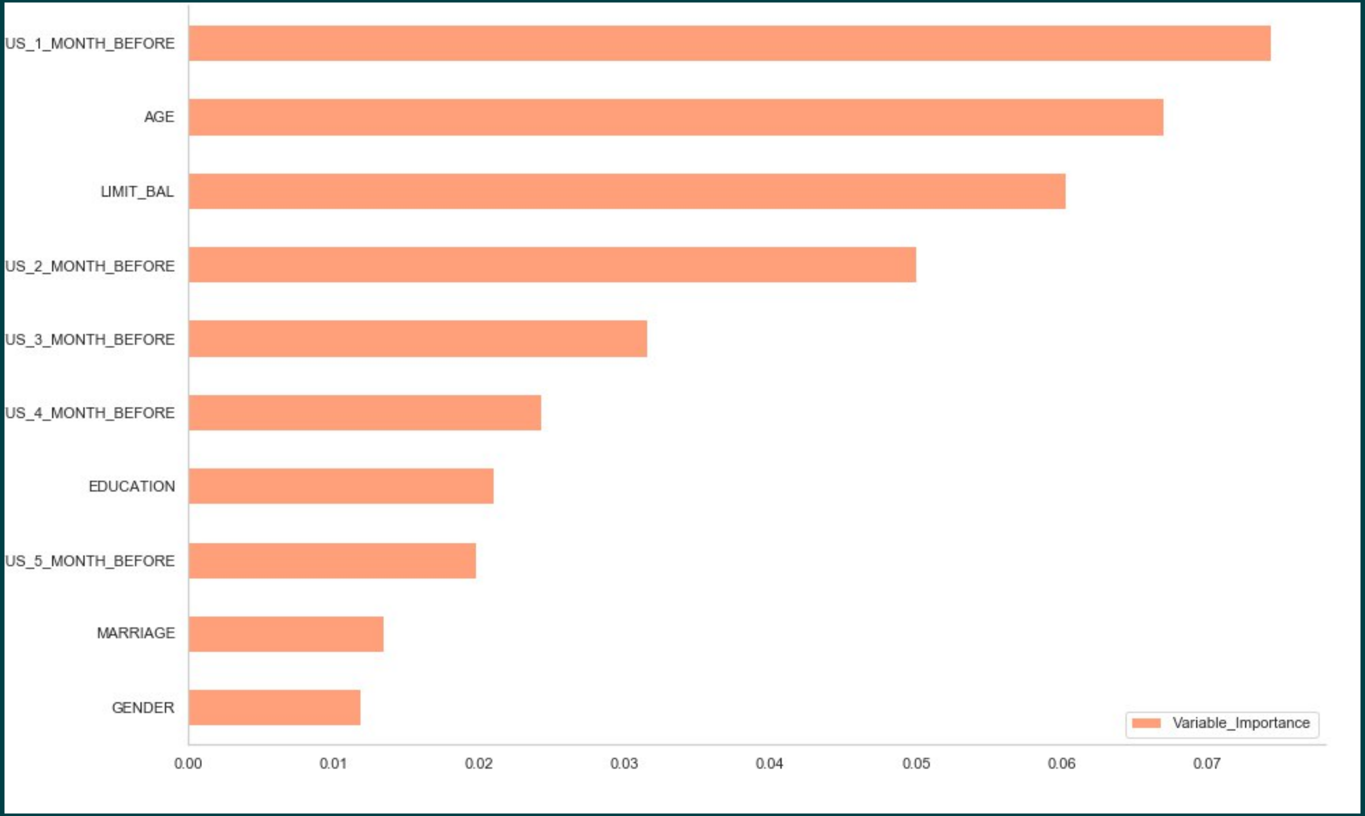
github.com/xianjinseow92/projects

EMAIL ADDRESS

xianjinseow92@gmail.com

APPENDIX

Feature Importance



Model Evaluation

- F1 Score
- Harmonic mean of *Precison* and *Recall*
- Considers both to compute score
- Best value at 1, worst value at 0

Precision

How accurate the model is from all it's own predictions

Recall

Out of all the defaulters, how many did the model actually get right?