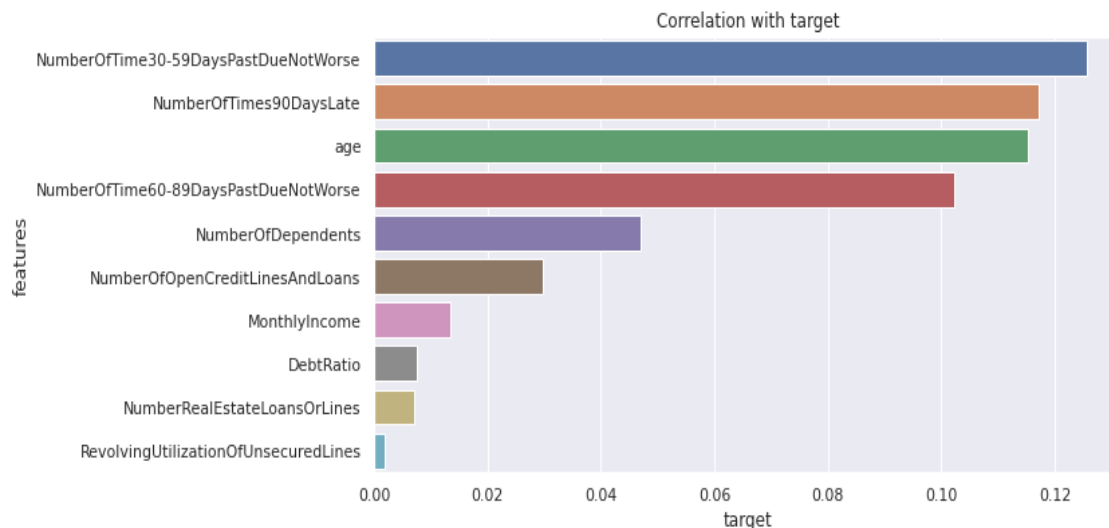


Part 1:

a) What are the factors that have high correlation with the probability of loan default?

Ans. According to the dataset provided factors that may have high correlation with the probability of loan default include:



- **NumberOfTime30-59DaysPastDueNotWorse:** This variable represents the number of times the borrower has been 30-59 days past due on their payments in the last 2 years. A high value for this variable may indicate that the borrower has a history of making late payments, which may increase the risk of default.
- **NumberOfTimes90DaysLate:** This variable represents the number of times the borrower has been 90 days or more past due on their payments. A high value for this variable may indicate that the borrower has a history of making very late payments or missing payments altogether, which may significantly increase the risk of default.
- **NumberOfTime60-89DaysPastDueNotWorse:** This variable represents the number of times the borrower has been 60-89 days past due on their payments in the last 2 years. A high value for this variable may indicate that the borrower has a history of making late payments, which may increase the risk of default.
- **Age:** Age also plays a crucial role in deciding if the user will default loan or not. If borrower is too old, then it could result in risk of default as this person might succumb to death or could get sick very often due to ageing.

It is important to note that these are just some of the factors that may have high correlation with the probability of loan default. Other factors, such as the `RevolvingUtilizationOfUnsecuredLines`, the number of dependents in their family or `NumberRealEstateLoansOrLines`, may also be important.

b) Are there interaction effects occurring among the variables?

Ans. From the information provided about the features in Data dictionary, we can assume that following features may have interaction effects among each other:

- **DebtRatio**
- **MonthlyIncome**
- **RevolvingUtilizationOfUnsecuredLines**
- **NumberOfOpenCreditLinesAndLoans**

Debt ratio: This variable represents the ratio of monthly debt payments (including alimony and living costs) to monthly gross income. A higher debt ratio may indicate that an individual has a higher burden of debt relative to their income, which could be a risk factor for financial distress. Monthly income: This variable represents the amount of money an individual earns each month. A higher monthly income may be associated with a lower debt ratio, as the individual has more income available to pay off their debts. Revolving utilization of unsecured lines: This variable represents the total balance on credit cards and personal lines of credit (excluding real estate and no instalment debt such as car loans) divided by the sum of credit limits. A higher revolving utilization may indicate that an individual is using a larger portion of their available credit, which could be a risk factor for financial distress. Number of open credit lines and loans: This variable represents the total number of open loans (instalment loans such as car loans or mortgages) and lines of credit (e.g., credit cards) that an individual has. A higher number of open credit lines and loans may be associated with a higher debt ratio and a higher revolving utilization of unsecured lines, as the individual has more debts to pay off.

Performing perform the **Kruskal-Wallis test** on the given dataset: -

In **Kruskal-Wallis** we test to compare the distribution of the variables in different groups. If the p-value of the test is less than the significance level (usually 0.05), it means that there is a statistically significant difference in the distribution of the variables between the groups. This suggests that there may be an interaction effect among the variables.

Group of columns [DebtRatio, MontlyIncome, RevolvingUtilizationOfUnsecuredLines, NumberOfOpenCreditLinesAndLoan]

```
Kruskal-Wallis statistic: 137365.06041338432
Kruskal-Wallis p-value: 0.0
There is a significant difference in the distribution of the groups.
```

Hence there is interaction between these features or variables

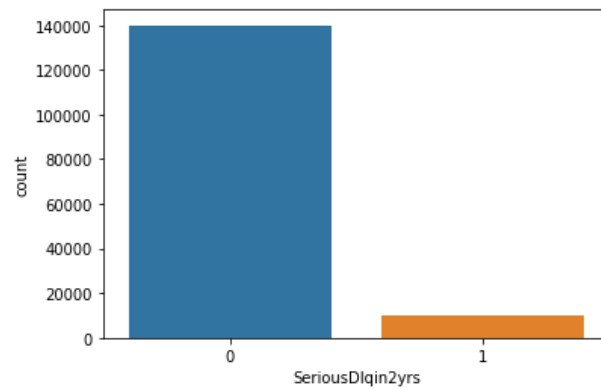
As the data is not normally distributed as seen from histogram of features, we can't use Anova test for finding the interaction effects between variables.

But we can also perform statistic model test using regression model to find interaction between individual variables:

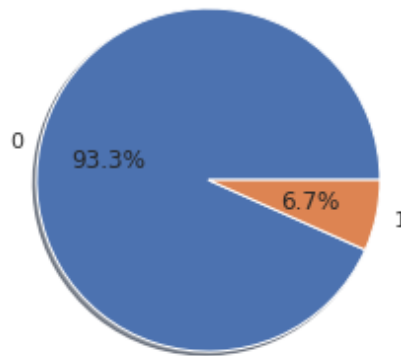
	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.1410	0.003	54.732	0.000	0.136	0.146
RevolvingUtilizationOfUnsecuredLines	-4.535e-06	2.6e-06	-1.747	0.081	-9.62e-06	5.54e-07
age	-0.0016	4.42e-05	-35.980	0.000	-0.002	-0.002
DebtRatio	-3.914e-07	3.08e-07	-1.269	0.204	-9.96e-07	2.13e-07
NumberOfTime30_59DaysPastDueNotWorse	0.0484	0.001	50.615	0.000	0.046	0.050
NumberOfTimes90DaysLate	0.0505	0.001	39.264	0.000	0.048	0.053
NumberOfTime60_89DaysPastDueNotWorse	-0.0927	0.001	-64.276	0.000	-0.095	-0.090
NumberOfOpenCreditLinesAndLoans	-0.0007	0.000	-5.628	0.000	-0.001	-0.000
NumberOfDependents	0.0049	0.001	8.425	0.000	0.004	0.006
DebtRatio:RevolvingUtilizationOfUnsecuredLines	4.226e-09	1.2e-09	3.527	0.000	1.88e-09	6.57e-09

c) Any other preliminary analysis of the given dataset?

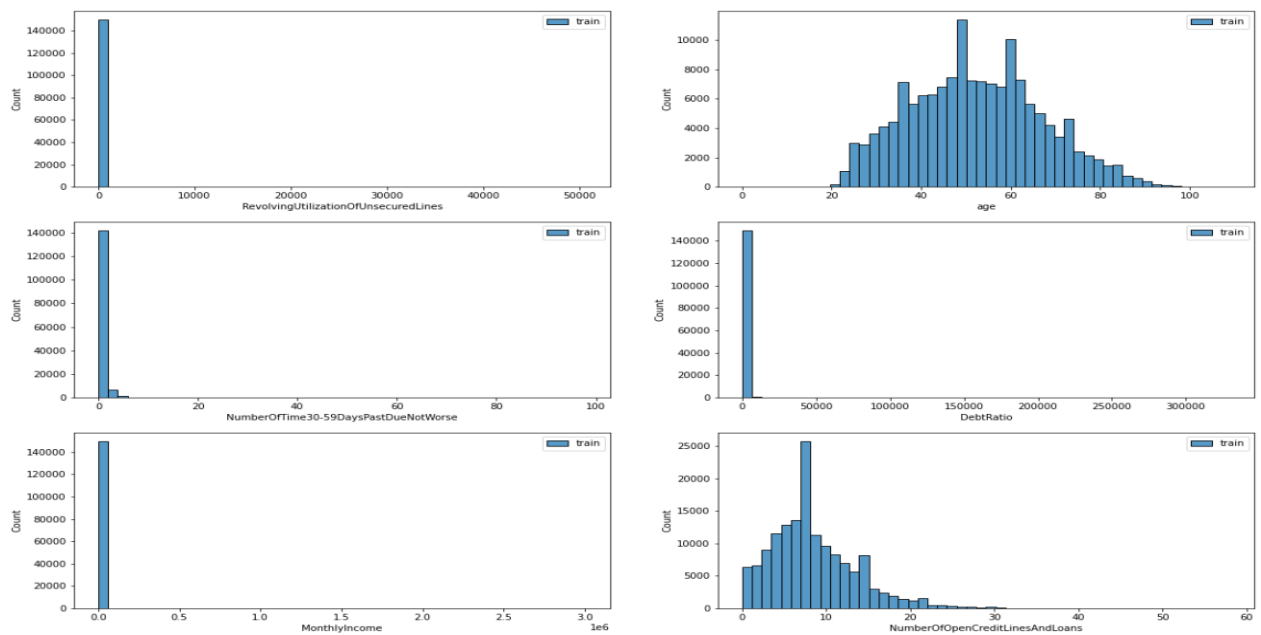
Ans. Target distribution

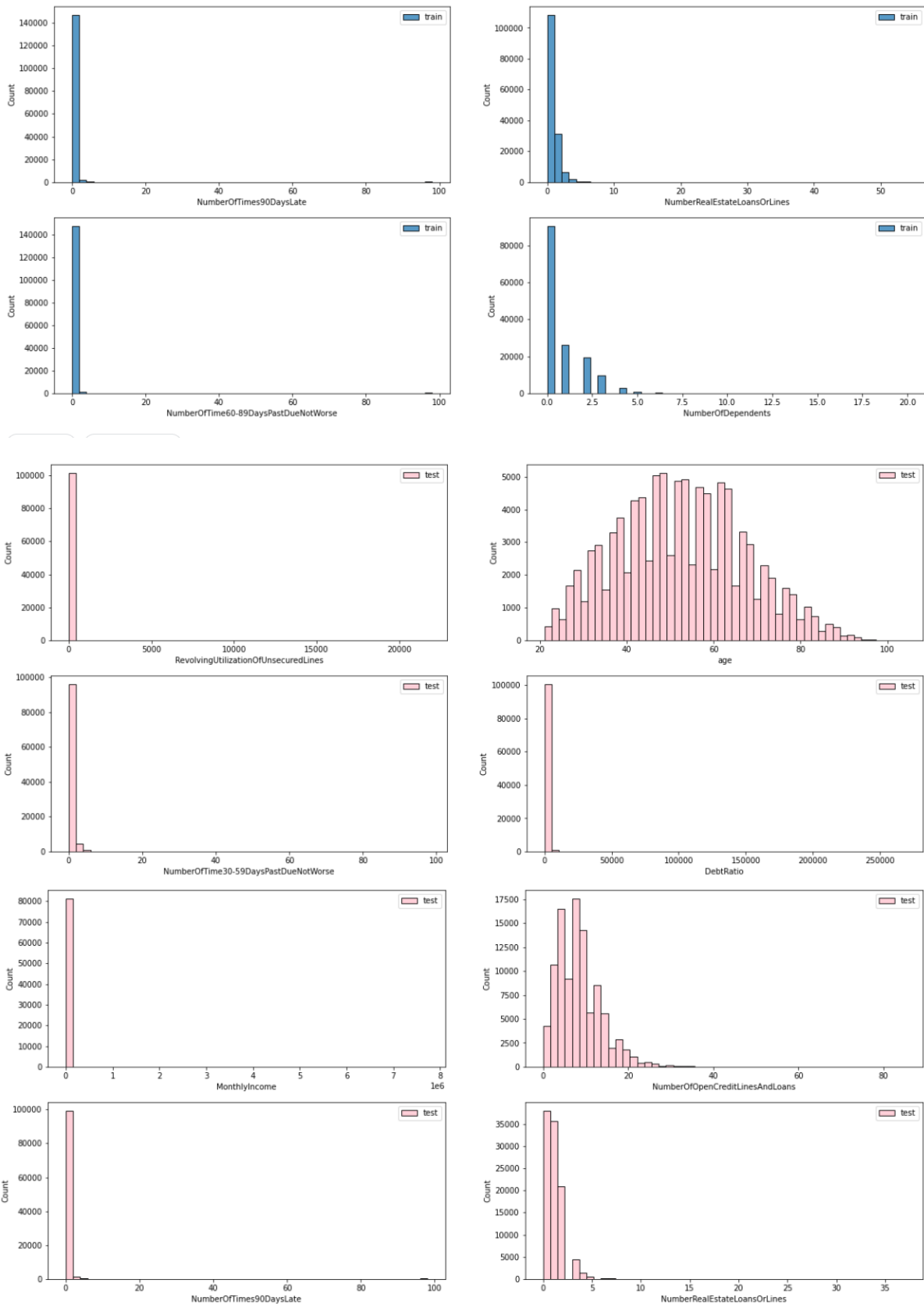


There is class imbalance in the target class. The underrepresented class of not_fraud(1) is way less than that of overrepresented is_fraud(0).



Histogram plots of features in train and test dataset:





It was observed that most of the features are either extremely right skewed or left skewed. Only age is normally distributed feature in both train and test dataset. MonthlyIncome has the highest min-max difference and NumberRealEstateLoansOrLines the lowest. Further analysis could be found at: Kaggle noteBook link - <https://www.kaggle.com/myyankshukla/give-credit>

Part 2:

a). Tell us how you validate your model and why you chose such evaluation technique(s).

Ans. Training and testing on different data: We split the input data set into training set and validation set. Model was trained on training set and was tested for accuracy on validation set.

```
[17] #splitting input dataset into train and validation set
      X_train, X_val, y_train, y_val = train_test_split(df_train.fillna(0), df_train_target['SeriousDlqin2yrs'].values, test_size=0.20,
```

We make an 8:2 split where training data consists of 8 parts of input data and validation set contains 20 parts of input data.

```
print(X_train.shape)
print(y_train.shape)
print(X_val.shape)
print(y_val.shape)
```

```
(120000, 10)
(120000,)
(30000, 10)
(30000,)
```

Fitting model on training set

```
# fir the model on best hyperparameters
pipe.fit(X_train, y_train)
```

Then validating the model on validation set.

```
# Test on Validation set
y_pred_val = pipe.predict(X_val)
print(accuracy_score(y_val, y_pred_val))
```

```
0.9371666666666667
```

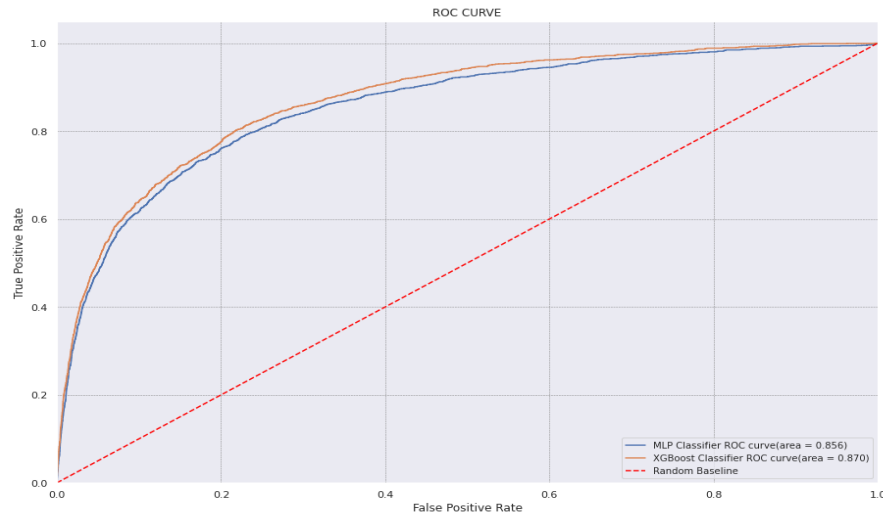
This method is useful for assessing the generalization error of the model, which is a measure of how well the model is expected to perform on new, unseen data.

Cross-validation: Cross-validation is a method for evaluating the performance of a model by training and testing it on different subsets of the data. here are several types of cross-validation, I used StratifiedKFold for cross validation. It is like K-fold However, in StratifiedKFold cross-validation, the folds are created such that the proportion of classes in the training and testing sets is the same as the proportion of classes in the entire dataset. This is useful for imbalanced datasets, where some classes are underrepresented, to ensure that the training and testing sets are representative of the overall distribution of classes in the dataset.

```
# Search for best hyperparameters
cv_strategy = StratifiedKFold(n_splits=2, shuffle=True) # Cross-validation Method
search = RandomizedSearchCV(pipe, parameter_space, n_jobs=-1, cv=cv_strategy, scoring='f1_weighted').fit(X_train, y_train)
```

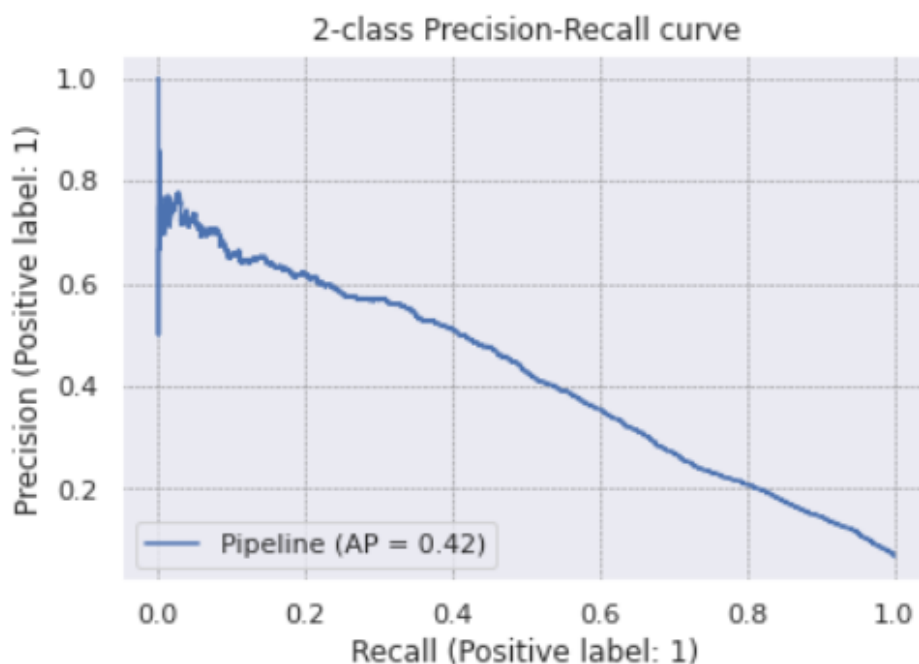
b) What is AUC? Why do you think AUC was used as the evaluation metric for this challenge? What other metrics do you think would also be suitable for this competition?

Ans. AUC (area under the curve) is a metric that is used to evaluate the performance of a binary classification model. It is the area under the receiver operating characteristic (ROC) curve, which is a plot of the true positive rate (sensitivity) against the false positive rate (1 - specificity) at different classification thresholds. AUC ranges from 0 to 1, with a higher value indicating a better model. AUC is a useful metric because it is insensitive to the classification threshold and is robust to imbalanced classes.



AUC was likely used as the evaluation metric because it is a suitable metric for evaluating the performance of a model that predicts the probability of financial distress. AUC is often used as an evaluation metric for binary classification models, where the goal is to predict a binary outcome (e.g., positive, or negative, fraud or not fraud). AUC is particularly useful in situations where the classes are imbalanced, where one class is underrepresented, because it is insensitive to the classification threshold and is robust to imbalanced classes.

Other metrics that would also be suitable for this competition include precision, recall, and F1 score, which are commonly used metrics for evaluating binary classification models.



PRC Curve for XGBoost

	precision	recall	f1-score	support
0	0.944793	0.991352	0.967512	27983.0000
1	0.620690	0.196331	0.298305	2017.0000
accuracy	0.937900	0.937900	0.937900	0.9379
macro avg	0.782741	0.593842	0.632909	30000.0000
weighted avg	0.923002	0.937900	0.922519	30000.0000

Quantitative Metrics for XGBoost

Following metric were also used for evaluating the models for this competition.

c) short explanation of what you tried. What worked and what did not work (ie. You might have tried different features/models before the final one).

Ans. Tried on 2 models with different hyper-parameter tuning methods and cross validation technique: -

1) Test 1:

Model used - MLP classifier

Feature Selection Method - RFECV

Estimator - LinearRegression

Encoding – Label

Cross validation strategy – StratifiedKFold

Hyperparameter tuning method – RandomizedSearchCV

```
parameter_space = {
    'classifier__hidden_layer_sizes': [(10,), (100,), (1000,),(50,50,50), (50,100,50), (10,100,10)],
    'classifier__activation': ['tanh', 'relu'],
    'classifier__solver': ['sgd', 'adam'],
    'classifier__alpha': [0.05,0.001,0.0001,0.00001,0.00001],
    'classifier__learning_rate': ['constant','adaptive'],
    'feature_select__min_features_to_select':[1,2,3]
}

# Search for best hyperparameters
cv_strategy = StratifiedKFold(n_splits=2, shuffle=True) # Cross-validation Method
search = RandomizedSearchCV(pipe, parameter_space, n_jobs=-1, cv=cv_strategy, scoring='f1_weighted').fit(X_train, y_train)
```

Accuracy on validation set: **93.71%**

```
# Test on Validation set
y_pred_val = pipe.predict(X_val)
print(accuracy_score(y_val, y_pred_val))
```

0.9371666666666667

Accuracy on Test set:



Test_Result.csv

Complete (after deadline) · 4d ago

0.83356

0.82558



1) Test 2:

Model used - XgBoost classifier

Feature Selection Method - SelectFromModel

Estimator - LinearSVC

Encoding – Label

Cross validation strategy – StratifiedKFold

Hyperparameter tuning method – RandomizedSearchCV

Accuracy on validation set: **93.81%**

```
!:\n y_pred_val1 = pipe1.predict(X_val)\n print(accuracy_score(y_val, y_pred_val1))
```

0.9381666666666667

Accuracy on Test set:



Test_Result.csv

Complete (after deadline) · 2h ago

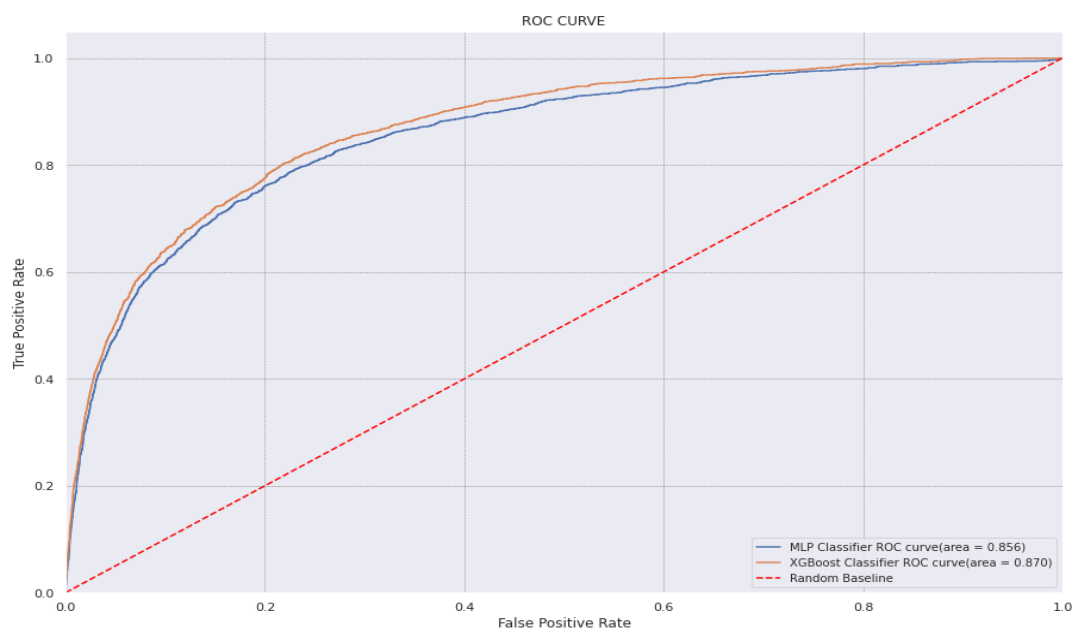
0.86731

0.86053



Xgboost performed better than mlp classifier on both validation and test set.

Comparing ROC curve of both models:



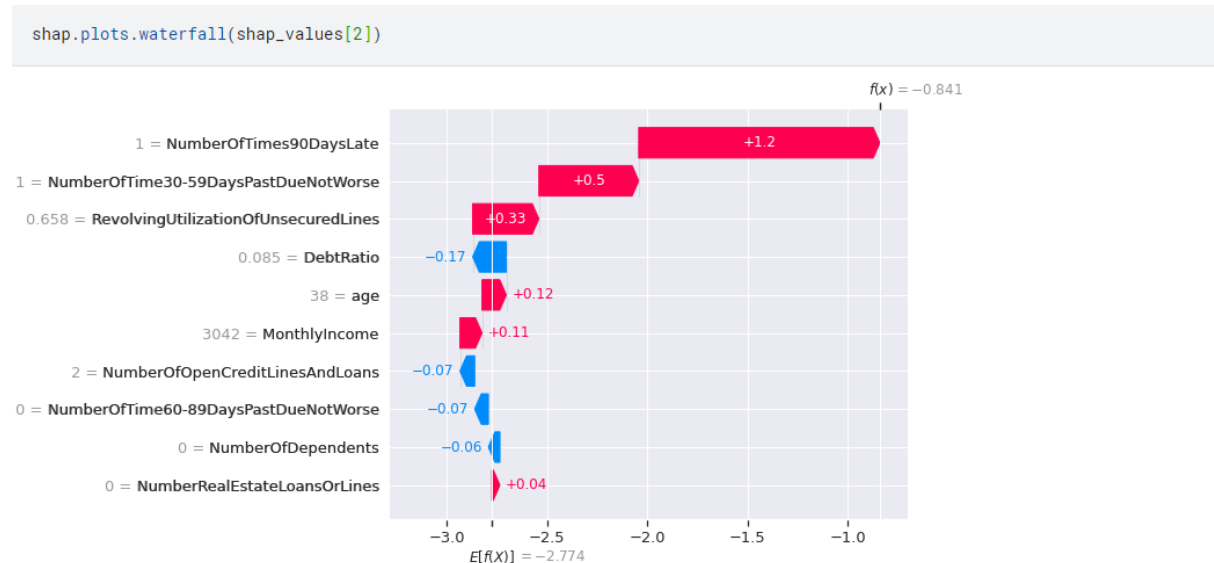
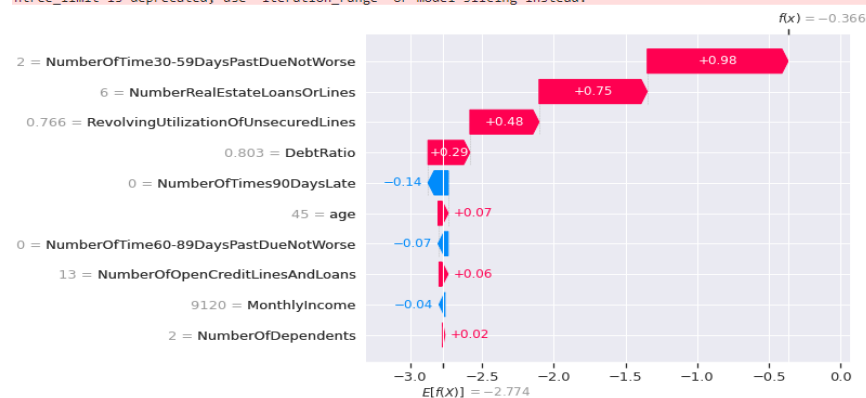
It is evident from this as xgboost cover more AUC it performs better than MLP.

d) What insight(s) do you have from your model(s)?

Ans. We used **SHAP** to find the which feature is more responsible for influencing the prediction

```
explainer = shap.Explainer(Model)
shap_values = explainer(df_train)
# visualize the first prediction's explanation
shap.plots.waterfall(shap_values[0])
```

ntree_limit is deprecated, use 'iteration_range' or model slicing instead.



The above explanation shows features each contributing to push the model output from the base value (the average model output over the training dataset we passed) to the model output. Features pushing the prediction higher are shown in red, those pushing the prediction lower are in blue.





















Following features have most effect on the prediction of model:

- **NumberOfTime30-59DaysPastDueNotWorse**
- **RevolvingUtilizationOfUnsecuredLines**
- **NumberRealEstateLoansOrLines**
- **Age**
- **Monthly Income**

From this we can infer these are important features on which we can decide whether a person will get a loan or not.

e) Can you get into the top 100 of the private leader board or even higher?

Ans. Highest accuracy achieved: **86.731%**

91	▲ 10	gnanajothy			0.86737	8	11y
92	▲ 66	AimHigh			0.86734	1	11y
93	▼ 90	Soil			0.86733	92	11y
94	▲ 8	PREDIKTIVA			0.86732	11	11y
95	▼ 80	Seyhan			0.86730	61	11y
96	▲ 36	weigeerdai			0.86730	2	11y
97	▲ 43	Matthew Roos			0.86728	25	11y
98	▼ 22	littleboy			0.86726	8	11y
99	▼ 8	tintin			0.86723	17	11y
100	▼ 66	bigboots			0.86723	7	11y

Accuracy of person at rank 95 - **86.730%**

Therefore, rank achieved - **95**