CHAPTER 4
ANALYSIS AND DESIGN

In this research project, to predict which creditors are accepted and rejected, Logistic Regression and Extreme Gradient Boosting algorithms will be implemented. The dataset used is taken from Kaggle which contains 13 variables including loan id, gender, marital status, dependents, education, self-employed, applicant income, applicant income, total loan, loan term, credit history, property area, loan status.

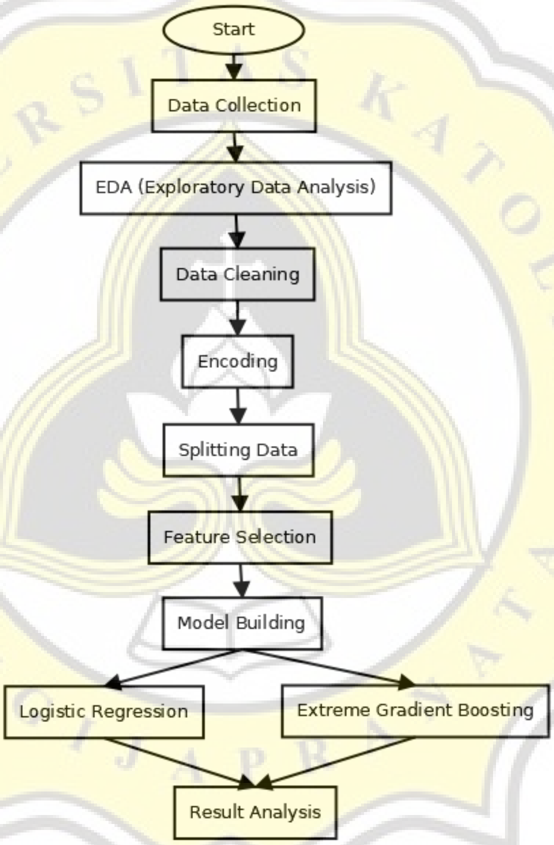

Figure 4.1 Workflow

4.1. Data Collection

The first step is to import NumPy and pandas which are used to read the dataset CSV file and perform calculations. Then data collection is carried out, namely reading the dataset, displaying the dataset information, the number of columns and rows from the dataset, and eliminating unnecessary columns such as Loan_Id.

Table 4.1.1 Dataset

Variable	Description	Data Type
Loan_ID	Unique Loan ID	Integer
Gender	Male/Female	Character
Married	Applicant marital status (Y/N)	Character
Dependent	Applicant number of dependent	Integer
Education	Applicant education (Graduate/Not Graduate)	String
Self_Employed	Self employed (Y/N)	Character
ApplicantIncome	Applicant income	Integer
CoapplicantIncome	Coapplicant income	Integer
LoanAmount	Loan amount in thousand	Integer
Loan_Amount_Term	Terms of loan in months	Integer
Credit_History	Credit history meets guidelines	Integer
Property_Area	Urban / Semi Urban / Rural	String
Loan_Status	Loan approved (Y/N)	String

Table 4.1.2 Dataset Information

#	Column	Non-Null Count	Dtype
0	Loan_ID	614 non-null	object
1	Gender	601 non-null	object
2	Married	611 non-null	object
3	Dependents	599 non-null	object
4	Education	614 non-null	object
5	Self_Employed	582 non-null	object
6	ApplicantIncome	614 non-null	float64
7	CoapplicantIncome	614 non-null	float64
8	LoanAmount	592 non-null	float64
9	Loan_Amount_Term	600 non-null	float64
10	Credit_History	564 non-null	float64
11	Property_Area	614 non-null	object
12	Loan_Status	614 non-null	object

4.2. EDA (Exploratory Data Analysis)

Then EDA (Exploratory Data Analysis) is carried out to find out and understand the contents of the dataset, from this step it can be seen which variables affect the results of the loan application decision (accepted or rejected).

Table 4.2.1 EDA

Column	EDA Result
Loan_Status	Y 422 N 192 Name: Loan_Status, dtype: int64
Gender	Male 489 Female 112 Name: Gender, dtype: int64 Male and Loan Status accepted: 339 Male and Loan Status not accepted: 150 Female and Loan Status accepted: 75 Female and Loan Status not accepted: 37
Married	Yes 398 No 213 Name: Married, dtype: int64 Married and Loan Status accepted: 285 Married and Loan Status not accepted: 113 Not Married and Loan Status accepted: 134 Not Married and Loan Status not accepted: 79
Dependents	0 345 1 102 2 101 3+ 51 Name: Dependents, dtype: int64 Dependents 0 and Loan Status accepted: 238 Dependents 0 and Loan Status not accepted: 107 Dependents 1 and Loan Status accepted: 66 Dependents 1 and Loan Status not accepted: 36 Dependents 2 and Loan Status accepted: 76 Dependents 2 and Loan Status not accepted: 25 Dependents 3 and Loan Status accepted: 33 Dependents 3 and Loan Status not accepted: 18

Column	EDA Result
Self_Employed	No 500 Yes 82 Name: Self_Employed, dtype: int64 Self Employed and Loan Status accepted: 56 Self Employed and Loan Status not accepted: 26 Not Self Employed and Loan Status accepted: 343 Not Self Employed and Loan Status not accepted: 157
ApplicantIncome	Minimum Applicant Income: 150 Maximum Applicant Income: 81000 Mean Applicant Income: 5403.459283387622 Accepted Applicant Income: 2500 8 3333 5 6250 4 2583 4 6000 4 .. 1863 1 3400 1 3900 1 1926 1 7787 1 Name: ApplicantIncome, Length: 364, dtype: int64 Declined Applicant Income: 4583 4 2600 3 10000 3 4166 3 5000 3 .. 3708 1 2917 1 1800 1 7333 1 6400 1 Name: ApplicantIncome, Length: 172, dtype: int64

Column	EDA Result
LoanAmount	Minimum Loan Amount: 9.0 Maximum Loan Amount: 700.0 Accepted Loan Amount: 120.0 17 110.0 12 100.0 11 130.0 10 187.0 9 .. 236.0 1 380.0 1 296.0 1 156.0 1 59.0 1 Name: LoanAmount, Length: 161, dtype: int64 Declined Loan Amount: 110.0 5 160.0 4 113.0 4 80.0 4 100.0 4 .. 308.0 1 124.0 1 570.0 1 111.0 1 214.0 1 Name: LoanAmount, Length: 119, dtype: int64

UNIVERSITAS KATOLIK SOEGIJAPRANATA

Column	EDA Result
Credit_History	1.0 475 0.0 89 Name: Credit_History, dtype: int64 Credit History 1 and Loan Status accepted: 378 Credit History 1 and Loan Status not accepted: 97 Credit History 0 and Loan Status accepted: 7 Credit History 0 and Loan Status not accepted: 82
Property_Area	Semiurban 233 Urban 202 Rural 179 Name: Property_Area, dtype: int64 Urban and Loan Status accepted: 133 Urban and Loan Status not accepted: 69 SemiUrban and Loan Status accepted: 179 SemiUrban and Loan Status not accepted: 54 Rural and Loan Status accepted: 110 Rural and Loan Status not accepted: 69

4.3. Data Cleaning

Then data cleaning will be carried out, this step is done so that the accuracy of the decision is better. Here it will be checked and filled in for the missing value. For categorical data such as Gender, Married, Dependents, Loan_Amount_Term, Credit_History, Credit_History, Self_Employed will be filled with the mode of each variable. And for numerical data such as LoanAmount, it will be filled with the mean of the variable.

Table 4.3.1 Sum of Missing Value

Column	Total missing value before	Total missing value after
Gender	13	0
Married	3	0
Dependents	15	0
Education	0	0
Self_Employed	32	0
ApplicantIncome	0	0
CoapplicantIncome	0	0
LoanAmount	22	0
Loan_Amount_Term	14	0
Credit_History	50	0
Property_Area	0	0
Loan_Status	0	0

Table 4.3.2 Before Data Cleaning

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
...	...	...	...	...	...	...	...	...	...	...	...	...
613	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0	Semiurban	N

Table 4.3.3 After Data Cleaning

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	Male	No	0	Graduate	No	5849	0.0	146.412162	360.0	1.0	Urban	Y
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
...	...	...	...	...	...	...	...	...	...	...	...	...
613	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0	Semiurban	N

4.4. Encoding

After that, so that the database can be read by the machine, encoding will be carried out. This step will change the Property_Area column from Rural / Semi Urban / Urban to 0 / 1 / 2, create new columns such as gender to gender_male and gender_female.

Table 4.4.1 Encoding

Column	Before encoding	After encoding
Gender	Male / Female	new column : Gender_Female, Gender_Male
Married	Graduate / Not Graduate	new column : Married_Yes Married_No
Dependents	0 / 1 / 2 / 3+	0 / 1 / 2 / 3
Education	Graduate / Not graduate	1 / 0
Self_Employed	Yes / No	new column : Self_Employed_Yes Self_Employed_No
Property_Area	Rural / Semiurban / Urban	0 / 1 / 2
Loan_Status	Y / N	1 / 0

4.5. Features Selection

In this step we will look for which variable have an effect on loan status using the correlation function.

Table 4.5.1 Features Selection

Variable	Values
Loan_Status	1.000000
CoapplicantIncome	0.540556
LoanAmount	0.036416
LoanAmount_Term	0.022549
ApplicantIncome	0.004710

4.6. Splitting Dataset

Then we will split the data into a data train to create a machine learning model and test data to test the performance of the model, here we will divide the data into 70% for the data train and 30% for the data test which divided by three so it will be three trials for testing, and we will divided it too into 60% for the data train and 40% for the data test which divided by three so it will be three trials for testing.

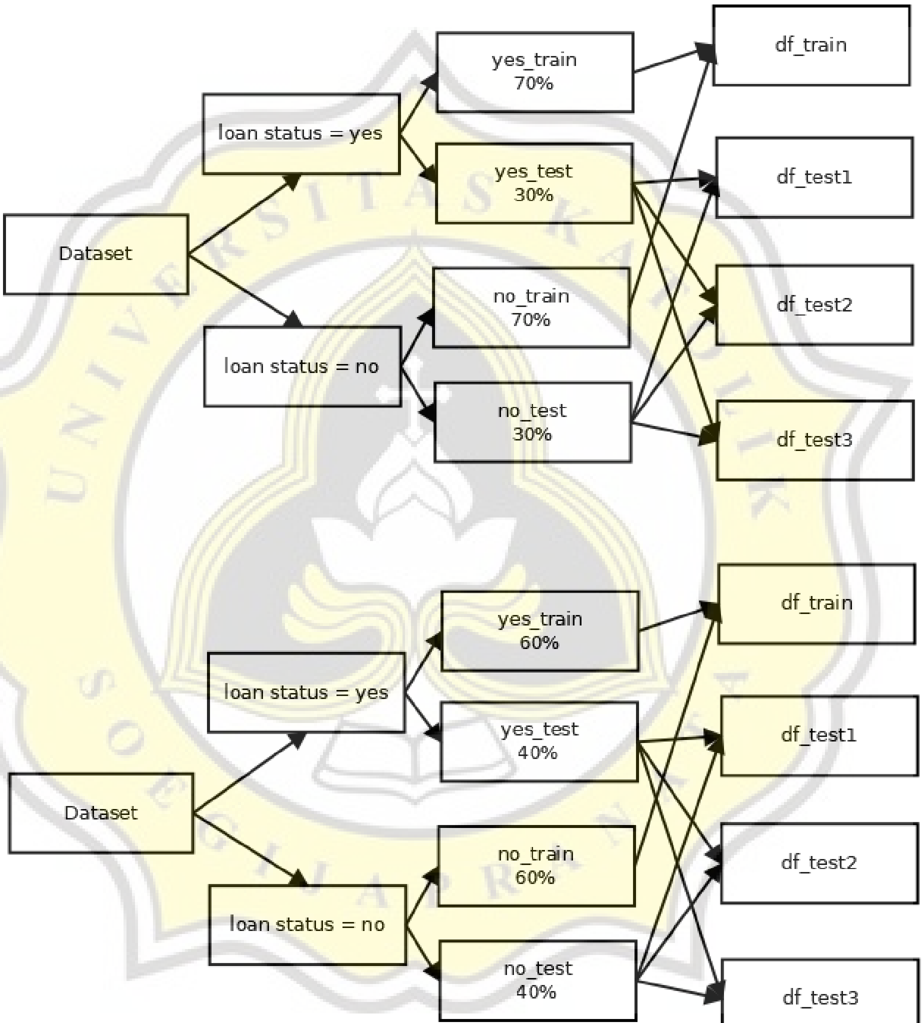

Figure 4.6.1 Splitting Dataset

4.7. Building model using Logistic Regression

After that, we will build a model using logistic regression. For logistic regression, the first author determines several parameters such as learning rate and number of iterations, and initializes weight and bias to 0.

Table 4.7.1 Initializes Weight and Bias

Features	**Weight**	**Bias**
Dependents	0	0
Education	0	0
ApplicantIncome	0	0
CoapplicantIncome	0	0
LoanAmount	0	0
Loan_Amount_Term	0	0
Credit_History	0	0
Property_Area	0	0
Loan_Status	0	0
Gender_Female	0	0
Gender_Male	0	0
Married_No	0	0
Married_Yes	0	0
Self_Employed_No	0	0
Self_Employed_Yes	0	0

Then the training process for some iteration parameters is carried out using this linear regression function :

$$z = w \,.\, x + \; b$$

Function 4.7.1 Linear Regression Function

z = Linear regression

w = weights

x = input data

b = bias

For the example of calculation author use 1 row data of dataset, shown as below

$$z = w \,.\, x + \; b$$

= [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] . [1 1 1 8080 2250 180 360 1 2 0 1 0 1 1 0] + 0

= 0

And using sigmoid function as shown below:

$$\hat{y} = \frac{1}{1 + e^{-(z)}}$$

Function 4.7.2 Sigmoid Function

$\hat{y}$ = Hypothesis / prediction

Z = Linear regression

For the example of calculation author use 1 row data of dataset, shown as below

$$\hat{y} = \frac{1}{1 + e^{-(z)}}$$

$$= \frac{1}{1 + e^{-(0)}}$$

= 0.5

Then we will calculate the gradient function to find the optimal values of the parameter, like new weight and new bias using this function:

$$dw = \left(\frac{1}{m}\right) * \ (\hat{y} - y) \,.\, x$$

Function 4.7.3 The Partial Derivative of Loss Function with Respect to Weight Function

$$db = \left(\frac{1}{m}\right) * \ (\hat{y} - y)$$

Function 4.7.4 The Partial Derivative of Loss Function with Respect to Bias Function

$$w := \ w - lr * dw$$

Function 4.7.5New Weight Function

$$b := \ b - lr * db$$

Function 4.7.6New Bias Function

w: = new weights

b: = new bias

lr = learning rate

w = weight

b = bias

dw = The partial derivative of loss function with respect to weight

db = The partial derivative of loss function with respect to bias

m = number of training data

$\hat{y}$ = Hypothesis / prediction

y = True value

X = Input data

For the example of calculation of dw author use 1 row data of dataset, shown as below

$$dw = \left(\frac{1}{m}\right) * \ (\hat{y} - y) \,.\, x$$

$$= \left(\frac{1}{1}\right) * \ (0.5 - 1) \,.\ [1\ 1\ 1\ 8080\ 2250\ 180\ 360\ 1\ \ 2\ 0\ 1\ 0\ 1\ 1\ 0]$$

= -0.5 . [1 1 1 8080 2250 180 360 1 2 0 1 0 1 1 0]

=$[-0.5\ \ -0.5\ \ -0.5\ \ -4040\ \ -1125\ \ -90\ \ -180\ \ -0.5\ \ -1\ 0\ \ -0.5\ 0\ \ -0.5\ \ -0.5\ \ 0]$

For the example of calculation of db author use 1 row data of dataset, shown as below

$$db = \left(\frac{1}{m}\right) * (\hat{y} - y)$$

$$= \left(\frac{1}{1}\right) * (0.5 - 1)$$

$$= -0.5$$

For the example of calculation of w: author use 1 row data of dataset, shown as below

$$w := w - lr * dw$$

$$= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0] - 0.0000001 * [-0.5\ \ -0.5\ \ -0.5\ \ -4040\ \ -1125\ \ -90\ \ -180\ \ -0.5\ \ -1\ \ 0\ \ -0.5\ \ 0 - 0.5\ \ -0.5\ 0]$$

$$= [-0.00000005\ \ -0.00000005\ \ -0.00000005\ \ -0.000404\ \ -0.0001125\ \ -0.000009\ \ \ -0.000018\ \ -0.00000005\ \ -0.0000001\ 0\ \ -0.00000005\ \ 0\ \ -0.00000005\ \ -0.00000005\ \ 0]$$

For the example of calculation of b: author use 1 row data of dataset, shown as below

$$b := b - lr * db$$

$$b := 0 - 0.0000001 * -0.5$$

$$= 0.00000005$$

After finding the new weight and new bias we will calculate the new linear regression function using that new weight and new bias, and we calculate the new sigmoid function. Then we calculate prediction using the function as show below

y=1 when $\hat{y} \geq 0.5$

y=0 when $\hat{y} < 0.5$

Function 4.7.7 Predict Function

y = true value

$\hat{y}$ = prediction value

For the example of calculation author use 1 row data of dataset, and the result as shown in **Function 4.6.2** is y = 0.5 so $\hat{y}$ =1

Then author will calculate the loss function using this function:

$$j(w,b) = \frac{1}{m}\sum_{i=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right)$$

$$= -\frac{1}{m}\sum_{i=1}^{m} \left[\left(\hat{y}^{(i)} \, log\, log\, \left(\hat{y}^{(i)}\right) + \left(1 - \hat{y}^{(i)}\right) log\, log\, \left(1 - \hat{y}^{(i)}\right)\right)\right]$$

Function 4.7.8 Loss Function

j(w,b) = loss of the training set

L = loss of the training example

i = data ke

m = number of training data

$\hat{y}$ = Hypothesis / prediction

y = True value

For the example of calculation author use 1 row data of dataset, shown as below

$$j(w,b) = -\frac{1}{m}\sum_{i=1}^{m} \left[\left(\hat{y}^{(i)} \, log\, log\, \left(\hat{y}^{(i)}\right) + \left(1 - \hat{y}^{(i)}\right) log\, log\, \left(1 - \hat{y}^{(i)}\right)\right)\right]$$

$$= -\frac{1}{1}[(0.5(0.5) + (1 - 0.5)\, log\, log\, (1 - 0.5))]$$

$$= -1[(0.5(0.5) + (1 - 0.5)\, log\, log\, (1 - 0.5))]$$

= -1 * -0.301029995

= 0.301029995

And for the last, we will analyze the result, like the accuracy, precision, recall, f1-score.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Function 4.7.9 Accuracy Function

The following is an accuracy function where the False Positive and False-negative values are almost the same. The description of this function is, TN is True-negative, FP is False-positive, and is False-negative.

$$Precision = \frac{TP}{TP + FP}$$

Function 4.7.10 Precision Function

Then precision meaning is the ratio of the correct positive predictions to the total positive predictions. The description of this function is, TP is True-positive, and is False-positive.

$$Recall = \frac{TP}{TP + FN}$$

Function 4.7.11 Recall Function

Then recall meaning is the ratio of the correct positive predictions from all the original data. The description of this function is,TP is True-positive, and is False-negative.

$$F1\ Score = \frac{2 \times recall \times precision}{recall + precision}$$

Function 4.7.12 F-1 Score Function

Fi score meaning is the average of recall and precision.

4.8. Building model using Extreme Gradient Boosting

In the first xgboost, the author will initialize the calculation of the prediction score with a value of 1.

And then the loss for each data will be calculated using the gradient & hessian function formula.

$$g = l'^{(\hat{y})} = \frac{\partial l}{\partial \hat{y}} = \frac{1}{1 + e^{-(\hat{y})}} - y$$

Function 4.8.1 Gradient Function

l' = first loss

$\hat{y}$ = prediction

y = true value

$$h = l''^{(\hat{y})} = \frac{\partial^2 l}{\partial^2 \hat{y}} = p(1 - p)$$

Function 4.8.2 Hessian Function

l" = second loss

$\hat{y}$ = prediction

p = sigmoid from prediction label

For the example of calculation author use 1 row data of dataset, shown as below

$$g \ = l'^{(\hat{y})} = \frac{\partial l}{\partial \hat{y}} = \frac{1}{1 + e^{-(\hat{y})}} - y$$

$$= \frac{1}{1+e^{-(1)}} - 1$$

$$= 0.268941421$$

$$h \ = l''^{(\hat{y})} = \frac{\partial^2 l}{\partial^2 \hat{y}} = \ p(1-p)$$

$$= \frac{1}{1+e^{\hat{y}}} \left(1 - \frac{1}{1+e^{\hat{y}}}\right)$$

$$= 0.731058578\ (1 - 0.731058578)$$

$$= 0.1966119335326179$$

Then from the results of the loss function, a tree will be created and implemented rank formula:

$$rk(z) = \frac{1}{\sum_{(x,h)\epsilon D_k} h} \sum_{(x,h)\epsilon D_k x<z} h$$

Function 4.8.3 Rank Function

$\sum_{(x,h)\epsilon D_k} h$ = total hessian from all data

$\sum_{(x,h)\epsilon D_k x<z} h$ = total hessian from all eligible data, where all data has a value less than the current value

h = hessian

For the example of calculation author use 1 feature data of dataset called credit history which the label is 1, shown as below

$$rk(z) = \frac{1}{\sum_{(x,h)\epsilon D_k} h} \sum_{(x,h)\epsilon D_k x<z} h$$

$$= \frac{1}{80.21766876252457} * 500 * 0.1966119335326179 \ = 1.2549$$

And for the limiting value for splitting data, using the split gain function formula:

$$L_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\epsilon IL} \quad g_i\right)^2}{\sum_{i\epsilon IL} \quad g_i + \lambda} + \frac{\left(\sum_{i\epsilon IR} \quad g_i\right)^2}{\sum_{i\epsilon IR} \quad g_i + \lambda} - \frac{\left(\sum_{i\epsilon I} \quad g_i\right)^2}{\sum_{i\epsilon I} \quad g_i + \lambda} - \gamma\right]$$

Function 4.8.4 Split Gain Function

$\sum_{i\epsilon IL} \quad g_i$ = total left gradient tree

$\sum_{i\epsilon IR} \quad g_i$ = total right gradient tree

$\sum_{i\epsilon I} \quad g_i$ = total gradien (left gradien tree + right gradien tree)

$\sum_{i\epsilon IL} \quad h_i$ = total left hessian tree

$\sum_{i\epsilon IR} \quad h$= total right hessian tree

$\sum_{i\epsilon I} \quad h_i$ = total hessian (left hessian tree + right hessian tree)

λ = lambda

γ = gamma

For the example of calculation author use 1 feature data of dataset called credit history which the label is 1, shown as below

$$L_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\epsilon IL} \quad g_i\right)^2}{\sum_{i\epsilon IL} \quad g_i + \lambda} + \frac{\left(\sum_{i\epsilon IR} \quad g_i\right)^2}{\sum_{i\epsilon IR} \quad g_i + \lambda} - \frac{\left(\sum_{i\epsilon I} \quad g_i\right)^2}{\sum_{i\epsilon I} \quad g_i + \lambda} - \gamma\right]$$

$$= \frac{1}{2}\left[\frac{(-0.2689)^2}{0.1966 + 1} + \frac{(16.5408)^2}{80.021 + 1} - \frac{(16.003)^2}{82.2176 + 1} - 0\right]$$

$$= \frac{1}{2}\left[\frac{(-0.2689)^2}{1.1966} + \frac{(16.5408)^2}{81.021} - \frac{(16.003)^2}{83.2176}\right]$$

$$= \frac{1}{2}\left[\frac{-0.07230721}{1.1966} + \frac{273.598}{81.021} - \frac{256.096009}{83.2176}\right]$$

$$= \frac{1}{2}[-0.06 + 3.37 - 3.07]$$

$$= \frac{1}{2}[0.24]$$

$$= 0.12$$

After that the author looks for candidates for tree branches with the formula candidate split function.

$$\left|rk(s_{kj}) - rk(s_{kj+1})\right| < \in, \qquad s_{k1} = min_i x_{ik}\ , \qquad s_{k1} = max_i x_{ik}$$

*Function 4.8.5*Find the Candidate Split Function

$rk(s_{kj})$ = ranking from the first data

$rk(s_{kj+1})$ = ranking from the second data

$\in$ = epsilon

$s_{k1} = min_i x_{ik}$ = first input data

$s_{k1} = max_i x_{ik}$ = last input data

For the example of calculation the author uses 1 feature data of the dataset called credit history which the label is 1, from the dataset the first data is 0 and second data is 0 and the epsilon is 0.003. Shown as below

$$\left|rk(s_{kj}) - rk(s_{kj+1})\right| < \in, \qquad s_{k1} = min_i x_{ik}\ , \qquad s_{k1} = max_i x_{ik}$$

$$= |0 - 0| < 0.003\ , so\ the\ result\ is\ the\ data\ propose\ a\ split$$

These results are used to calculate the new prediction score, by adding the old prediction score with the prediction from the tree that has been formed (leaf function score) multiplied by the learning rate.

$$wj = \frac{g}{h + \lambda}$$

Function 4.8.6 Score Function

$$\hat{y} := \hat{y} - lr * wj$$

Function 4.8.7 New Score Prediction Function

wj = score leaf

g = gradient

h= hessian

λ = lambda

$\hat{y}:$ = new prediction

$\hat{y}$ = prediction

For the example of calculation author use 1 feature data of dataset called credit history which the label is 1, shown as below

$$wj = \frac{g}{h + \lambda}$$

$$= \frac{-0.2689}{0.1966 + \lambda}$$

$$= 0.22472$$

$$\hat{y} := y - lr * wj$$

$$= 1 - 0.03 * 0.22472$$

$$= 1 - 0.0067416$$

$$= 0.9932584$$

Then it is iterated again by calculating the loss of the latest prediction score. To get prediction results, if the prediction score of a data is below average, the prediction score for all data will be 0, whereas if the prediction score for a data is above average, the prediction score for all data will be 1.

$$y=1 \text{ when } \hat{y} \geq 0.5$$

$$y=0 \text{ when } \hat{y} < 0.5$$

Function 4.8.8 Predict Function

y = true value

$\hat{y}$ = prediction value

For the example of calculation from calculation **Function 4.7.9** the prediction result is y=1

After that the author calculates the accuracy, precision, recall, and f1-score.

The following is an accuracy function where the False Positive and False-negative values are almost the same. The description of this function is, TN is True-negative, FP is False-positive, and is False-negative.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Function 4.8.9 Accuracy Function

Then precision meaning is the ratio of the correct positive predictions to the total positive predictions. The description of this function is, TP is True-positive, and is False-positive.

$$Precision = \frac{TP}{TP + FP}$$

Function 4.8.10 Precision Function

Then recall meaning is the ratio of the correct positive predictions from all the original data. The description of this function is,TP is True-positive, and is False-negative.

$$Recall = \frac{TP}{TP + FN}$$

Function 4.8.11 Recall Function

Fi score meaning is the average of recall and precision.

$$F1\ Score = \frac{2 \times recall \times precision}{recall + precision}$$

Function 4.8.12 F1-Score Function