By:

Karim Akmal & Omnia Elmenshawy

Credit Score Classification

Agenda

- Intro to dataset
- Dataset Problems
- Fixing the Data Noise
- Unnecessary Columns
- Categorical Variables
- Missing Values
- Outliers and Noise
- Skewness

Intro to dataset

Overview

Dataset Statistics

Number of Variables	28
Number of Rows	100000
Missing Cells	60071
Missing Cells (%)	2.1%
Duplicate Rows	0
Duplicate Rows (%)	0.0%
Total Size in Memory	135.5 MB
Average Row Size in Memory	1.4 KB
Variable Types	Categorical: 20 Numerical: 8

Dataset Insights

`Name` has 9985 (9.98%) missing values	Missing
`Monthly_Inhand_Salary` has 15002 (15.0%) missing values	Missing
`Type_of_Loan` has 11408 (11.41%) missing values	Missing
`Num_of_Delayed_Payment` has 7002 (7.0%) missing values	Missing
`Num_Credit_Inquiries` has 1965 (1.96%) missing values	Missing
`Credit_History_Age` has 9030 (9.03%) missing values	Missing
`Amount_invested_monthly` has 4479 (4.48%) missing values	Missing
`Monthly_Balance` has 1200 (1.2%) missing values	Missing
`Num_Bank_Accounts` is skewed	Skewed
`Num_Credit_Card` is skewed	Skewed

1 2 3

Dataset Insights

`Type_of_Loan` has a high cardinality: 6260 distinct values	High Cardinality
`Num_of_Delayed_Payment` has a high cardinality: 749 distinct values	High Cardinality
`Changed_Credit_Limit` has a high cardinality: 4384 distinct values	High Cardinality
`Outstanding_Debt` has a high cardinality: 13178 distinct values	High Cardinality
`Credit_History_Age` has a high cardinality: 404 distinct values	High Cardinality
`Amount_invested_monthly` has a high cardinality: 91049 distinct values	High Cardinality
`Monthly_Balance` has a high cardinality: 98792 distinct values	High Cardinality
`ID` has all distinct values	Unique
`Num_Credit_Inquiries` has 6972 (6.97%) zeros	Zeros
`Total_EMI_per_month` has 10613 (10.61%) zeros	Zeros

Dataset Insights

`Interest_Rate` is skewed	Skewed
`Num_Credit_Inquiries` is skewed	Skewed
`Total_EMI_per_month` is skewed	Skewed
`ID` has a high cardinality: 100000 distinct values	High Cardinality
`Customer_ID` has a high cardinality: 12500 distinct values	High Cardinality
`Name` has a high cardinality: 10139 distinct values	High Cardinality
`Age` has a high cardinality: 1788 distinct values	High Cardinality
`SSN` has a high cardinality: 12501 distinct values	High Cardinality
`Annual_Income` has a high cardinality: 18940 distinct values	High Cardinality
`Num_of_Loan` has a high cardinality: 434 distinct values	High Cardinality

Dataset Problems

- Some columns such as Age and number of bank account have negative values which could be considered as data noise.
- Some columns has extreme values, such as the Age column, which has customers aging around 8600 years old.
- Some columns are skewed.
- Some columns has values that does not have meaning such as the “NM” value in the

  Payment_of_min_amount column.

Handling dataset problems column by column

ID, Name, and SSN columns

- These are unnecessary columns which has no correlation with our dataset target, so they were dropped.

Categorical Columns handled by Label Encoder

- Customer_ID, Month
- Occupation
- Credit_Mix
- Payment_of_min_amount
- Credit_Score
- Payment_Behaviour and columns

Numerical columns classified as Objects, handled by the reges function

- Age
- Annual_income
- Num_of_loan
- Num_of_delayed_payment
- Changed_credit_limit
- Outstanding_debtCredit_Mix
- Payment_of_min_amount
- Amount_invested_monthly
- Monthly _BalanceC

Outliers has been handled by the IQR of each column

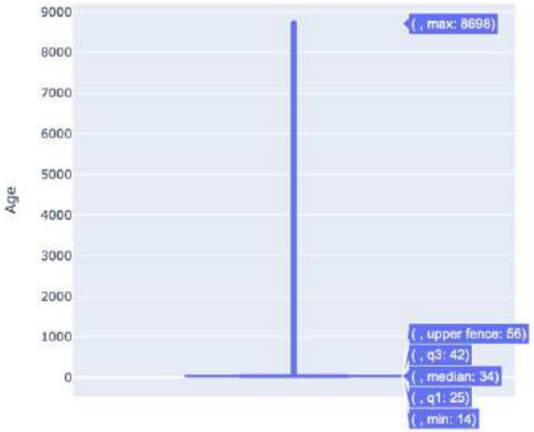
Example: Age column

Noise

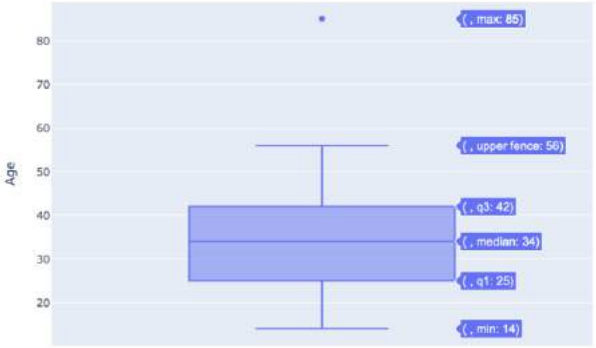
- Age column has negative values such as -500

Outliers

- Age column has extreme values with max of 8698 years old.



- We have put a lower bound = 14 years old, and upper Bound of 85 years old and replaced the outliers



Customer ID Column

Noise

- The column is a categorical column, thus, we deleted the characters of each row, so that the column is transformed into numerical value.

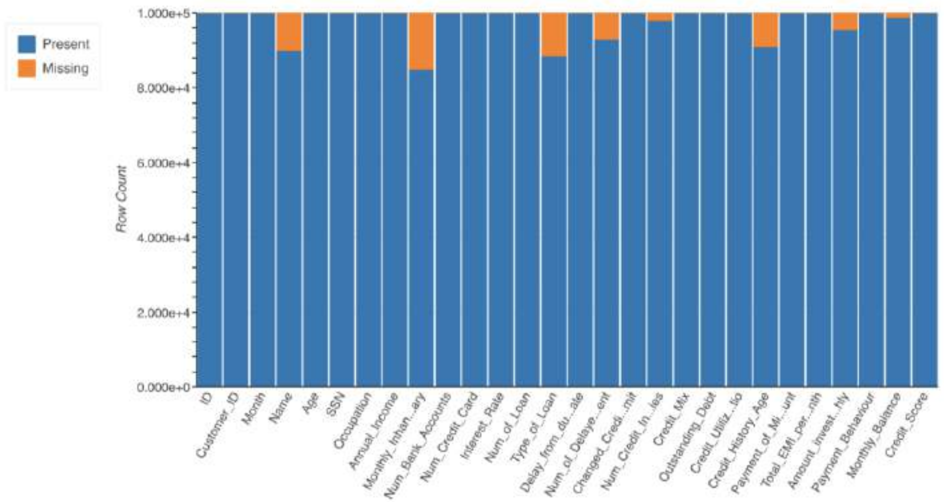
Analysis

- The column divides the dataset into 12500 customer, each Customer_ID maps and strongly relates to some other features.
- Thus, The missing values of the dataset will be imputed relative to this column.

Missing Values

- As every column in the dataset is related to the Customer ID column, we conclude that each customer has 8 rows in the dataset.

- Thus, using KNN Imputer with 5 neighbors would give the best predictions for filling the missing values.




```
ID                            0
Customer_ID                   0
Month                         0
Name                       9985
Age                           0
SSN                           0
Occupation                    0
Annual_Income                 0
Monthly_Inhand_Salary     15002
Num_Bank_Accounts             0
Num_Credit_Card               0
Interest_Rate                 0
Num_of_Loan                   0
Type_of_Loan              11408
Delay_from_due_date           0
Num_of_Delayed_Payment     7002
Changed_Credit_Limit          0
Num_Credit_Inquiries       1965
Credit_Mix                    0
Outstanding_Debt              0
Credit_Utilization_Ratio      0
Credit_History_Age         9030
Payment_of_Min_Amount         0
Total_EMI_per_month           0
Amount_invested_monthly    4479
Payment_Behaviour             0
Monthly_Balance            1200
Credit_Score                  0
dtype: int64
```

```
Customer_ID                 0
Month                       0
Age                         0
Occupation                  0
Annual_Income               0
Monthly_Inhand_Salary       0
Num_Bank_Accounts           0
Num_Credit_Card             0
Interest_Rate               0
Num_of_Loan                 0
Delay_from_due_date         0
Num_of_Delayed_Payment      0
Changed_Credit_Limit        0
Num_Credit_Inquiries        0
Credit_Mix                  0
Outstanding_Debt            0
Credit_Utilization_Ratio    0
Credit_History_Age          0
Payment_of_Min_Amount       0
Total_EMI_per_month         0
Amount_invested_monthly     0
Payment_Behaviour           0
Monthly_Balance             0
Credit_Score                0
dtype: int64
```

Robust Scaler

We used Robust Scaler to rescale the dataset.

```
: array([[ 0.97135771, -0.14285714, -0.64705882, ...,  0.        ,
           0.        ,  0.        ],
        [ 0.97135771, -0.42857143, -0.64705882, ...,  0.        ,
           0.        ,  0.        ],
        [ 0.97135771,  0.71428571, 27.41176471, ...,  0.        ,
           0.        ,  0.        ],
        ...,
        [ 0.41339307,  0.42857143, -0.52941176, ...,  0.        ,
           0.        ,  0.        ],
        [ 0.41339307,  0.14285714, -0.52941176, ...,  0.        ,
           0.        ,  0.        ],
        [ 0.41339307, -0.71428571, -0.52941176, ...,  0.        ,
           0.        ,  0.        ]])
```

Modelling

Models Applied:

Logistic Classifier
KNN
GaussianNB
SVC
Random Forest
XGBOOST

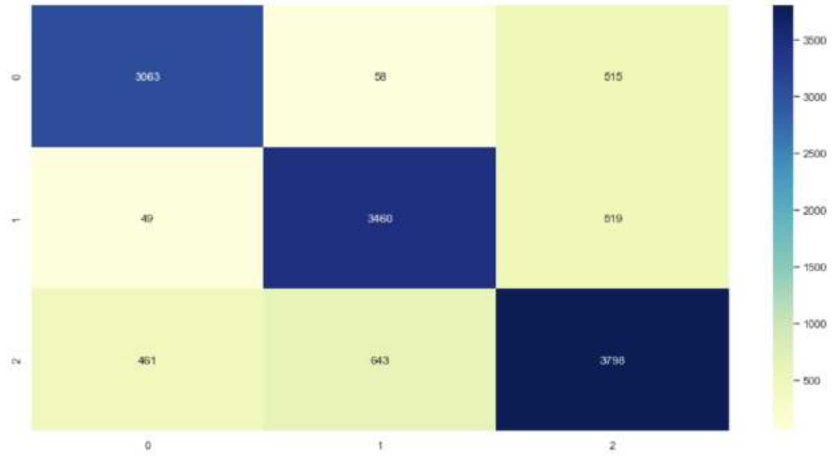
Best Model:

KNN

	precision	recall	f1-score	support
0.0	0.85	0.84	0.85	3636
1.0	0.83	0.86	0.84	4028
2.0	0.79	0.77	0.78	4902
accuracy			0.82	12566
macro avg	0.82	0.82	0.82	12566
weighted avg	0.82	0.82	0.82	12566

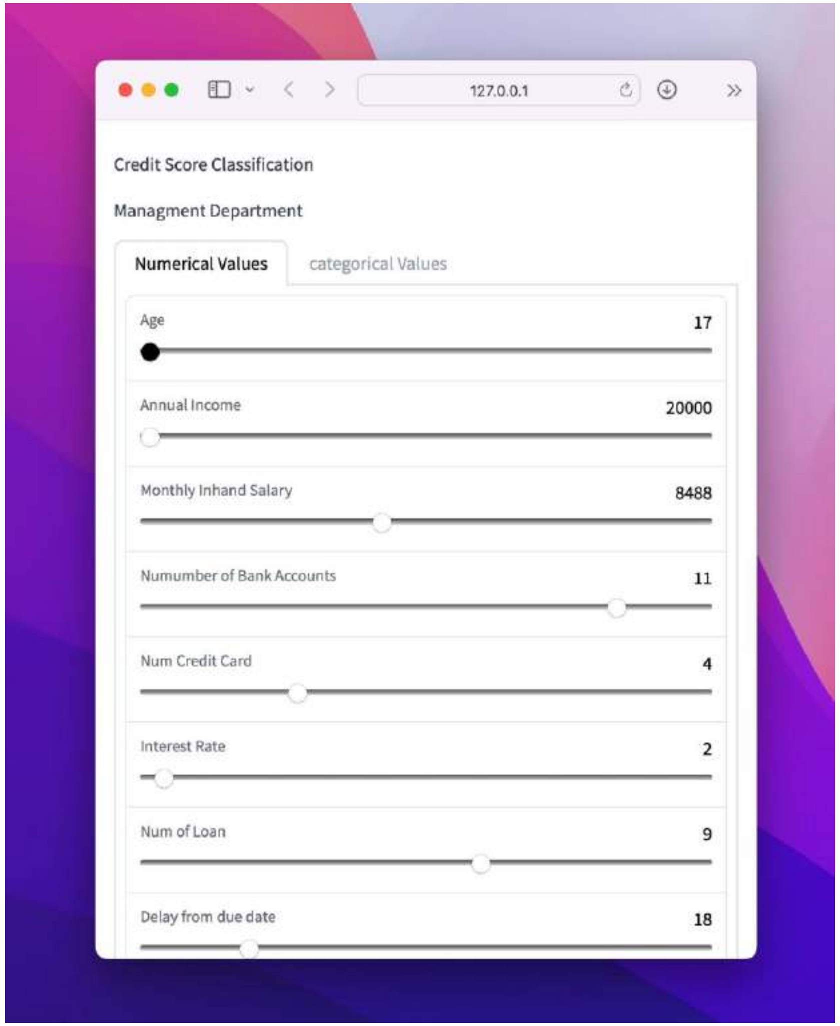
```
the score on train dataset is
0.9043412518403565
```

```
Test Accuracy :  0.8187171733248448
```



Model Deployment

We applied Gradio API as it gives a good quality user experince



Thank You!