UNIVERSITY OF EDINBURGH
Business School

2021-22

CMSE11122 Credit Risk Management

Building and validating predictive models

B199420

Word count: 2021

Contents

1 Introduction **2**

2 Data Pre-processing **2**
2.1 Data Description 2
2.2 Data Cleaning and Sample Selection 2
2.3 Feature Selection 3

3 Model Building **4**

4 Model Evaluation **5**

5 Scorecard Creation **6**

6 Investigation **6**
6.1 Model Comparison 7
6.2 Selecting the optimal cut-off 8

1 Introduction

We are asked to build and validate predictive models in this individual project. We will use the data preprocessed in group assignment 2, which will be introduced in **section 2**. For building models, we will build the Logistic Regression model to predict whether a customer is good or bad and evaluate the model by using AUROC, accuracy, precision, recall, f1 score and confusion matrix. This part will be shown in **section 3 and section 4**. Then, we will create a scorecard in **section 5**. After that, we will build a non-linear classification model and compare the linear classification model with the non-linear one. Finally, we will investigate the problem: selecting the optimal cut-off in **section 6**.

2 Data Pre-processing

In this part, we will discuss the process of data pre-processing, including **data description, data cleaning, sample selection, and feature selection.** This part is completed in group assignment 2 by our group.

2.1 Data Description

We choose the data from Freddie Mac. Freddie Mac is a financial firm that guarantees inexpensive financing for home loans. We use data on credit performance on all mortgages at the loan level bought or insured by this company in 2018. In comparison to other datasets, ours offers two benefits. Firstly, it possesses a legitimate credit history and provenance, as well as a reasonable time range. Additionally, it collects continuous data on loan repayment behavior following loan approval, which may be helpful for later studies. We select 18 variables from 62 original variables, which we think are significant to build the credit risk model. The definition of variable are in Table 1.

Table 1: Variable Defination

Variable	Definition
CREDIT SCORE	the borrower's creditworthiness
FIRST PAYMENT DATE	The date of the first scheduled mortgage payment
FIRST TIME HOMEBUYER FLAG	A Borrower, or one of a group of Borrowers
MATURITY DATE	final monthly payment on the mortgage
NUMBER OF UNITS	The number of unit property
OCCUPANCY STATUS	Mortgage type
ORIGINAL DEBT-TO-INCOME (DTI) RATIO	Disclosure of the debt to income ratio
ORIGINAL UPB	The UPB of the mortgage on the note date
ORIGINAL INTEREST RATE	The original note rate as indicated on the mortgage note
PROPERTY TYPE	The property type
LOAN SEQUENCE NUMBER	Unique identifier assigned to each loan
LOAN PURPOSE	Cash-out, No Cash-out or a Purchase mortgage.
ORIGINAL LOAN TERM	Scheduled monthly payments
HARP INDICATOR	Indicator that identifies whether the loan is HARP or non-harp.
CURRENT LOAN DELINQUENCY STATUS	The number of days the borrower is delinquent
ORIGINAL LOAN-TO-VALUE (LTV)	Mortgage amount divided by property's appraised value
ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	The ratio of a purchase or refinance mortgage loan
DEFAULT	The account is in default or not

2.2 Data Cleaning and Sample Selection

The dataset has two tables: one carrying loan original data and another containing loan performance data on a monthly basis. Both tables contain the same loan-level data elements as the complete dataset. We merge these two datasets based on the same variable: Loan Sequence Number. A single loan sequence number corresponds to a single consumer. According to Z Wang and J Crook(2020) [2], we utilise accounts created within the twelve months of 2018. September 2020 is the date of observation. We consider a client to be in default

if the customer account's payment history records a delinquent of more than 90 days according to Sousa et al [1].

The variable 'Current Loan Delinquency Status' is a qualitative variable, with a value of one unit increasing for every thirty days of delinquency. Variable default was defined as a result of this. If a loan is late for more than 90 days, the borrower is classified as default and considered bad. We use this principle to define the train and test dataset 'DEFAULT' variable. In our dataset, if the customer is 'good,' the 'DEFAULT' variable is 0. Otherwise, the 'DEFAULT' variable is 1.

For the variable **CREDIT SCORE, ORIGINAL COMBINED LOAN-TO-VALUE (CLTV) and ORIGINAL DEBT-TO-INCOME (DTI) RATIO**, there are some missing values in the table. The missing value is shown in Table 2. We delete the instances which contain these values.

Name of Variable	Value of unavailable observations	Variable notes
CREDIT SCORE	9999	9999 = Not Available, if Credit Score is < 300 or > 850.
ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	999	999 = Not Available
ORIGINAL DEBT-TO-INCOME (DTI) RATIO	999	999 = Not Available

Table 2: Missing Values

Finally, we calculate the number of instances of good and bad customers in the datasets, which is shown in Table 3.

	Number of goods	Number of bads
Dataset	1200036	74667

Table 3: Datasets Instances

As the number of instances in the dataset is so large, we only choose **200,000** instances from the dataset, containing all the bad samples in the dataset.

2.3 Feature Selection

In this part, we will calculate the WOE and IV of each variable that we selected before. Depending on the variable type, feature selection is divided into two parts. For numeric type variables, we first divide a variable into ten roughly equal groups. And for each group, we calculate the odds. However, considering the prediction accuracy, the number of groups for each variable does not exceed 7 (except CREDIT SCORE). For feature selection, we calculate the WOE of each variable class and the IV of each variable and select the best variable as the model's feature. Our team members implement these processes through the SAS language.

We pick variables of which the value is more significant than 0.01. As the value of IV is shown in Table 4, we finally choose six variables:**'CREDIT SCORE' , 'ORIGINAL DEBT-TO-INCOME (DTI) RATIO' , 'ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)' , 'ORIGINAL INTEREST RATE' , 'FIRST TIME HOMEBUYER FLAG' and 'OCCUPANCY STATUS'.**

Variable	IV
CREDIT SCORE	0.549
ORIGINAL DEBT-TO-INCOME (DTI) RATIO	0.181
ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	0.166
ORIGINAL INTEREST RATE	0.121
FIRST TIME HOMEBUYER FLAG	0.020
OCCUPANCY STATUS	0.010
NUMBER OF UNITS	0.007
PROPERTY TYPE	0.003
LOAM PURPOSE	0.006
HARP INDICATOR	0.008
OROGINAL UPB	0.002

Table 4: IV Value

3 Model Building

In this part, we will build the logistic regression model to predict whether a customer is good or bad. After feature selection, we select six features that we can use to train the model. However, some of these features are not numerical variables. We should convert these variables to numerical ones according to the coarse class of each variable. We use the data after coarse classification to train our model. The overall data is shown in Figure 1.

```
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 34 columns):
 #   Column                                                Non-Null Count   Dtype
---  ------                                                --------------   -----
 0   CREDIT SCORE:(577.739, 604.1]                         200000 non-null  int64
 1   CREDIT SCORE: (604.1, 630.2]                          200000 non-null  int64
 2   CREDIT SCORE: (656.3, 682.4]                          200000 non-null  int64
 3   CREDIT SCORE: (812.9, 839.0]                          200000 non-null  int64
 4   CREDIT SCORE: (786.8, 812.9]                          200000 non-null  int64
 5   CREDIT SCORE: (734.6, 760.7]                          200000 non-null  int64
 6   CREDIT SCORE: (630.2, 656.3]                          200000 non-null  int64
 7   CREDIT SCORE: (682.4, 708.5]                          200000 non-null  int64
 8   CREDIT SCORE: (708.5, 734.6]                          200000 non-null  int64
 9   CREDIT SCORE: (760.7, 786.8]                          200000 non-null  int64
 10  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (8.143, 15.286]  200000 non-null  int64
 11  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (0.95, 8.143]    200000 non-null  int64
 12  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (15.286, 22.429] 200000 non-null  int64
 13  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (29.571, 36.714] 200000 non-null  int64
 14  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (22.429, 29.571] 200000 non-null  int64
 15  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (43.857, 51.0]   200000 non-null  int64
 16  ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (36.714, 43.857] 200000 non-null  int64
 17  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (3.899, 18.429]  200000 non-null  int64
 18  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (32.857, 47.286]  200000 non-null  int64
 19  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (18.429, 32.857]  200000 non-null  int64
 20  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (47.286, 61.714]  200000 non-null  int64
 21  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (76.143, 90.571]  200000 non-null  int64
 22  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (90.571, 105.0]   200000 non-null  int64
 23  ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (61.714, 76.143]  200000 non-null  int64
 24  ORIGINAL INTEREST RATE: (2.496, 3.375]                200000 non-null  int64
 25  ORIGINAL INTEREST RATE: (6.0, 6.875]                  200000 non-null  int64
 26  ORIGINAL INTEREST RATE: (5.125, 6.0]                  200000 non-null  int64
 27  ORIGINAL INTEREST RATE: (4.25, 5.125]                 200000 non-null  int64
 28  ORIGINAL INTEREST RATE: (3.375, 4.25]                 200000 non-null  int64
 29  FIRST TIME HOMEBUYER FLAG:0                           200000 non-null  int64
 30  FIRST TIME HOMEBUYER FLAG:1                           200000 non-null  int64
 31  OCCUPANCY STATUS:I                                    200000 non-null  int64
 32  OCCUPANCY STATUS:P                                    200000 non-null  int64
 33  OCCUPANCY STATUS:S                                    200000 non-null  int64
dtypes: int64(34)
memory usage: 51.9 MB
```

Figure 1: Data

We split the dataset into train and test datasets. we split 20% of all data into the test set and the remaining 80% on the train set. After that, we can build the logistic regression model. We use Python language and

scikit-learn package to help us build the model as it is straightforward to do it with Python language. We use the train dataset to fit the model. After several minutes, we build the model already.(Figure 2)

```
#build the model
lr = LogisticRegression()
lr.fit(Xtrn,Ytrn)
```

Figure 2: LR model

4 Model Evaluation

In this section, we will use Area Under the Receiver Operating Characteristics Curve(AUROC), Confusion matrix, accuracy, precision, recall and f1 score to evaluate our model. The AUROC is shown in Figure 4 and Confusion Matrix is shown in Figure 3. The values of accuracy, precision, recall, and f1 score are shown in Table 5.

Accuracy	Precision	Recall	F1-score
0.6887	0.6631	0.6631	0.6420

Table 5: LR Score

		Actual Values	
Predictive Values		Positive	Negative
	Positive	6848	4487
	Negative	7967	20698

Figure 3: LR Confusion Matrix

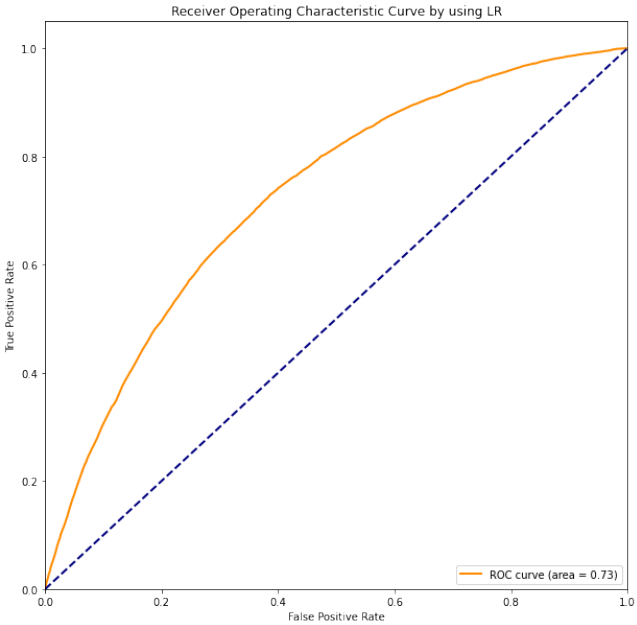

Figure 4: LR AUROC

From Figure 4, we can see that the AUROC is 0.73. From Table 5, we can find the accuracy of the model is 0.6618 and f1 score of the model is 0.6490. From the confusion matrix, we can see the True Positive, True Negative, False Positive and False Negative values. We think the Logistic Regression model is acceptable from the evaluation information above. Maybe other models have higher AUROC, accuracy, and f1 score.

5 Scorecard Creation

The assignment's last stage is to create a readable and acceptable scorecard. In the Logistic Regression model, we calculate the coefficients for each feature. Then, we establish our scorecard's minimum and maximum thresholds. The minimum is 300. The highest value is 850. Following that, we add the lowest and maximum coefficients for each category contained inside the original feature name. The score is calculated by multiplying the coefficient by the ratio of the differences between the highest and minimum score and the maximum and minimum sum of the coefficients. Finally, we update the Intercept's estimated score (i.e. the default score for each loan), as seen in Figure 5. This final scorecard indicates that each user has a base score of 589, which should be increased or decreased based on the person's unique feature values.

	Feature name	Coefficients	original feature name	Score - Calculation	Score - Preliminary
0	Intercept	-0.003808	Intercept	589.019728	589.0
1	CREDIT SCORE:(577.739, 604.1]	1.455858	CREDIT SCORE	134.591949	135.0
2	CREDIT SCORE: (604.1, 630.2]	1.208438	CREDIT SCORE	111.718334	112.0
3	CREDIT SCORE: (656.3, 682.4]	0.571144	CREDIT SCORE	52.801408	53.0
4	CREDIT SCORE: (812.9, 839.0]	-1.593104	CREDIT SCORE	-147.280218	-147.0
5	CREDIT SCORE: (786.8, 812.9]	-1.387465	CREDIT SCORE	-128.269125	-128.0
6	CREDIT SCORE: (734.6, 760.7]	-0.445164	CREDIT SCORE	-41.154759	-41.0
7	CREDIT SCORE: (630.2, 656.3]	0.955047	CREDIT SCORE	88.292680	88.0
8	CREDIT SCORE: (682.4, 708.5]	0.275620	CREDIT SCORE	25.480650	25.0
9	CREDIT SCORE: (708.5, 734.6]	-0.105194	CREDIT SCORE	-9.725049	-10.0
10	CREDIT SCORE: (760.7, 786.8]	-0.909543	CREDIT SCORE	-84.085929	-84.0
11	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (8.143, 1...	-0.420913	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	-38.912810	-39.0
12	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (0.95, 8....	0.099754	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	9.222122	9.0
13	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (15.286, ...	-0.397712	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	-36.767894	-37.0
14	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (29.571, ...	0.023157	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	2.140794	2.0
15	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (22.429, ...	-0.165917	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	-15.338783	-15.0
16	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (43.857, ...	0.570259	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	52.719595	53.0
17	ORIGINAL DEBT-TO-INCOME (DTI) RATIO: (36.714, ...	0.317008	ORIGINAL DEBT-TO-INCOME (DTI) RATIO	29.306917	29.0
18	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (3.899...	-0.550997	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	-50.938838	-51.0
19	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (32.85...	-0.359862	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	-33.268683	-33.0
20	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (18.42...	-0.499405	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	-46.169308	-46.0
21	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (47.28...	-0.251969	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	-23.294157	-23.0
22	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (76.14...	0.000000	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	0.000000	0.0
23	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (90.57...	0.000000	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	0.000000	0.0
24	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV): (61.71...	0.000000	ORIGINAL COMBINED LOAN-TO-VALUE (CLTV)	0.000000	0.0
25	ORIGINAL INTEREST RATE: (2.496, 3.375]	-0.235734	ORIGINAL INTEREST RATE	-21.793265	-22.0
26	ORIGINAL INTEREST RATE: (6.0, 6.875]	0.424573	ORIGINAL INTEREST RATE	39.251168	39.0
27	ORIGINAL INTEREST RATE: (5.125, 6.0]	0.139241	ORIGINAL INTEREST RATE	12.872664	13.0
28	ORIGINAL INTEREST RATE: (4.25, 5.125]	-0.078987	ORIGINAL INTEREST RATE	-7.302235	-7.0
29	ORIGINAL INTEREST RATE: (3.375, 4.25]	-0.223458	ORIGINAL INTEREST RATE	-20.658390	-21.0
30	FIRST TIME HOMEBUYER FLAG:0	-0.124187	FIRST TIME HOMEBUYER FLAG	-11.480931	-11.0
31	FIRST TIME HOMEBUYER FLAG:1	0.149822	FIRST TIME HOMEBUYER FLAG	13.850873	14.0
32	OCCUPANCY STATUS:I	0.004510	OCCUPANCY STATUS	0.416922	0.0
33	OCCUPANCY STATUS:P	0.222466	OCCUPANCY STATUS	20.566686	21.0
34	OCCUPANCY STATUS:S	-0.201341	OCCUPANCY STATUS	-18.613666	-19.0

Figure 5: Scorecard

6 Investigation

In this section, we will use a non-linear model to make a comparison and investigate the optimal cut-off.

6.1 Model Comparison

We want to use a non-linear model to compare because Logistic Regression model is a linear model. We would like to see the differences between linear model and non-linear model. In this case, we use the SVM model with a 'polynomial' kernel to compare with the Logistic Regression model. We calculate the Area Under the Receiver Operating Characteristics Curve(AUROC), Confusion matrix, accuracy, precision, recall and f1 score to compare the two models. The AUROC is shown in Figure 6 , the Confusion Matrix is shown in Figure 7. The values of accuracy, precision, recall and f1 score are shown in Table 6.

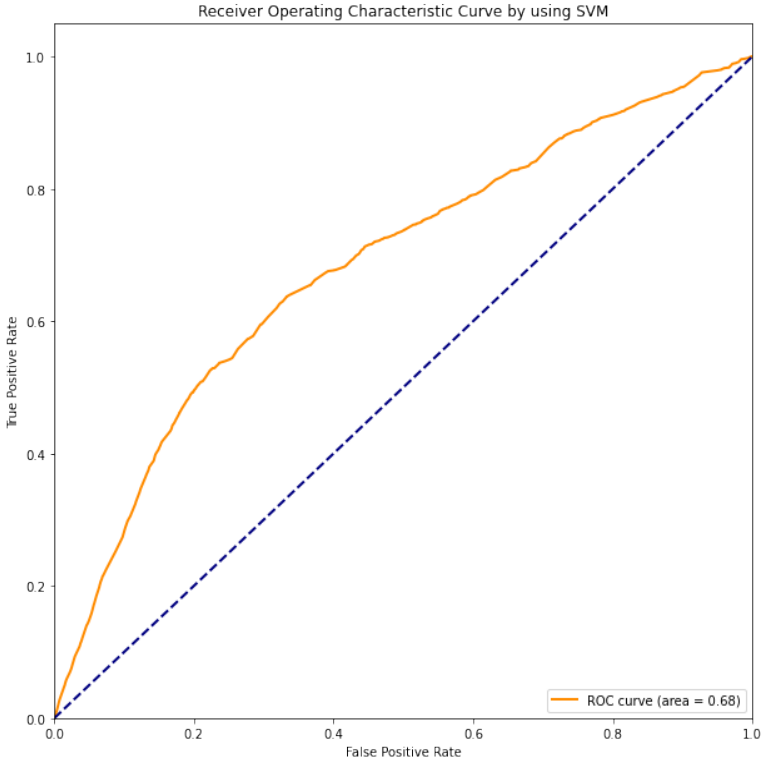

Figure 6: SVM AUROC

Accuracy	Precision	Recall	F1-score
0.6878	0.6618	0.6450	0.6490

Table 6: SVM Score

		Actual Values	
Predictive Values		Positive	Negative
	Positive	7109	4783
	Negative	7706	20402

Figure 7: SVM Confusion Matrix

Compared with the Logistic Regression model, the SVM accuracy, precision, recall and f1 score are lower. At the same time, the AUROC of the Logistic Regression is higher. According to the confusion matrix, the two models perform similarly. We think that the Logistic Regression model performs better than the SVM model. The AUROC and accuracy, precision, recall and F1 score of logistic Regression model are higher than those of SVM model. However, based on our research of academic resources, we believe that both are acceptable ROC results in the credit scoring domain.

6.2 Selecting the optimal cut-off

In this part, we use different methods to select the best cut-off for our model. These methods include Youden's J statistic, precision-Recall curve and threshold tuning.

Youden's J statistics is one of the metrics that will be explored. By optimising Youden's J statistics, the classification threshold will be determined. The code is in Figure 8. Precision-recall curves are graphs that illustrate the link between precision and recall. We calculate the F1 score to find the optimal threshold. The code is in Figure 9.Threshold tuning is a widely used approach for determining the best-unbalanced classification threshold. The sequence of thresholds is determined by the researcher's requirements, whereas prior methodologies used the Receiver Operating Characteristics and Precision and Recall to construct a series of those thresholds. The advantages are that the threshold sequence may be customised to meet specific requirements, albeit at a greater computational cost. The code is in Figure 10.

```
# Calculate the Youden's J statistic
youdenJ = tpr - fpr

# Find the optimal threshold
index = np.argmax(youdenJ)
thresholdOpt = round(thresholds[index], ndigits = 4)
youdenJOpt = round(gmean[index], ndigits = 4)
fprOpt = round(fpr[index], ndigits = 4)
tprOpt = round(tpr[index], ndigits = 4)
print('Best Threshold: {} with Youden J statistic: {}'.format(thresholdOpt, youdenJOpt))
print('FPR: {}, TPR: {}'.format(fprOpt, tprOpt))
```

```
Best Threshold: 0.2966 with Youden J statistic: 0.6576
FPR: 0.3822, TPR: 0.7
```

Figure 8: Youden's J threshold

```
# Calculate the f-score
fscore = (2 * precision * recall) / (precision + recall)

# Find the optimal threshold
index = np.argmax(fscore)
thresholdOpt = round(thresholds[index], ndigits = 4)
fscoreOpt = round(fscore[index], ndigits = 4)
recallOpt = round(recall[index], ndigits = 4)
precisionOpt = round(precision[index], ndigits = 4)
print('Best Threshold: {} with F-Score: {}'.format(thresholdOpt, fscoreOpt))
print('Recall: {}, Precision: {}'.format(recallOpt, precisionOpt))
```

```
Best Threshold: 0.9016 with F-Score: nan
Recall: 0.0, Precision: 0.0
```

Figure 9: f-score threshold

```python
#threshold tuning
# Array for finding the optimal threshold
thresholds = np.arange(0.0, 1.0, 0.0001)
fscore = np.zeros(shape=(len(thresholds)))
print('Length of sequence: {}'.format(len(thresholds)))

# Fit the model
for index, elem in enumerate(thresholds):
    # Corrected probabilities
    y_pred_prob = (y_pred > elem).astype('int')
    # Calculate the f-score
    fscore[index] = f1_score(Ytst, y_pred_prob)

# Find the optimal threshold
index = np.argmax(fscore)
thresholdOpt = round(thresholds[index], ndigits = 4)
fscoreOpt = round(fscore[index], ndigits = 4)
print('Best Threshold: {} with F-Score: {}'.format(thresholdOpt, fscoreOpt))
```

```
Length of sequence: 10000
Best Threshold: 0.2074 with F-Score: 0.6854
```

Figure 10: Tuning threshold

The results of the four threshold methods are in Table 7. The ideal threshold for binary classification is 0.2966, as determined using Youden's J measure. This suggests that the optimal probability threshold, based on the Youden's J statistic, is 0.2966, which minimises the FPR and maximises the TPR - all samples with a projected probability greater than this should be categorised as in Default, and vice versa. When using the Precision-Recall curve and F1-score, it generates a threshold of 0.9016 for deciding whether an observation belongs to the good or bad class. This threshold is too high because of its approaches, which means we should reject the majority of the customers. Threshold Tuning attempts to determine the best threshold with the subject using a looping method in order to maximize the F1 score as an unbiased statistic. Finally, the looping process was terminated and the ideal threshold of 0.2074 was written out. This value is quite close to the threshold created by Youden's J.

Method	Optimal Threshold
Youden's J	0.2966
F-score	0.9016
Threshold Tuning	0.2074

Table 7: Threshold

From the results above, we can see a big difference between the thresholds calculated by different methods. The threshold value calculated using F1-score is very unreasonable. It means that we should The values computed using Youden's J and Threshold Tuning are very close. The large difference in cut-off values calculated by different methods may be because the dataset has fewer features used in the Logistic Regression model. What's more, the imbalanced dataset can also lead to the difference.

We think 0.2966 or 0.2074 can be chosen as the cut-off. After selecting the ideal optimal thresholds, we have to find the corresponding acceptance and rejection rates on the test dataset. Assuming that all credit applications above a given "good" probability will be approved, we calculate the number of approved applications and the number of rejected applications separately for all thresholds. The approval rate is equal to the ratio of approved applications to all applications. The rejection rate is equal to one minus the approval rate. We look at the approval rate and rejection rate corresponding to the value of the threshold found using Youden's J statistic(Figure 12) and Thredhold Tuning (Figure 10) respectively. We find that when the threshold value is 0.2074, the rejection rate is lower, which does not bring huge losses to the business.

```
df_cutoffs[df_cutoffs['thresholds'].between(0.2966, 0.2967)]
```

	thresholds	Score	N Approved	N Rejected	Approval Rate	Rejection Rate
12026	0.296686	510.0	24971	15029	0.624275	0.375725
12027	0.296682	510.0	24972	15028	0.624300	0.375700

Figure 11: Threshold and Credit Score using Youden's J

```
df_cutoffs[df_cutoffs['thresholds'].between(0.2074, 0.2075)]
```

	thresholds	Score	N Approved	N Rejected	Approval Rate	Rejection Rate
14577	0.207429	465.0	31587	8413	0.789675	0.210325
14578	0.207413	465.0	31588	8412	0.789700	0.210300

Figure 12: Threshold and Credit Score using Threshold Tuning

In conclusion, we decide to use 0.2074 as the threshold. And the corresponding Credit Score is 465.

References

[1] Maria Rocha Sousa, João Gama, and Elísio Brandão. Dynamic credit score modeling with short-term and long-term memories: the case of freddie mac's database. 2016.

[2] Zheqi Wang, Jonathan Crook, and Galina Andreeva. Reducing estimation risk using a bayesian posterior distribution approach: Application to stress testing mortgage loan default. *European Journal of Operational Research*, 287(2):725–738, 2020.