# NATIONAL ECONOMICS UNIVERITY

-------***-------

## RISK MANAGEMEMNT ASSIGNMENT:

# Credit Risk Analysis for extending Bank Loans

**Full name : Trần Thị Thu Trang**
**Class : 62DSEB**
**Code student :11208164**
**Instructor : Nguyen Thi Lien**

**Ha Noi , 08 / 02 / 2023**

# I- INTRODUCTION

Investigational research demonstrates the link between credit scoring and the bank's approval of loan extensions. Credit scoring is perhaps one of the most "classic" applications for predictive modeling, to predict whether or not credit extended to an applicant will likely result in profit or losses for the lending institution. The collection includes details on potential bank loan applicants and their corresponding credit scores. It contains information about employment, earnings, age, education, marital status, possessions, liabilities, and other financial indicators. In order to reduce loan risk, the goal is to estimate the borrower's creditworthiness based on these variables. The dataset may be utilized for credit risk analysis, machine learning, and classification and regression modeling.

# II- THEORETICAL BACKGROUND

## 1. Weight of Evidence (WOE) .

The weight of evidence tells the predictive power of an independent variable in relation to the dependent variable. Since it evolved from credit scoring world, it is generally described as a measure of the separation of good and bad customers.

$$WOE = ln\left(\frac{Distribution\ of\ Goods}{Distribution\ of\ Bads}\right)$$

If you do not understand the terms goods/bads as we are from different background than the credit risk. It's good to understand the concept of WOE in terms of events and non-events.

$$WOE = \ln\left(\frac{\%\ of\ on - events}{\%\ of\ events}\right)$$

## 2. Information Value (IV).

Information value is one of the most useful technique to select important variables in a predictive model. It helps to rank variables on the basis of their importance.

$$IV = \sum(\%\ of\ non - event - \%\ of\ event) * WOE$$

Rulse aleatd to IV:

| Information | Variable predictiveness |
| --- | --- |
| < 0,02 | Useless for making predictions |
| 0,02 – 0,1 | Poor Predictive Power |
| 0,1 – 0,3 | Medium predictive Power |
| 0,2 - 0,5 | Strong predictive Power |
| > 0,5 | Suspicious Predictive Power |

3. **Logistic (orginal – WOE).**

**Table 1. Probability of default**

| 0 | 1 |
| --- | --- |
| 0.739 | 0.261 |

The results in this table show There is a class bias, a condition observed when the proportion of events is much smaller than proportion of non-events. So we must sample the observations in approximately equal proportions to get better models.

***Using the Logit model in credit scoring:***

Split the dataset into train 70% and test 30%.

a) Train data.

```
> train.data
# A tibble: 498 × 9
     age    ed employ address income debtinc creddebt
   <dbl> <dbl>  <dbl>   <dbl>  <dbl> <chr>   <chr>
 1    41     3     17      12    176 9.3     11.3593…
 2    40     1     15      14     55 5.5     0.856075
 3    41     1     15      14    120 2.9     2.65872
 4    41     2      5       5     25 10.2    0.3927
 5    36     1      0      13     25 19.7    2.7777
 6    27     1      0       1     16 1.7     0.182512
 7    25     1      4       0     23 5.2     0.252356
 8    52     1     24      14     64 10      3.9296
 9    37     1      6       9     29 16.3    1.715901
10    36     2      9       6     49 8.6     0.817516
# … with 488 more rows, and 2 more variables:
#   othdebt <chr>, default <dbl>
# ℹ Use `print(n = ...)` to see more rows, and `colnames()` to see
all variable names
```

The Inmation Value (IV) for index of the variables on the set train.data:

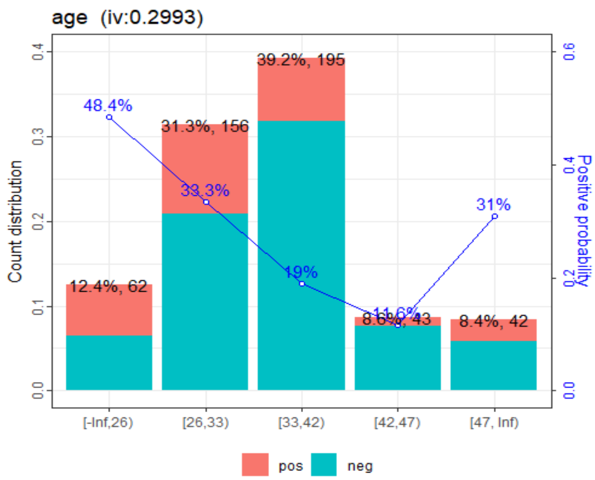
**Table 2. IV in train data**

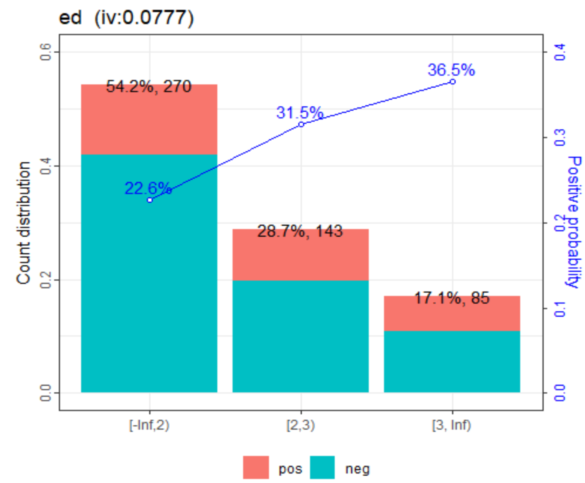
| | Variable | IV |
|---|---|---|
| 1 | age | 0.21879511 |
| 2 | ed | 0.07928365 |
| 3 | employ | 0.59223715 |
| 4 | address | 0.33592954 |
| 5 | income | 0.15465404 |
| 6 | debtinc | 0.49148165 |
| 7 | creddebt | 0.00366636 |
| 8 | othdebt | 0.00366636 |

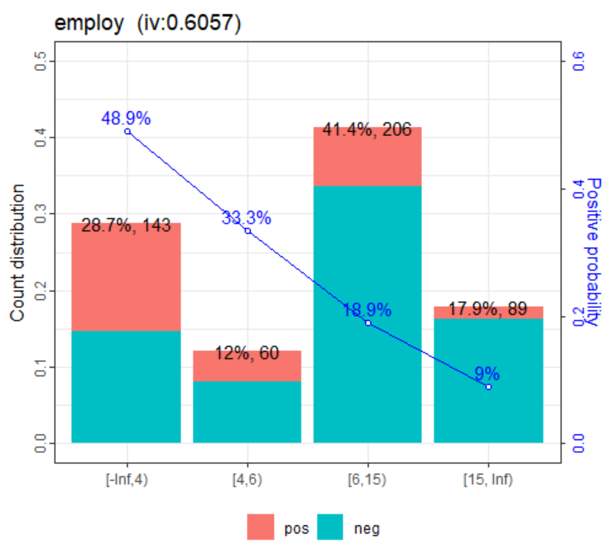
According to the above table the predictor is useless for modeling if the IV statistic is less than 0.02. (separating the non-default part from the default).So we will remove the columns less than 0.02 because it does not help us in the analysis.
We have a new train.data table after deleting columns with IVs less than 0.02.

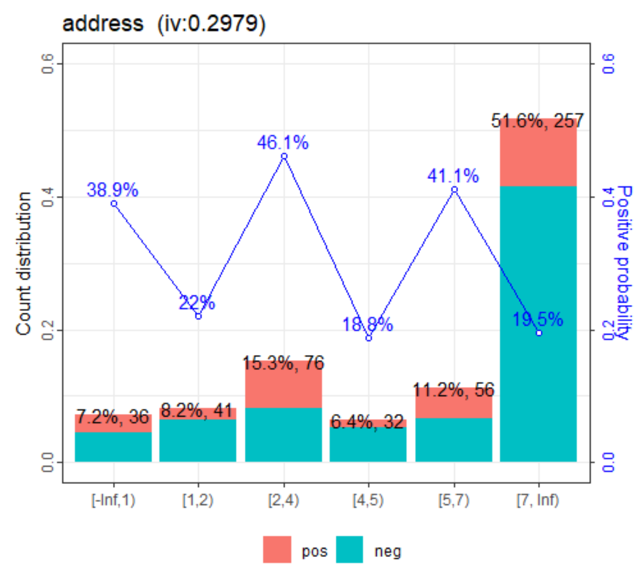
```
> train.data_removed
# A tibble: 498 × 7
     age    ed employ address income debtinc default
   <dbl> <dbl>  <dbl>   <dbl>  <dbl> <chr>     <dbl>
 1    41     3     17      12    176 9.3           1
 2    40     1     15      14     55 5.5           0
 3    41     1     15      14    120 2.9           0
 4    41     2      5       5     25 10.2          0
 5    36     1      0      13     25 19.7          0
 6    27     1      0       1     16 1.7           0
 7    25     1      4       0     23 5.2           0
 8    52     1     24      14     64 10            0
 9    37     1      6       9     29 16.3          0
10    36     2      9       6     49 8.6           1
# … with 488 more rows
# ℹ Use `print(n = ...)` to see more rows
```

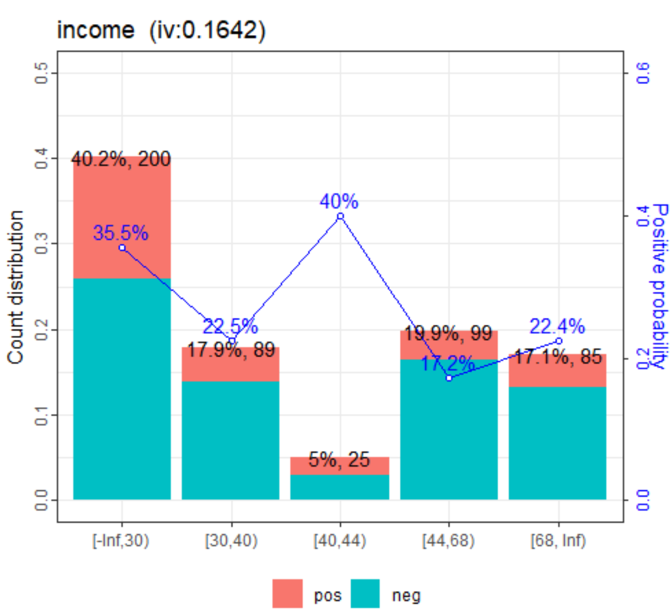
Bin variables according to Weight of Evidence (WoE) has the effect of dividing bins groups and statistics the number of observations, the number of default and non-default in each group.

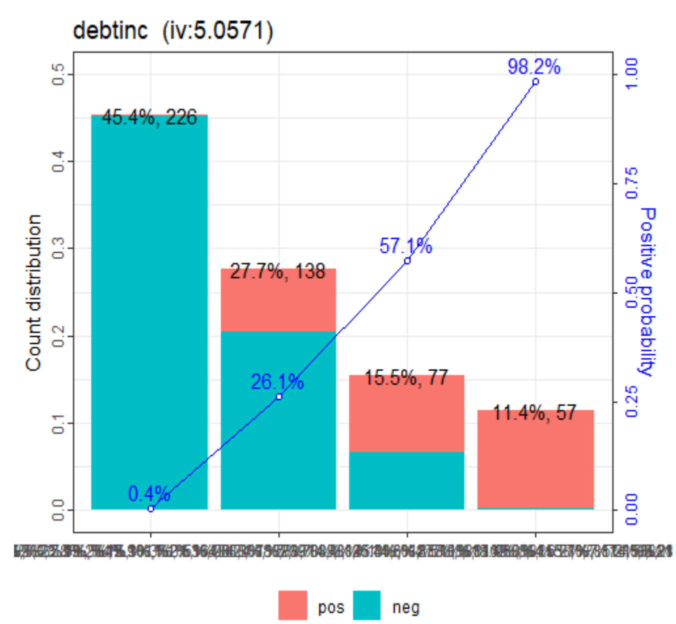

age (iv:0.2993)
Count distribution
Positive probability
12.4%, 62
31.3%, 156
39.2%, 195
8.6%, 43
8.4%, 42
48.4%
33.3%
19%
11.6%
31%
[-Inf,26)
[26,33)
[33,42)
[42,47)
[47, Inf)
pos
neg


ed (iv:0.0777)
Count distribution
Positive probability
54.2%, 270
28.7%, 143
17.1%, 85
22.6%
31.5%
36.5%
[-Inf,2)
[2,3)
[3, Inf)
pos
neg


employ (iv:0.6057)
Count distribution
Positive probability
28.7%, 143
12%, 60
41.4%, 206
17.9%, 89
48.9%
33.3%
18.9%
9%
[-Inf,4)
[4,6)
[6,15)
[15, Inf)
pos
neg


address (iv:0.2979)
Count distribution
Positive probability
7.2%, 36
8.2%, 41
15.3%, 76
6.4%, 32
11.2%, 56
51.6%, 257
38.9%
22%
46.1%
18.8%
41.1%
19.5%
[-Inf,1)
[1,2)
[2,4)
[4,5)
[5,7)
[7, Inf)
pos
neg


income (iv:0.1642)
Count distribution
Positive probability
40.2%, 200
17.9%, 89
5%, 25
19.9%, 99
17.1%, 85
35.5%
22.5%
40%
17.2%
22.4%
[-Inf,30)
[30,40)
[40,44)
[44,68)
[68, Inf)
pos
neg


debtinc (iv:5.0571)
Count distribution
Positive probability
45.4%, 226
27.7%, 138
15.5%, 77
11.4%, 57
0.4%
26.1%
57.1%
98.2%
pos
neg

Run the logit model for the training set for the initial data

```
Call:
glm(formula = default ~ employ + address + income, family = binomial(link = "logi
t"),
    data = train.data_removed)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.3853  -0.8339  -0.5719   1.1450   2.7569

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.11591    0.19512  -0.594  0.55248
employ      -0.15922    0.02589  -6.151 7.72e-10 ***
address     -0.03992    0.01870  -2.135  0.03276 *
income       0.01298    0.00428   3.032  0.00243 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 585.91  on 497  degrees of freedom
Residual deviance: 526.59  on 494  degrees of freedom
AIC: 534.59

Number of Fisher Scoring iterations: 4
```

Run logit model for training set for data binning according to WOE

```
Call:
glm(formula = default ~ ., family = binomial(link = "logit"),
    data = train.data_woe)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.02326  -0.31094  -0.05771   0.08218   2.81663

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.0114     0.1835  -5.511 3.57e-08 ***
age_woe       1.0600     0.3495   3.033  0.00242 **
ed_woe        2.0714     0.6352   3.261  0.00111 **
employ_woe    0.6998     0.2717   2.576  0.01001 *
address_woe   0.6560     0.3145   2.086  0.03696 *
income_woe   -0.1853     0.5170  -0.358  0.72010
debtinc_woe   1.0821     0.1419   7.627 2.40e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 585.91  on 497  degrees of freedom
Residual deviance: 226.56  on 491  degrees of freedom
AIC: 240.56

Number of Fisher Scoring iterations: 7
```

b) Test data.

```
> test.data
# A tibble: 202 × 9
     age    ed employ address income debtinc creddebt
   <dbl> <dbl>  <dbl>   <dbl>  <dbl> <chr>   <chr>
 1    27     1     10       6     31 17.3    1.362202
 2    24     2      2       0     28 17.3    1.787436
 3    39     1     20       9     67 30.6    3.833874
 4    43     1     12      11     38 3.6     0.128592
 5    24     1      3       4     19 24.4    1.358348
 6    48     1     22      15    100 9.1     3.7037
 7    39     1      6       9     61 5.7     0.563274
 8    39     1     22       3     52 3.2     1.154816
 9    47     1     17      21     43 5.6     0.587552
10    28     1      3       6     26 10      0.4316
# … with 192 more rows, and 2 more variables:
#   othdebt <chr>, default <dbl>
# ℹ Use `print(n = ...)` to see more rows, and `colnames()` to see
all variable names
```

c) Scored

**Employ:**

```
   variable        bin count count_distr neg pos    posprob        woe      bin_iv
1:   employ  [-Inf,4)   143   0.2871486  73  70 0.48951049  0.9269328 0.286174605
2:   employ     [4,6)    60   0.1204819  40  20 0.33333333  0.2757499 0.009701453
3:   employ    [6,15)   206   0.4136546 167  39 0.18932039 -0.4855351 0.086392405
4:   employ [15, Inf)    89   0.1787149  81   8 0.08988764 -1.3461106 0.223430895
    total_iv breaks is_special_values points
1: 0.6056994      4             FALSE     11
2: 0.6056994      6             FALSE      3
3: 0.6056994     15             FALSE     -6
4: 0.6056994    Inf             FALSE    -15
```

**Address:**

```
$address
   variable       bin count count_distr neg pos   posprob        woe      bin_iv
1:  address [-Inf,1)     36  0.07228916  22  14 0.3888889  0.5169119 0.021321558
2:  address     [1,2)    41  0.08232932  32   9 0.2195122 -0.2996143 0.006875917
3:  address     [2,4)    76  0.15261044  41  35 0.4605263  0.8106730 0.115035350
4:  address     [4,5)    32  0.06425703  26   6 0.1875000 -0.4974400 0.014041011
5:  address     [5,7)    56  0.11244980  33  23 0.4107143  0.6078837 0.046485151
6:  address [7, Inf)    257  0.51606426 207  50 0.1945525 -0.4517988 0.094174604
    total_iv breaks is_special_values points
1: 0.2979336      1             FALSE      1
2: 0.2979336      2             FALSE     -1
3: 0.2979336      4             FALSE      2
4: 0.2979336      5             FALSE     -1
5: 0.2979336      7             FALSE      2
6: 0.2979336    Inf             FALSE     -1
```

**Income:**

```
   variable        bin count count_distr neg pos   posprob        woe     bin_iv
1:   income [-Inf,30)   200   0.4016064 129  71 0.3550000  0.3717645 0.05981968
2:   income    [30,40)    89   0.1787149  69  20 0.2247191 -0.2694772 0.01216699
3:   income    [40,44)    25   0.0502008  15  10 0.4000000  0.5634319 0.01771512
4:   income    [44,68)    99   0.1987952  82  17 0.1717172 -0.6046089 0.06231052
5:   income [68, Inf)    85   0.1706827  66  19 0.2235294 -0.2763187 0.01219653
    total_iv breaks is_special_values points
1: 0.1642088     30             FALSE      0
2: 0.1642088     40             FALSE      0
3: 0.1642088     44             FALSE     -1
4: 0.1642088     68             FALSE      1
5: 0.1642088    Inf             FALSE      0
```

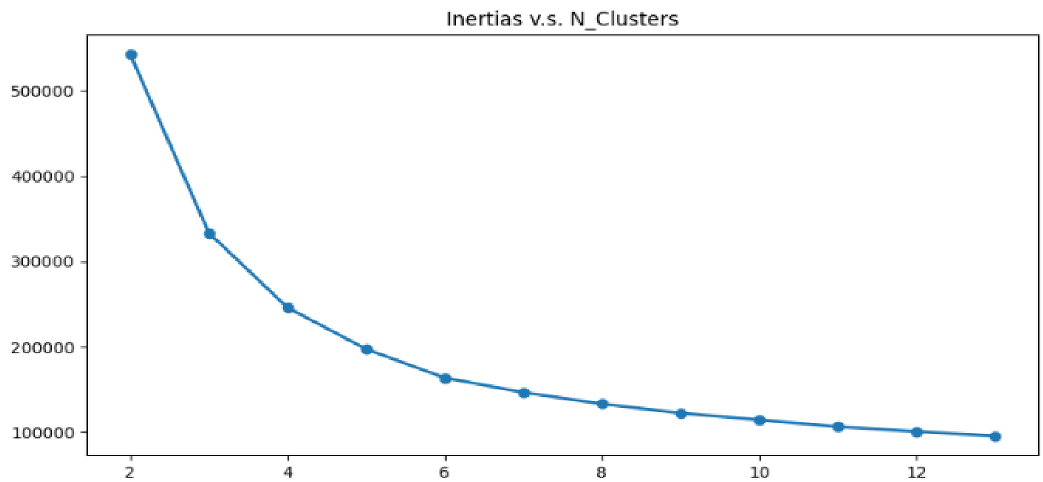
d) Scoredcard.

```
> z<-log(logit.pred.prob_woe/(1-logit.pred.prob_woe))
> head(z,10)
        1         2         3         4         5         6         7         8 
-1.278955  1.202838        NA -8.082592        NA -2.642901 -2.757276 -7.222463 
        9        10 
-7.368769 -5.084318 
```

## 4. K - Nearest Neighbors ( KNN model)

❖ Clustering.

Elbow method:




We must choose the value of k at the "elbow" in order to establish the ideal number of clusters. the point after which the distortion or inertia starts to linearly diminish. Hence, we determine that 4 is the optimal number of clusters for the data.

❖ Predicting Risk: Using the K-NN Classification Model.

```
X, y = df.drop("default", axis=1), df["default"]
X_train, X_test, y_train, y_test = \
model_selection.train_test_split(X, y, test_size=0.20,random_state=0)
```

```
max_score = 0
max_k = 0
for k in range(1, 100):
    neigh = KNeighborsClassifier(n_neighbors=k)
    neigh.fit(X_train,y_train)
    score = f1_score(y_test, neigh.predict(X_test),average='micro')
    if score > max_score:
        max_k = k
        max_score = score
```

If we use K-Nearest Neighbors Classification, then the value of K is 32 to get the best prediction, then the average accuracy is 75%.

## III- DATA

### 1. Exploration of data characteristics.

a) Variable Descriptions.

The table below lists the statistical descriptions of the variables. Each variable has 1150 observations, bank loan.

**Table 1.** List of variables

| Variable Name | Description | Mesurement |
|---|---|---|
| *age* | Age of the Customers | Years old |
| *ed* | Education Level | |
| *employ* | Work Experience | Year |
| *address* | Address of the Customer | |
| *income* | Yearly Income of the customer | $ |
| *debtinc* | Debt Income Ratio | % |
| *creddebt* | Credit to Debt ratio | % |
| *othdebt* | Other debts | |
| default | Customer defaulted in the past<br>=1 if defaulted<br>= 0 if non defaulted | |

A summary of statistics along with the correlation matrix are presented respectively in Table 2 and 3 below.

b) Description statistics

Sau khi xóa các dòng ko có giá trị thì chúng ta có 700 quan sát

**Table 2.** Descriptive statistics

| Variable | Mean | SD | Min | Max | $1^{st}\ Qu$ | Median | $3^{rd}\ Qu$ |
|---|---|---|---|---|---|---|---|
| **age** | 34.9 | 8 | 20 | 56 | 29 | 34 | 40 |
| **ed** | 1.72 | 0.93 | 1 | 5 | 1 | 1 | 2 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **employ** | 8.4 | 6.66 | 0 | 31 | 3 | 7 | 12 |
| **address** | 8.485 | 6.978 | 0 | 34 | 3 | 7 | 12 |
| **income** | 45.6 | 36.8 | 14 | 446 | 24 | 34 | 55 |
| **debinc** | 10.26 | 6.83 | 0.4 | 41.3 | 5 | 8.6 | 14.125 |
| **creddebt** | 1.55 | 2.12 | 0.012 | 20.6 | 0.37 | 0.85 | 1.9 |
| **othdebt** | 3.1 | 3.29 | 0.05 | 27 | 1.05 | 1.99 | 3.92 |
| **default** | 0.26 | 0.44 | 0 | 1 | 0 | 0 | 1 |

The mean of "default" is the ratio of bad debt contracts and accounts for 26 %.

c) Check if there is any multicollinearity between variables.

**Table 3.** Correlation Matrix

| | ***Age*** | ***ed*** | ***employ*** | ***address*** | ***income*** | ***debtinc*** | ***creddebt*** | ***othdebt*** | ***default*** |
|---|---|---|---|---|---|---|---|---|---|
| ***age*** | 1 | | | | | | | | |
| ***ed*** | 0.022 | 1 | | | | | | | |
| ***employ*** | 0.54 | -0.15 | 1 | | | | | | |
| ***address*** | 0.6 | 0.057 | 0.32 | 1 | | | | | |
| ***income*** | 0.48 | 0.24 | 0.62 | 0.32 | 1 | | | | |
| ***debtinc*** | 0.016 | 0.0088 | -0.031 | 0.011 | -0.027 | 1 | | | |
| ***creddebt*** | 0.3 | 0.088 | 0.4 | 0.21 | 0.57 | 0.5 | 1 | | |
| ***othdebt*** | 0.34 | 0.17 | 0.41 | 0.23 | 0.61 | 0.58 | 0.63 | 1 | |
| ***default*** | -0.14 | 0.11 | -0.28 | -0.16 | -0.071 | 0.39 | 0.24 | 0.15 | 1 |

## 2. Data visualization

It can be seen that the bank has more than 500 potential customers; The remaining nearly 200 borrowershave been defaulted in the past and are often no longer viable customers. they will often not be approved for extending a loan by the bank because they are not able to pay back the amount they are borrowing.

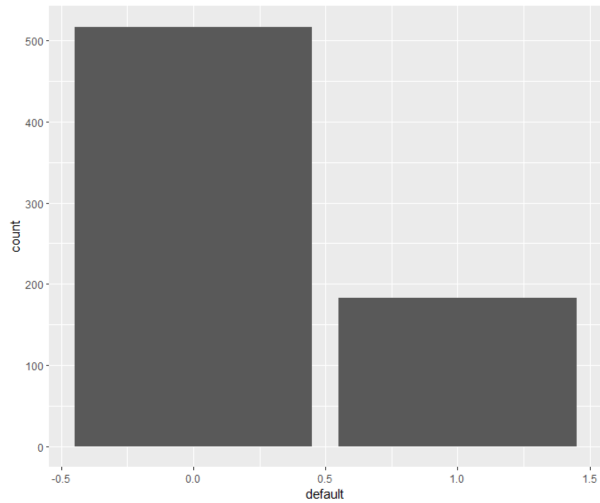
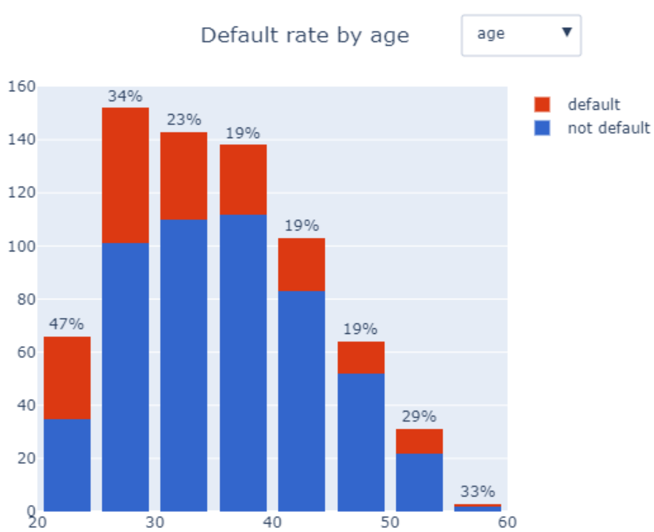



**Figure 1. Customer defaulted in the past**

**Figure 2. default rate by age**

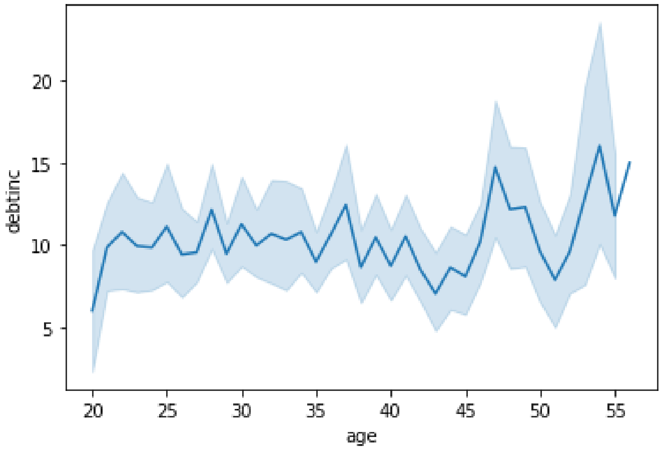


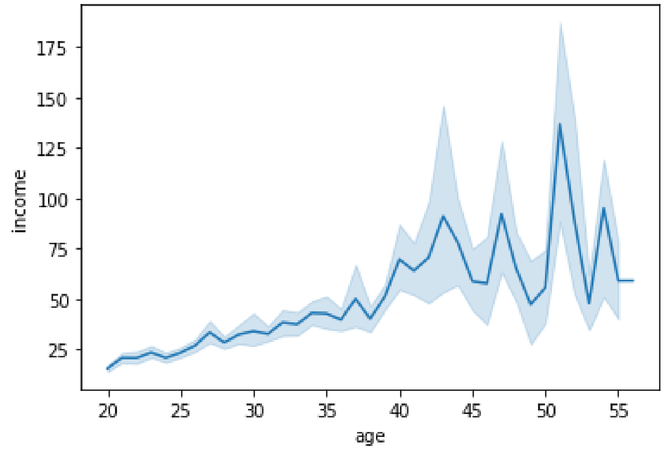

Through the image below, we can see that the customers owed their wives very little in the past, especially the older they are, the smaller the debt ratio.

Most of the customers are in the age group of 30 to 50, the repayment ability of customers is quite high, there are some customers in this age group who have defaulted in the past and these subjects are quite difficult to extend their debt in bank.

**Figure 3. Age against Debt-income-ratio**

**Figure 4. Age against Income**




The lower the debt-to-income ratio, the more attractive a borrower becomes. Moving lines mostly move between 5%-15% which can be considered quite attractive to lenders.
Lenders normally want a rate of no more than 36% since a borrower's DTI of 43% is often the most rate they can obtain while still being eligible for a mortgage.

The most noticeable difference between the income levels of costomers of different ages is that the older they are, the more job experience they have, which results in a very high average yearly income.
This shows that the income of customers who can afford to pay their debts, so their potential is great. This makes the borrower more attractive to the bank.

Figure 5. Education with debt-income-ratio

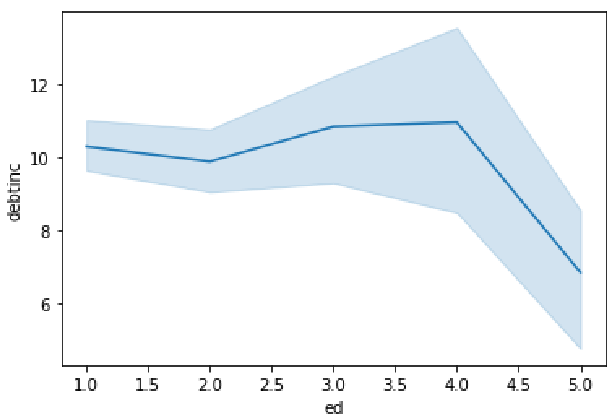
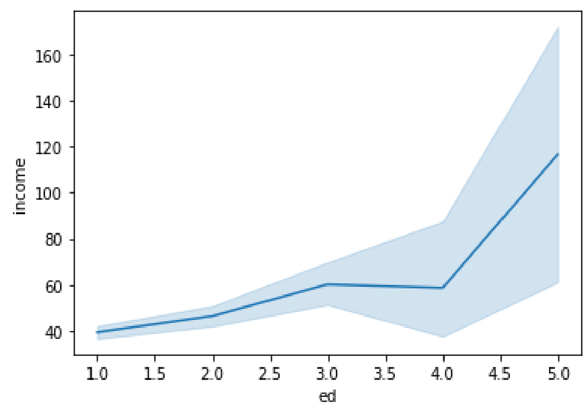



Figure 6. Education against Income




The higher the education, the higher the concurrent income. This is part of making loan extension easier.

**Figure 7. Linear Regression Model plot**

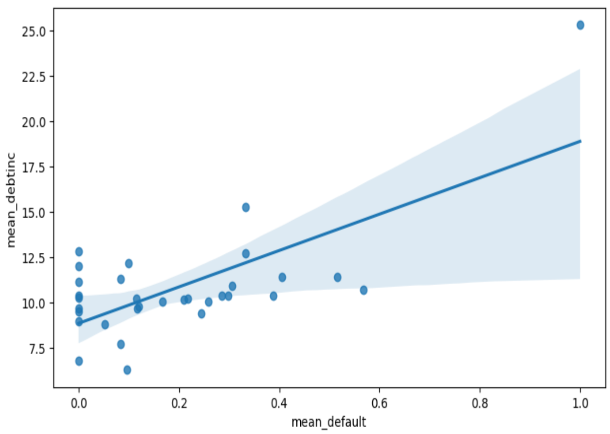



**Figure 7.1. Debt-income-ratio against default**

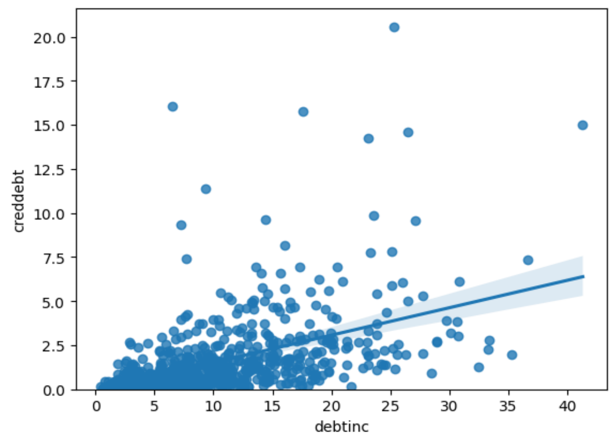



**Figure 7.2. Credit-to-debt-ratio with debt-income-ratio**

## 3. Data analysis

a) Missing value

**Table 1: Check for missing value**

| | Total | Percent |
|---|---|---|
| **default** | 450 | 39.130435 |
| **age** | 0 | 0.000000 |
| **Ed** | 0 | 0.000000 |
| **employ** | 0 | 0.000000 |
| **address** | 0 | 0.000000 |
| **income** | 0 | 0.000000 |
| **debtinc** | 0 | 0.000000 |
| **creddebt** | 0 | 0.000000 |
| **Othdebt** | 0 | 0.000000 |

We can see that the "default" observation is missing 450 observations so we need to delete the rows that are missing the "default" value.

b) Outliers

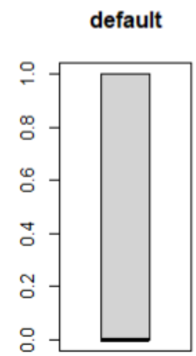
For determining if the highest or lowest value in a dataset is an outlier, we utilize the Grubbs test.

The null and alternative hypotheses are as follows since the Grubbs test discovers one outlier at a time (highest or lowest value):

$$\begin{cases} H_0: The\ highest\ value\ is\ not\ an\ outlier \\ H_1: The\ highest\ vaue\ is\ an\ outlier \end{cases}$$

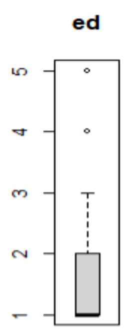
❖ Grubbs test for default:

```
        Grubbs test for one outlier

data:  data.rm$default
G = 1.67961, U = 0.99596, p-value = 1
alternative hypothesis: highest value 1 is an outlier
```




Since p-value is 1. At the 5% significance level, we do not reject the hypothesis that the highest value 1 is not an outlier.

❖ Grubbs test for ed:

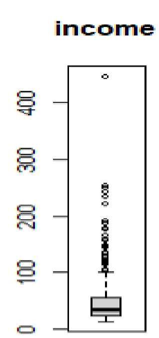



```
        Grubbs test for one outlier

data:  data.rm$ed
G = 0.77877, U = 0.99913, p-value = 1
alternative hypothesis: lowest value 1 i
s an outlier
```

We do not reject the hypothesis that the lowest value 1 is not an outlier. Because at 5% significant level , p-value is 1 bigger than 5%.

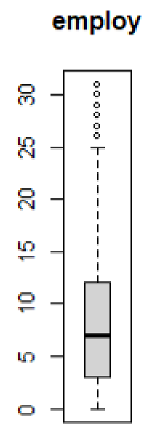
❖ Grubbs test for income

```
        Grubbs test for one outlier

data:  data.rm$income
G = 0.85840, U = 0.99894, p-value = 1
alternative hypothesis: lowest value 14 is an o
utlier
```




Since p-value is 1. At the 5% significance level, we do not reject the hypothesis that the lowest value 14 is not an outlier.

❖ Grubbs test for employ




```
        Grubbs test for one outlier

data:  data.rm$employ
G = 1.25992, U = 0.99773, p-value = 1
alternative hypothesis: lowest value 0 is an outlier
```

At the 5% significance level, we do not reject the hypothesis that the lowest value 0 is not an outlier. Because p-value is 1 bigger than 5%.

## IV- FINDINGS.

### 1. Confution matrix

❖ Model on **"train"** set for logit model with initial data.

| | Reference | | |
|---|---|---|---|
| **Prediction** | | **0** | **1** |
| | **0** | 345 | 120 |
| | **1** | 16 | 17 |

❖ Model on **"test"** set for logit model with initial data.

| | Reference | | |
|---|---|---|---|
| **Prediction** | | **0** | **1** |
| | **0** | 150 | 41 |
| | **1** | 6 | 5 |

Thus, the model test set correctly predicts 150 Non-defaulted cases and 5 Defaulted cases. Forecast error 41 cases Defaulted to Non-defaulted; correct forecast 6 cases Non-defaulted to Defaulted. The prediction accuracy rate is 76.73%.

❖ Model on test.data set for logit model with data binning according to WOE.

| | Reference |
|---|---|

| | | | |
|---|---|---|---|
| | | **0** | **1** |
| **Prediction** | **0** | 337 | 31 |
| | **1** | 24 | 106 |

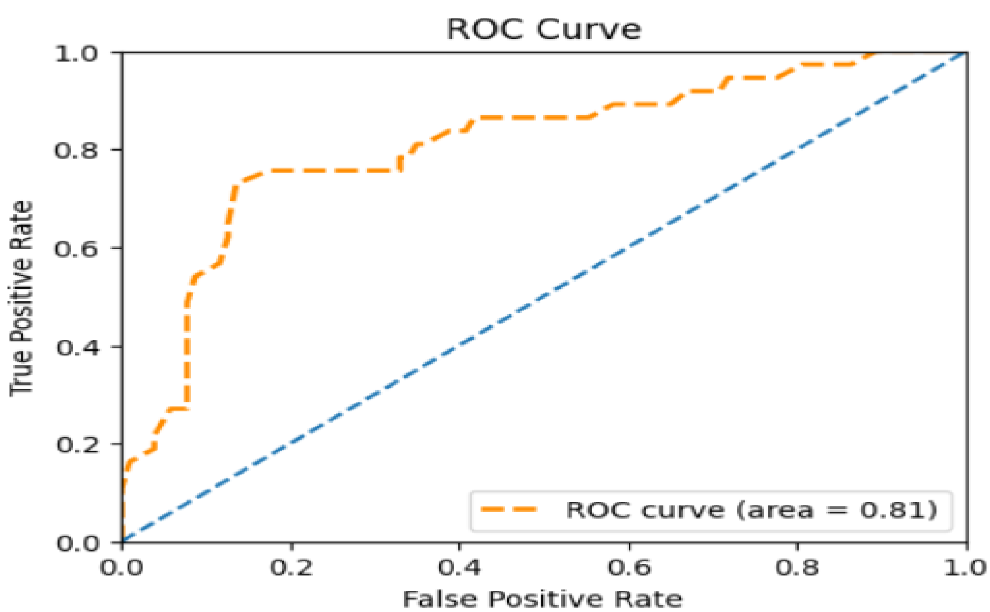
❖ Model on test.data set for logit model with data binning according to WOE.

| | **Reference** | | |
|---|---|---|---|
| | | **0** | **1** |
| **Prediction** | **0** | 114 | 17 |
| | **1** | 30 | 16 |

The model test set correctly predicts 114 Non-defaulted cases and 16 Defaulted cases. Forecast error 17 cases Defaulted to Non-defaulted; correct forecast 30 cases Non-defaulted to Defaulted. The prediction accuracy rate is 73.45 %.

**2. AUROC.**

❖ ROC curve on test set




The AUC (area under curve) index measures the area under the ROC curve, indicating whether the classification ability of the Default/ Non-default contracts of the logistic regression model is strong or weak.

AUC ∈ [0,1], the larger its value, the better the model. For this logistic regression model, AUC = 0.81 is quite high, showing that the model's predictive ability is good and the model can be applied in practice.

### 3. GINI.

Gini impurity is a measure of misclassification, which applies in a multiclass classifier context.

$$GINI = 2 * AUROC - 1$$
$$Gini = 2 * 0.81 - 1 = 0.62$$

## V- CONCLUSION

The dataset contains a variety of factors that might impact a loan applicant's credit score and credit eligibility, such as education level, age, income, and loan features. File data is particularly helpful for creating predictive models in credit analysis since it includes the aspects that are frequently taken into account when making loan decisions.

A potential limitation of the dataset is that it is relatively small, with a total of just over 1,000 observations, and after removing blanks, only 700 observations remain. However, the data is well structured and includes many different variables, making it a good choice for regression analysis and machine learning.

In conclusion, the available credit risk analysis datasets are a valuable resource for anyone interested in analyzing credit risk and building predictive models. It contains a wealth of information about loan applicants, which can be used to gauge their creditworthiness and predict the likelihood of default on a loan. The information may be utilized to create classification, analysis, which can assist financial organizations in lowering risk while extending credit to their clients.

## VI- APPENDIX.

### 1. DATA

```
In [2]: df = pd.read_excel('bankloans.xlsx')
        df
```

Out[2]:

| | age | ed | employ | address | income | debtinc | creddebt | othdebt | default |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 41 | 3 | 17 | 12 | 176 | 9.3 | 11.359392 | 5.008608 | 1.0 |
| **1** | 27 | 1 | 10 | 6 | 31 | 17.3 | 1.362202 | 4.000798 | 0.0 |
| **2** | 40 | 1 | 15 | 14 | 55 | 5.5 | 0.856075 | 2.168925 | 0.0 |
| **3** | 41 | 1 | 15 | 14 | 120 | 2.9 | 2.658720 | 0.821280 | 0.0 |
| **4** | 24 | 2 | 2 | 0 | 28 | 17.3 | 1.787436 | 3.056564 | 1.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **1145** | 34 | 1 | 12 | 15 | 32 | 2.7 | 0.239328 | 0.624672 | NaN |
| **1146** | 32 | 2 | 12 | 11 | 116 | 5.7 | 4.026708 | 2.585292 | NaN |
| **1147** | 48 | 1 | 13 | 11 | 38 | 10.8 | 0.722304 | 3.381696 | NaN |
| **1148** | 35 | 2 | 1 | 11 | 24 | 7.8 | 0.417456 | 1.454544 | NaN |
| **1149** | 37 | 1 | 20 | 13 | 41 | 12.9 | 0.899130 | 4.389870 | NaN |

1150 rows × 9 columns

### 2. MISSING VALUE

```
In [118]: def missing_value(val):
              total = val.isnull().sum().sort_values(ascending = False)

              percent = (val.isnull().sum()/val.isnull().count()*100).sort_values(
                  ascending = False)

              return pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
```

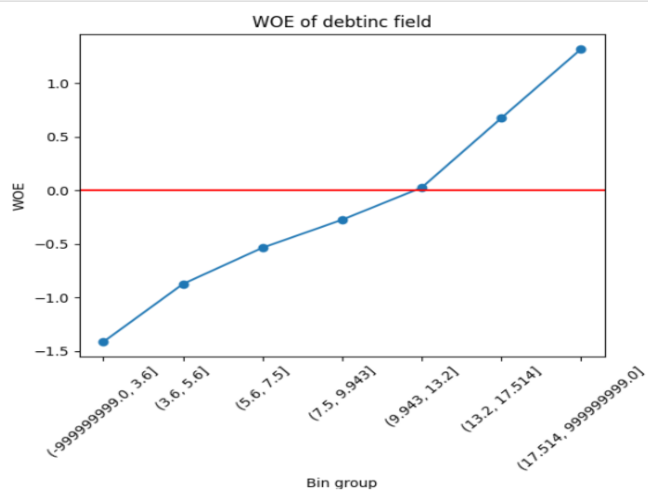
```
In [119]: missing_value(data)
```

Out[119]:

| | Total | Percent |
|---|---|---|
| default | 450 | 39.130435 |
| age | 0 | 0.000000 |
| ed | 0 | 0.000000 |
| employ | 0 | 0.000000 |
| address | 0 | 0.000000 |
| income | 0 | 0.000000 |
| debtinc | 0 | 0.000000 |
| creddebt | 0 | 0.000000 |
| othdebt | 0 | 0.000000 |

## 3. WOE OF debtinc

```
In [59]: def _plot(df_summary):
             colname = list(df_summary['COLUMN'].unique())[0]
             df_summary['WOE'].plot(linestyle='-', marker='o')
             plt.title('WOE of {} field'.format(colname))
             plt.axhline(y=0, color = 'red')
             plt.xticks(rotation=45)
             plt.ylabel('WOE')
             plt.xlabel('Bin group')

         _plot(df_summary)
```




## 4. Tranfer data to WOE

```
> train.data_woe <- woebin_ply(train.data_removed, bins)
i Converting into woe values ...
✓ Woe transformating on 498 rows and 6 columns in 00:00:02
> head(train.data_woe)
   default    age_woe     ed_woe employ_woe address_woe income_woe debtinc_woe
1:       1 -0.4827801  0.4139002 -1.3461106  -0.4517988 -0.2763187 -0.07255684
2:       0 -0.4827801 -0.2625634 -1.3461106  -0.4517988 -0.6046089 -4.44720337
3:       0 -0.4827801 -0.2625634 -1.3461106  -0.4517988 -0.2763187 -4.44720337
4:       0 -0.4827801  0.1905920  0.2757499   0.6078837  0.3717645 -4.44720337
5:       0 -0.4827801 -0.2625634  0.9269328  -0.4517988  0.3717645 -4.44720337
6:       0  0.2757499 -0.2625634  0.9269328  -0.2996143  0.3717645 -4.44720337
```

## 5. Confusion Matrix and Statistic of test set for initial data

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 150  41
         1   6   5

               Accuracy : 0.7673
                 95% CI : (0.7029, 0.8238)
    No Information Rate : 0.7723
    P-Value [Acc > NIR] : 0.6048

                  Kappa : 0.096

 Mcnemar's Test P-Value : 7.071e-07

            Sensitivity : 0.10870
            Specificity : 0.96154
         Pos Pred Value : 0.45455
         Neg Pred Value : 0.78534
             Prevalence : 0.22772
         Detection Rate : 0.02475
   Detection Prevalence : 0.05446
      Balanced Accuracy : 0.53512

       'Positive' Class : 1
```

## 6. Confusion matrix and Statistic of test set for data binning according to WOE.

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 114  17
         1  30  16

               Accuracy : 0.7345
                 95% CI : (0.663, 0.7979)
    No Information Rate : 0.8136
    P-Value [Acc > NIR] : 0.99637

                  Kappa : 0.2401

 Mcnemar's Test P-Value : 0.08005

            Sensitivity : 0.4848
            Specificity : 0.7917
         Pos Pred Value : 0.3478
         Neg Pred Value : 0.8702
             Prevalence : 0.1864
         Detection Rate : 0.0904
   Detection Prevalence : 0.2599
      Balanced Accuracy : 0.6383

       'Positive' Class : 1
```