Content

Chapter 1

Introduction

1.1 Problem Description

The objective of this project is to predict the bank loan default case with the help of various factors of customers. By achieving this goal, it would be possible to help accommodate in managing the customer for their loans on a daily basis, and providing better services to its customer.

1.2 Data

Our challenge is to build a high accuracy Binary Classification models which will be responsible for predicting bank loan default case based on certain variables. Given below is the sample data that is used. Data is been splitted based on target variable as test and train from index 700.

a) **Data frame** of csv file with top 5 entries as sample

	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

Number of attributes: There are 8 attributes/features/variables

Var. #	Variable Name	Description	Variable Type
1.	Age	Age of each customer	Numerical
2.	Education	Education categories	Categorical
3	Employment	Employment status - Corresponds to job status and being converted to numeric format	Numerical
4	Address	Geographic area - Converted to numeric values	Numerical
5	Income	Gross Income of each customer	Numerical
6	debtinc	Individual's debt payment to his or her gross income	Numerical
7	creddebt	debt-to-credit ratio is a measurement of how much you owe your creditors as a percentage of your available credit (credit limits)	Numerical

8	othdebt	Any other debts	Numerical

b) **Data types**

```
age            int64
ed             int64
employ         int64
address        int64
income         int64
debtinc      float64
creddebt     float64
othdebt      float64
default      float64
dtype: object
```

c) **Data Description** (Mean, Counts, Standard Deviation, Percentile, MinMax)

	age	ed	employ	address	income	debtinc	creddebt	othdebt	default
count	850.000000	850.000000	850.000000	850.000000	850.000000	850.000000	850.000000	850.000000	700.000000
mean	35.029412	1.710588	8.565882	8.371765	46.675294	10.171647	1.576805	3.078789	0.261429
std	8.041432	0.927784	6.777884	6.895016	38.543054	6.719441	2.125840	3.398803	0.439727
min	20.000000	1.000000	0.000000	0.000000	13.000000	0.100000	0.011696	0.045584	0.000000
25%	29.000000	1.000000	3.000000	3.000000	24.000000	5.100000	0.382176	1.045942	0.000000
50%	34.000000	1.000000	7.000000	7.000000	35.000000	8.700000	0.885091	2.003243	0.000000
75%	41.000000	2.000000	13.000000	12.000000	55.750000	13.800000	1.898440	3.903001	1.000000
max	56.000000	5.000000	33.000000	34.000000	446.000000	41.300000	20.561310	35.197500	1.000000

Chapter 2

Methodology

2.1 Pre-Processing

This Part is valuable to data science projects since it allows to get closer to the certainty that the future results will be valid, correctly interpreted, and applicable to the desired business contexts. Such level of certainty can be achieved only after raw data is validated and checked for anomalies, ensuring that the data set was collected without errors. EDA also helps to find insights that were not evident or worth investigating to business stakeholders and data scientists but can be very informative about a particular business.

This is called Exploratory Data Analysis (EDA) which helps to answer all these questions, ensuring the best outcomes for the project. It is an approach for summarizing, visualizing, and becoming intimately familiar with the important characteristics of a data set.

2.1.1 Missing Value

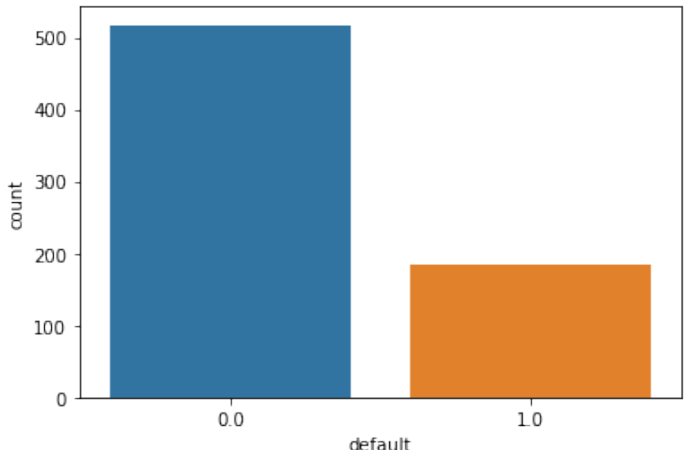
It is the first step to identify the rows that contains missing data which will lead to undesired results or create some errors if not removed.

	Missing Counts
age	0
ed	0
employ	0
address	0
income	0
debtinc	0
creddebt	0
othdebt	0

We can observe from this data frame that there is no missing Data in our Dataset. We can further proceed with EDA or visualization.

2.1.2 EDA Visualization

a) Distribution of target variable



Mapping of target variable as Default : 1, Non-Default : 0
From count plot, there is imbalance in target variable as **1 counts 517** and **0 counts 183**

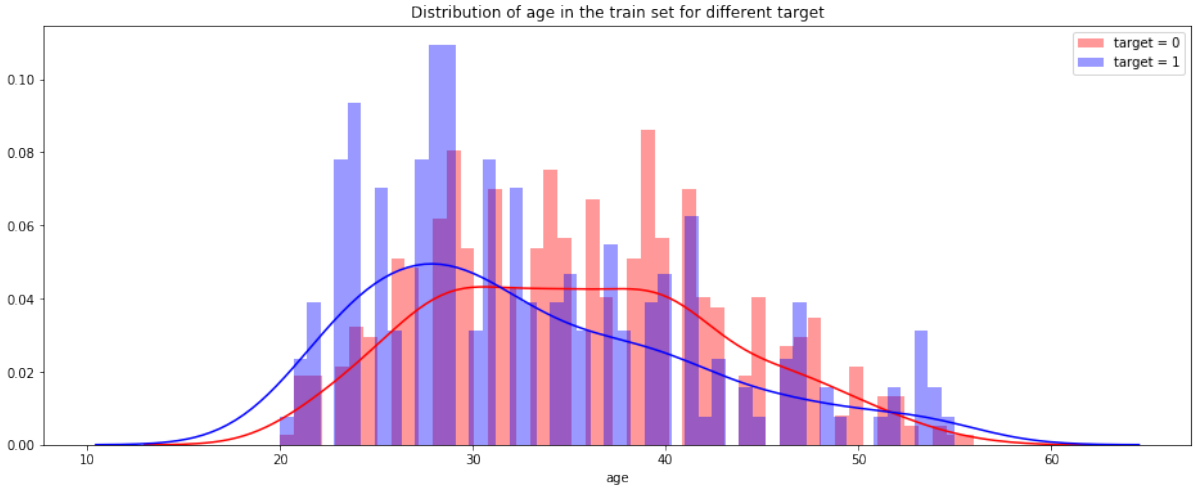
	default	0.0	1.0
age	mean	35.514507	33.010929
	median	35.000000	31.000000
	std	7.707736	8.517589
income	mean	47.154739	41.213115
	median	36.000000	29.000000
	std	34.220150	43.115529
creddebt	mean	1.245493	2.423865
	median	0.729000	1.376844
	std	1.422312	3.232522
othdebt	mean	2.773409	3.862807
	median	1.879790	2.529508
	std	2.813939	4.263684
employ	mean	9.508704	5.224044
	median	9.000000	3.000000
	std	6.663741	5.542946
debtinc	mean	8.679304	14.727869
	median	7.300000	13.800000
	std	5.615197	7.902798
address	mean	8.945841	6.393443
	median	8.000000	5.000000
	std	7.000621	5.925208

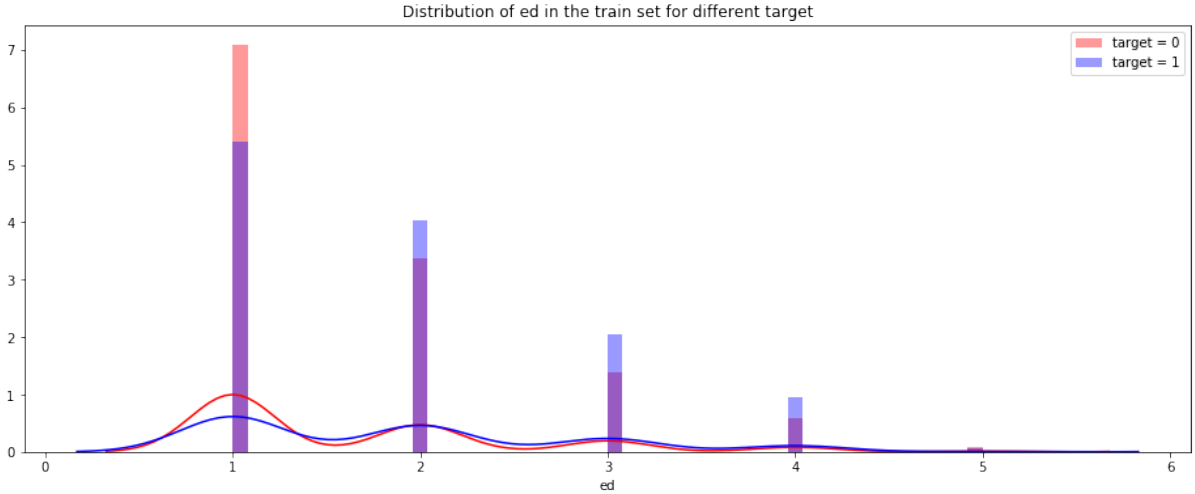
Distribution of each variable on the basis of target variable shown in bar graph and cumulative data present in table on left side.

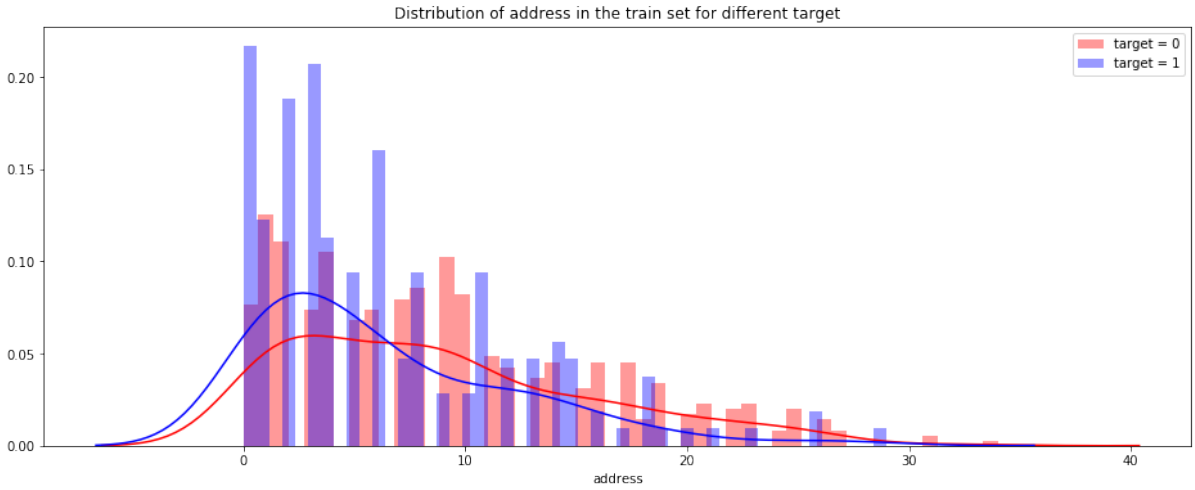
As overview of table there not so much difference between target variable 0 and 1. And Standard Deviation is different for every variable for there target variable.

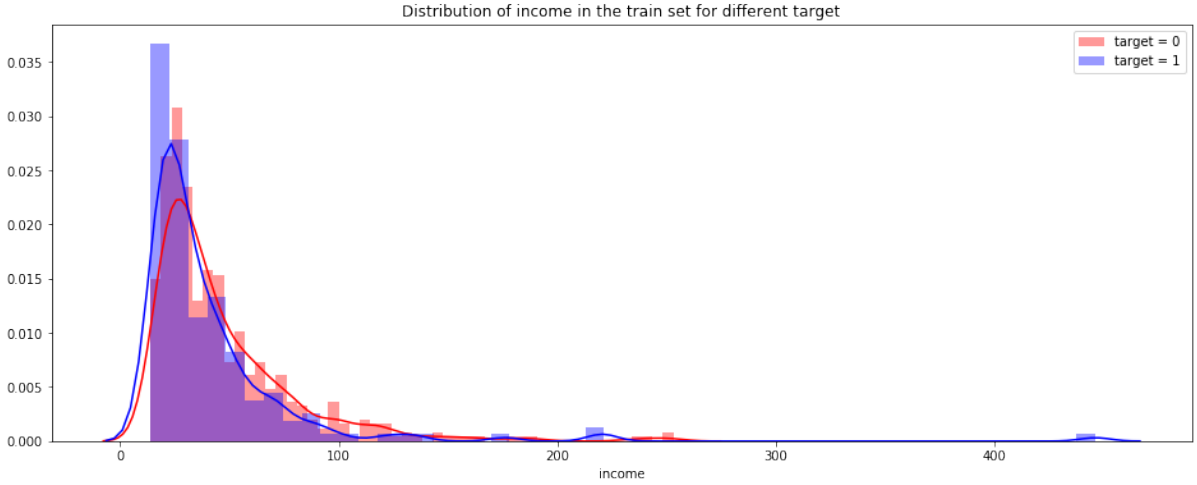
Distribution of variables shown in bar graph can be seen as distinguishable for target variables .

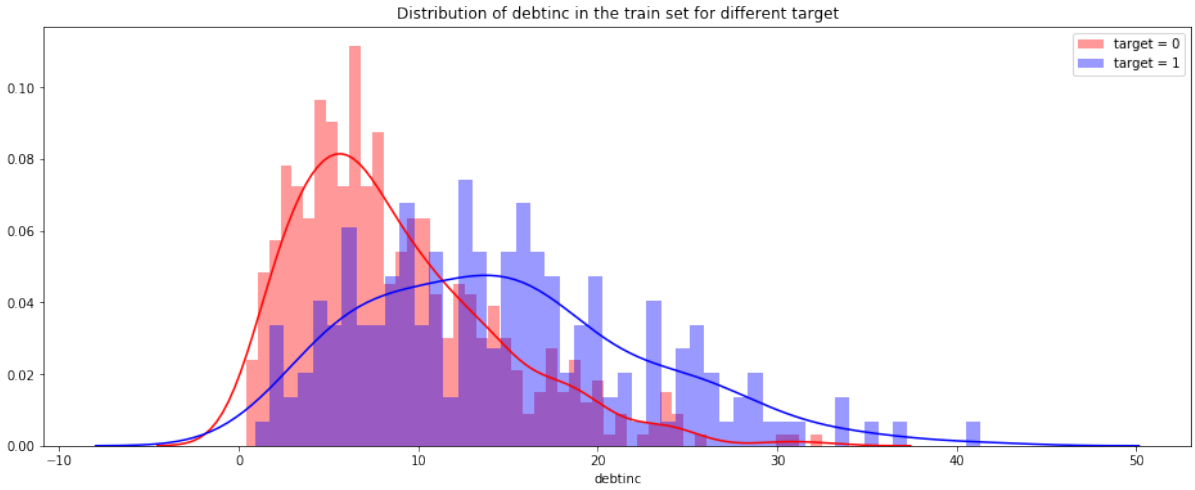
b) Distribution variables for different target variable

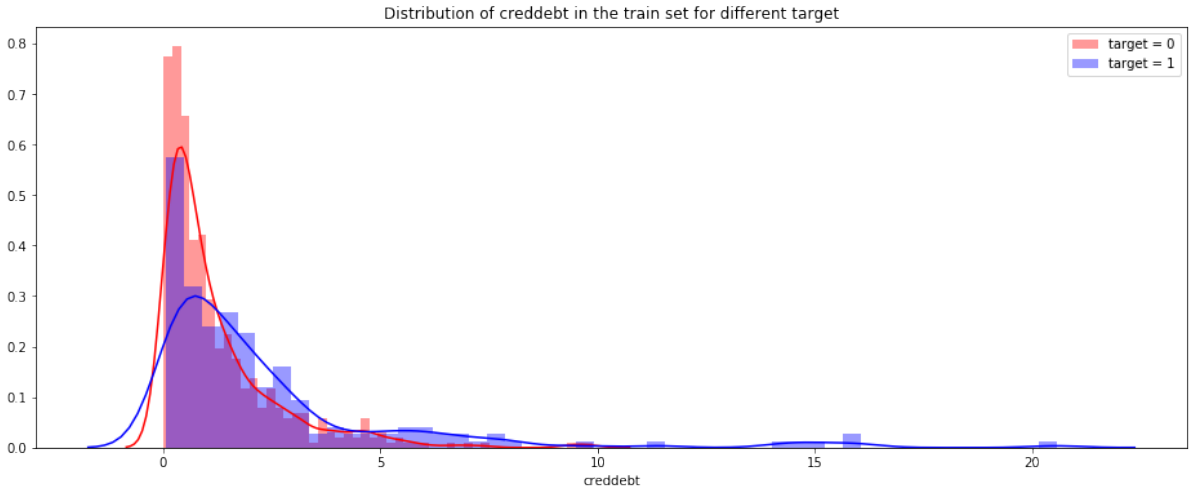


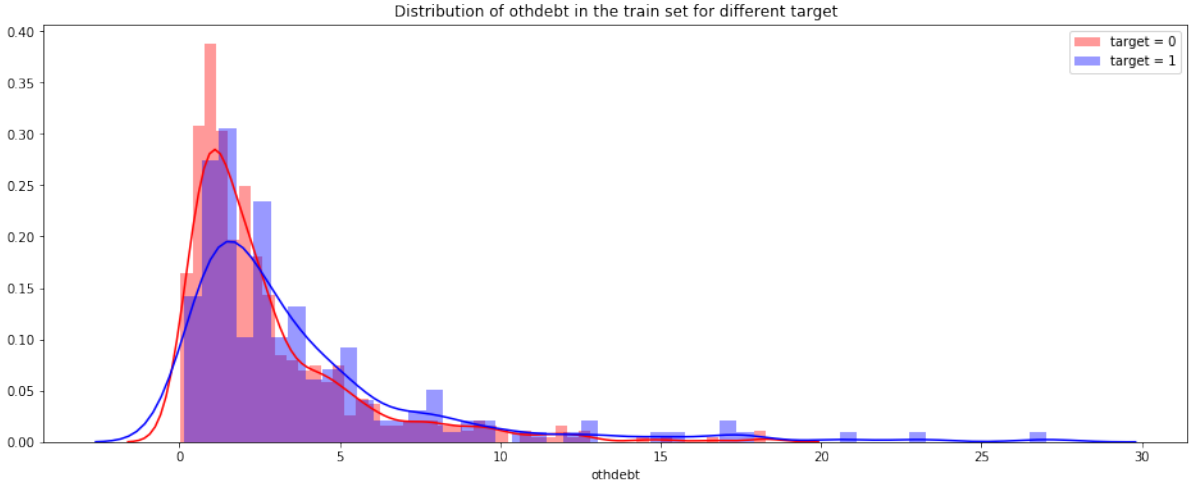



Distribution of address in the train set for different target
target = 0
target = 1
0.20
0.15
0.10
0.05
0.00
0
10
20
30
40
address


Distribution of income in the train set for different target
target = 0
target = 1
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
100
200
300
400
income


Distribution of debtinc in the train set for different target
target = 0
target = 1
0.10
0.08
0.06
0.04
0.02
0.00
−10
0
10
20
30
40
50
debtinc

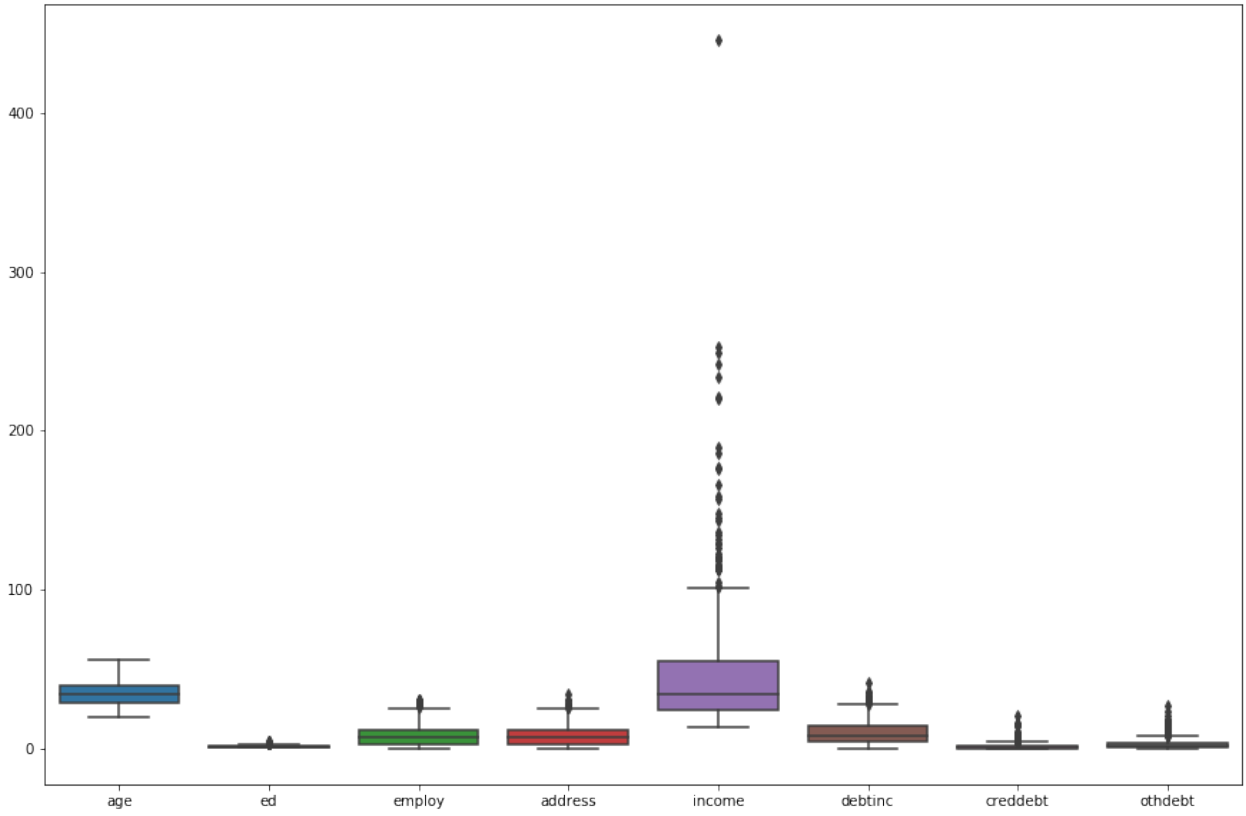

Distribution of creddebt in the train set for different target
target = 0
target = 1
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
creddebt


Distribution of othdebt in the train set for different target
target = 0
target = 1
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
25
30
othdebt

2.1.2 Outlier Analysis

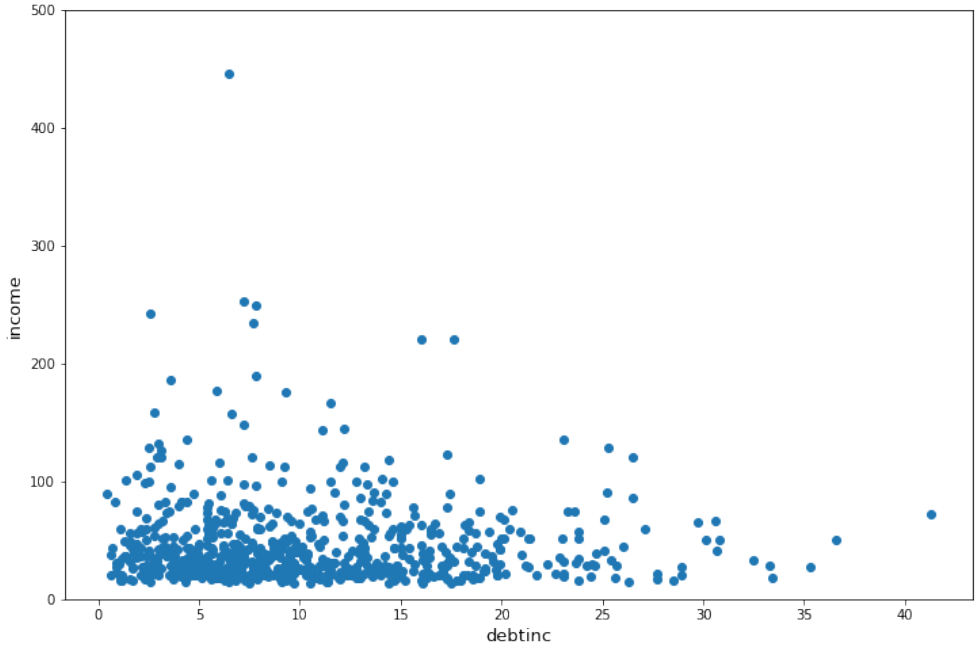
An outlier is an element of a data set that distinctly stands out from the rest of the data. In other words, outliers are those data points that lie outside the overall pattern of distribution.

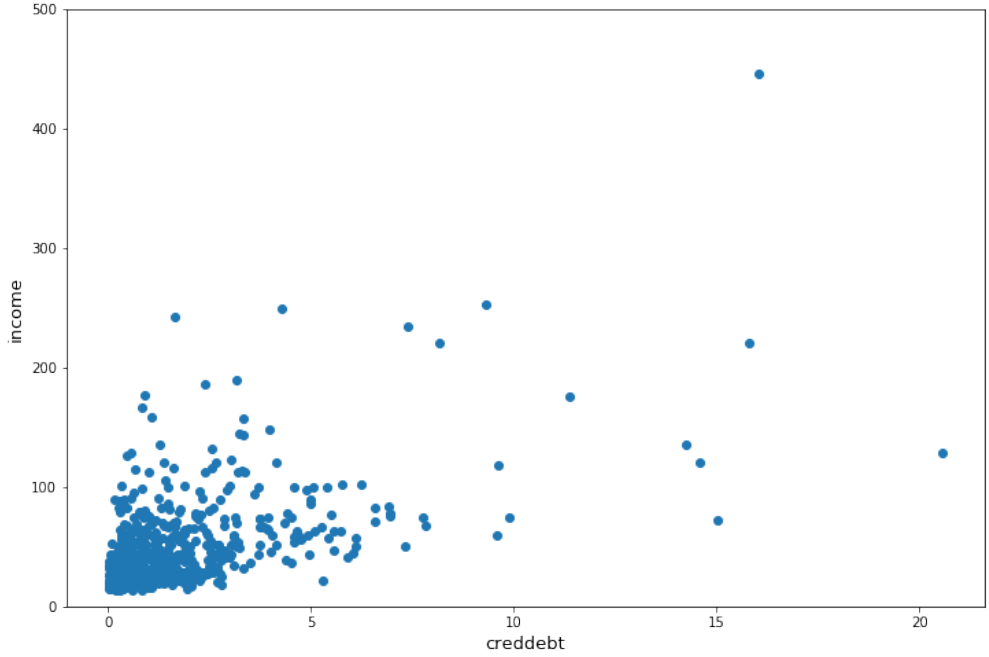
The easiest way to detect outliers is to create a graph. Plots such as Box plots, Scatterplots and Histograms can help to detect outliers. Alternatively, we can use mean and standard deviation to list out the outliers. Interquartile Range and Quartiles can also be used to detect outliers.



As we can see from above, the outliers in employ will not be removed because there may be case of some rare jobs with their status so we actually don't know what kind of employment is. Similarly for address variable, there may be case of some geographic areas where people are very less or with very high density. Not removing income outlier because there may be case where people could have very less income or very high income. Not removing othdebt outlier because there may be case where customers other debt may be different from other customers. We will remove only debtinc, creddebt outlier because after analysis of debtinc outlier, Individual debt of customer cannot be higher than its gross income.

For debtinc and creddebt outlier removal, scatter plot against with income can be efficient way to do. Below are scatter plots,



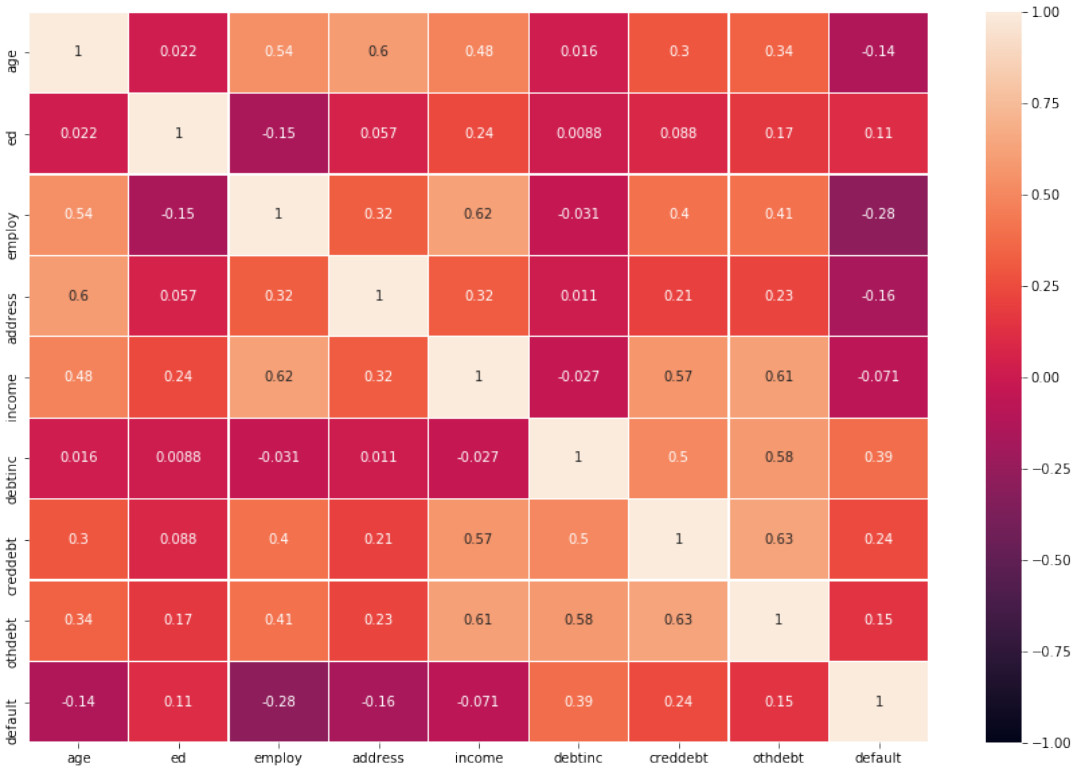


From analysing both figure, there are few outliers which are removed.

2.1.4 Multi Collinearity & Feature Selection

Multicollinearity occurs when independent variables in a Classification model are correlated. This correlation is a problem because independent variables should be independent. If the degree of correlation between variables is high enough, it can cause problems when you fit the model and interpret the results.

Below is graph of cross- correlation of variables



After filter the correlation there is no highly correlated in table shown above, Since there is not much correlation between variables thus dropping variable might not worth. Dropping an uncorrelated loose some of information which lead to decreases in evaluation.

VIF is also good tool to drop variable according to its value. Condition for dropping the variable needs the VIF in range of 5-10.

2.2 Modeling

2.2.1 Model Selection

Firstly very important method to split the data according to target variable. Since it binary classification target variable distributed randomly and if test train splitting is done randomly then it might chance of high difference between majority and minority group. Thus splitting proportionally in both train and valid dataset is efficient way which also knows as Stratification.

Before applying any algorithm Scaling is must since the variable which has high value i.e. Income in our case brings biasness towards them and making model performance bad. There are major 2 types of scaler one is standard Scaling and another one is MinMax which also known as Normalization scaling. By looking each variable histogram or distribution it is random and thus Standardization is not good way. Normalization will work better in this case.

Dealing with these imbalanced datasets

Imbalanced classes are a common problem in machine learning classification where there are a disproportionate ratio of observations in each class. Class imbalance can be found in many different areas including medical diagnosis, spam filtering, and fraud detection.

1. Change the performance metric

Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be misleading. Metrics that can provide better insight include:

- **Confusion Matrix:** a table showing correct predictions and types of incorrect predictions.
- **Precision:** the number of true positives divided by all positive predictions. Precision is also called Positive Predictive Value. It is a measure of a classifier's exactness. Low precision indicates a high number of false positives.
- **Recall:** the number of true positives divided by the number of positive values in the test data. Recall is also called Sensitivity or the True Positive Rate. It is a measure of a classifier's completeness. Low recall indicates a high number of false negatives.
- **F1: Score:** the weighted average of precision and recall.

Since our main objective with the dataset is to prioritize accurately classifying fraud cases the recall score can be considered our main metric to use for evaluating outcomes.

2. Change the algorithm

While in every machine learning problem, it’s a good rule of thumb to try a variety of algorithms, it can be especially **beneficial with imbalanced datasets**. Decision trees frequently perform well on imbalanced data. They work by learning a hierarchy of if/else questions. This can force both classes to be addressed.

Resampling Techniques

3. Oversampling Minority Class

Oversampling can be defined as **adding more copies of the minority class**. Oversampling can be a good choice when you don't have a ton of data to work with. A con to consider when under-sampling is that it can cause **overfitting and poor generalization** to your test set.
We will use the resampling module from Scikit-Learn to randomly replicate samples from the minority class.

Always split into test and train sets BEFORE trying any resampling techniques! Oversampling before splitting the data can allow the exact same observations to be present in both the test and train sets! This can allow our model to simply memorize specific data points and cause overfitting

4. Undersampling Majority Class

Undersampling can be defined as **removing some observations of the majority class**. Undersampling can be a good choice when you have a ton of data -think millions of rows. But a drawback to undersampling is that we are removing information that may be valuable.
We will again use the resampling module from Scikit-Learn to randomly remove samples from the majority class.

5. Boosting-Based Ensembling

Boosting is an ensemble technique to **combine weak learners to create a strong learner** that can make accurate predictions. Boosting starts out with a base classifier / weak classifier that is prepared on the training data.
The base learners / Classifiers are weak learners i.e. the prediction accuracy is only slightly **better than average**. A classifier learning algorithm is said to be weak when small changes in data induce big changes in the classification model.
In the next iteration, the new classifier **focuses on or places more weight** to those cases which were incorrectly classified in the last round.
I will use Recall, Precision, F1_score (harmonic mean of Precision and recall), AUC-ROC score along with Accuracy for model evaluation.

Algorithms

2.2.2 Logistic Regression

Logistic regression is the go-to method for binary classification problems (problems with two class values). Logistic regression is named for the function used at the core of the method, the logistic function. The logistic function, also called the sigmoid function.

This is the classification problem. We can use logistic regression for this problem. Logistic Regression is used when the dependent variable(target) is categorical. It has a **bias** towards classes which have large number of instances. It tends to only **predict the majority class data**. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of **misclassification** of the minority class as compared to the majority class.

	precision	recall	f1-score	support
0.0	0.83	0.90	0.87	103
1.0	0.63	0.47	0.54	36
accuracy			0.79	139
macro avg	0.73	0.69	0.70	139
weighted avg	0.78	0.79	0.78	139

2.2.3 Decision Tree

	precision	recall	f1-score	support
0.0	0.80	0.80	0.80	103
1.0	0.42	0.42	0.42	36
accuracy			0.70	139
macro avg	0.61	0.61	0.61	139
weighted avg	0.70	0.70	0.70	139

2.2.4 KNN

A k-nearest-neighbour algorithm, often abbreviated **k-nn**, is an approach to data **classification** that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in.
The k-nearest-neighbour is an example of a **"lazy learner"** algorithm, meaning that it does not build a model using the training set until a query of the data set is performed.

	precision	recall	f1-score	support
0.0	0.78	0.87	0.83	103
1.0	0.46	0.31	0.37	36
accuracy			0.73	139
macro avg	0.62	0.59	0.60	139
weighted avg	0.70	0.73	0.71	139

2.2.5. Random Forest

Random Forest is a bagging based **ensemble learning model**. Random forests is **slow** in generating predictions because it has **multiple decision trees**. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming. Thus result (AUC-score) shown below for random Forest is not modified if num round would increase the score will be better but speed will be very slow. These results are shown for default values of Parameters.

	precision	recall	f1-score	support
0.0	0.82	0.90	0.86	103
1.0	0.60	0.42	0.49	36
accuracy			0.78	139
macro avg	0.71	0.66	0.67	139
weighted avg	0.76	0.78	0.76	139

2.2.6 SVM

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both **classification** or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in **n-dimensional space** (where n is the number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the **hyper-plane** that differentiate the two classes very well. Support Vectors are simply the coordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line). It is effective in cases where the number of dimensions is greater than the number of samples. But it doesn’t perform well, when we have large data set because the required training time is higher.

	precision	recall	f1-score	support
0.0	0.84	0.90	0.87	103
1.0	0.64	0.50	0.56	36
accuracy			0.80	139
macro avg	0.74	0.70	0.72	139
weighted avg	0.79	0.80	0.79	139

Conclusion

This was a classification problem on a typically **unbalanced dataset with no missing values.** Predictor variables are numeric and target variable is numeric. Visualizing descriptive features and finally I got to know that these variables are not correlated among themselves. After that I decided to treat data with different models with original data and chosen **Logistic Regression** for resampling of minority class as my final model then using the same model with feature engineered data we got **Precision of 81**.

	precision	recall	f1-score	support
0.0	0.83	0.90	0.87	103
1.0	0.63	0.47	0.54	36
accuracy			0.79	139
macro avg	0.73	0.69	0.70	139
weighted avg	0.78	0.79	0.78	139