A federated learning based approach for loan defaults prediction

Geet Shingi
Dept. of Computer Engineering,
Pune Institute of Computer Technology,
Maharashtra, India.

2020 International Conference on Data Mining Workshops (ICDMW) | 978-1-7281-9012-9/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICDMW51313.2020.00057

***Abstract*—The number of defaults in bank loans have recently been increasing in the past years. However, the process of sanctioning the loan has still been done manually in many of the banking organizations. Dependency on human intervention and delay in results have been the biggest obstacles in this system. While implementing machine learning models for banking applications, the security of sensitive customer banking data has always been a crucial concern and with strong legislative rules in place, sharing of data with other organizations is not possible. Along with this, the loan dataset is highly imbalanced, there are very few samples of defaults as compared to repaid loans. Hence, these problems make the default prediction system difficult to learn the patterns of defaults and thus difficult to predict them. Previous machine learning-based approaches to automate the process have been training models on the same organization's data but in today's world, classifying the loan application on the data within the organizations is no longer sufficient and a feasible solution. In this paper, we propose a federated learning-based approach for the prediction of loan applications that are less likely to be repaid which helps in resolving the above mentioned issues by sharing the weight of the model which are aggregated at the central server. The federated system is coupled with Synthetic Minority Over-sampling Technique(SMOTE) to solve the problem of imbalanced training data. Further, The federated system is coupled with a weighted aggregation based on the number of samples and performance of a worker on his dataset to further augment the performance. The improved performance by our model on publicly available real-world data further validates the same. Flexible, aggregated models can prove to be crucial in keeping out the defaulters in loan applications.**

***Index Terms*—Federated Learning, Defaults prediction, Data privacy and security**

I. Introduction

There has been a constant rise in the number of loans issued by banks in recent times [1]. Currently, the process of sanctioning is done manually in many of the banking organizations. Dependency on human intervention and delay in results have been the biggest obstacles in this system. Such moderation increases the cost as well as the time required for completion of the objective which can be reduced by automating this process. The loans intend to provide a way for issuers to meet their exorbitant requirements. However, there are instances where the issuers are not able to repay the loan amount which affects the bank organization and their other customers. Lack of serious evaluation has been a major factor contributing to the defaults in loans.

There has been ample amount of research in recent years to automate the process of sanctioning using various supervised machine learning and deep learning algorithms [4], [5], [7]. Advanced machine learning approaches made use of the under-sampling or oversampling mechanisms [16], [17] to overcome the problem of imbalanced data. However, as they do not account for the data outside the organization these approaches are unfortunate for the banks and are not enough to limit the incidents of loan defaults. Thus, the prediction of defaults in loan applications accurately is the need of the hour.

Many constraints and challenges prevent the development of an ideal default prediction system for banks. Although many approaches have achieved good performance they fail to consider the out-house data. Thus, the existing system tends to be inefficient, with low accuracy or false alarms due to concerns like dataset insufficiency and imbalanced data distribution.

- **Dataset Insufficiency:** One of the major concerns associated with the existing default prediction system is the availability of datasets. Additionally, increasing data privacy rules impose barriers to data sharing for bank organizations. Also, many of the banking organizations develop a default prediction system based on the in-house data and also conceal the model details for data security. Thus, a reliable default prediction system is impossible to develop due to the limitation of the available dataset.
- **Imbalance Class Distribution:** Loan defaults in banks are highly imbalanced where only a few loans issued are defaulted as compared to the loan repaid. We can say that more than 95% of loan issues are repaid and less than 5% of them are defaults. A model trained on such data will fail to discover the patterns in minority class and thus fail to predict the default due to the bias present in the dataset.

In this paper, we propose the use of a federated learning approach for the task in hand and to overcome the issues discussed previously. Firstly, the Synthetic Minority Over-sampling Technique(SMOTE) [18] approach is proposed to overcome the class imbalance issue. Secondly, we focus on a default prediction system protecting data sensitivity, meanwhile, it can be shared with other banks. A decentralized machine learning algorithm-federated learning with the data balance approach is proposed which is different from the previous default prediction system. The Federated default prediction approach enables different banks to collaboratively learn a shared model while keeping all the training data on their private database. We further make use of number of samples and performance based weighted aggregation of models to improve our results. The proposed architecture is found to be superior and provides accurate results in predicting defaults on the LTFS [19] data across various metrics. Our main contributions in this paper could be listed as:

1) We have been able to deal with the default prediction problem and construct an effective distributive system.
2) We have been able to overcome the problem of imbalanced data using the oversampling approach.
3) We have been able to improve the performance using the number of samples and performance based weighted aggregation of models.

The rest of the paper is structured as follows: Section 2 talks about related work in this area while the proposed methodology is explained in section 3. Dataset description and the results obtained are discussed in section 4. Our analysis of the work conducted is presented in section 5 while the paper is concluded in section 6.

II. BACKGROUND AND RELATED WORK

Default prediction has been a much-discussed topic in the industry but the number of publicly available works are rather limited. The major reason contributing to this is the protection of data sources. Myriad approaches have been tried in the past to solve issues faced in the area of default prediction chiefly due to the rise in the use of machine learning algorithms. One of the earlier and commonly used methods was the asset-based approach by Merton [2] which was further developed by [3]. [4] was one of the first attempts to predict defaults with the help of neural networks. [5] used the indicators inspired by [2] along with neural networks for default prediction. [6] and [7] made use of Support Vector Machines(SVM) for default prediction. G. Zhang et al compared the neural network approach with logistic regression and employed a five-fold cross-validation approach on a sample of firms [8]. [9], [10], and [11] are some of the approaches based on neural networks to predict defaults. Kiviluoto et al proposed an approach using self-organizing maps on an extensive database of Finnish firms and show that it obtains comparable results to linear discriminant analysis and learning vector quantization [12]. Kvamme et al proposed a hybrid CNN + Random forest model for predicting mortgage defaults [13]. [14] combined deep learning and genetic programming to enhance the results in default prediction. Although almost all of the approaches were based on supervised learning Moise et al proposed an unsupervised learning-based system by using fuzzy logic and uncertainty coefficients [15].

The majority of the work is done considering the classes are balanced. Recently, Chen et al proposed a diversified sensitivity undersampling along with neural networks based approach to overcome the issue of class imbalance but they did not account for the data insufficiency issue [16]. Tian et al proposed an incremental ensemble approach to overcome the problem of class imbalance and also accounts for incremental data [17]. A notable observation is that all of these approaches have not taken into consideration the data insufficiency issue and have proposed centralized approaches. Xu et al proposed a federated system in the field of healthcare to overcome the limitations of centralized approach [23]. However, the relatively contemporary federated learning approach has not been used in this domain.

III. METHODOLOGY

Our task is to predict a given loan application into default or repayable. We explain our solution by describing our approach for each phase of the standard default prediction pipeline- data preprocessing, model training, and model architecture

A. Data preprocessing

The data is stripped of any non-related features as these primarily do not add to the output for accurate and reliable predictions. The quasi-constant features(having more than 99% similar values for the output features) were dropped. Further, the features which were more linearly dependent on the output features were dropped using the correlation matrix. And lastly, label encoder is applied to convert the categorical features into numerical form. The retained features dataset is standardized by using Scaler which computes the median and scales the data according to the Interquartile Range. However, mean and scaling to unit variance is performed typically but sample mean and variance can be often negatively influenced by outliers. In such cases, median and the interquartile range often give better results.

On analysis, it was found that the data is imbalanced and there was a class imbalance problem. To reduce the variance in the training data we used an oversampling approach known as Synthetic Minority OverSampling Approach(SMOTE) [18]. The algorithm operates by joining all the nearest neighbours of the k minority class for every minority class sample. The class imbalance problem can be resolved by either undersampling or oversampling. In 2009, Tang et al. presented a system based on the undersampling of the majority class [20]. However, due to undersampling crucial data may be lost. Therefore, to overcome the problem of class imbalance we synthetically oversampled minority class samples in training data. Initially, the data had a ratio of 30:1 for number of majority class to minority class samples and after the synthetic data was generated, the ratio achieved was 1:1.

B. Model Training

A common architecture is used in the case of all the workers. We propose the use of a 4 layer neural network as the principal component of all the workers. By making use of the neural network, the worker can train a model based on his private data whose weights contribute to developing a

Algorithm 1: SMOTE Algorithm

```
Input: Number of minority class samples Y; Number
       of majority class samples Z; Number of nearest
       neighbors k
Output: Z minority class samples
1 for $i$ = $1$ to $Y$ do
2     Compute k nearest neighbors for i
3     while $Y$ != $Z$ do
4         Randomly chose one of the k nearest neighbor
            of i in feature space, say jj
5         Compute vector between jj and i and multiply
            it with a random number between 0 and 1
6         Synthetic sample = i + computed vector
7     end
8 end
9 Return $Y$
```

shared global model. The model architecture is as shown in fig 1.

1) Loss functions:
In order to learn from the data present each model has a loss function defined on its weights w for each data sample. The loss function calculates the error of the default prediction model on training data. The loss for a data sample (x_i, y_i) with model weights w is defined as $\ell(\mathrm{x}_i,\ y_i;\ \mathrm{w})$. As the task in hand is a binary classification we make use of the binary cross-entropy loss.

$$\ell_p(x) = \frac{-1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1-y_i)\cdot \log(1-p(y_i)) \tag{1}$$

where y is the label (1 for default and 0 for repaid) and p(y) is the predicted probability of the sample being 1(default) for all N samples.

2) Activation functions:

Activations functions are used to introduce nonlinear properties to the model. In our architecture, we have used two activation functions: ReLu and Sigmoid. ReLu is used after every input and hidden layer whereas the sigmoid is used in the final output layer to give us the probability.

C. Model Architecture

We propose the use of a federated architecture [21] as the principal mechanism of the model architecture. By making use of the federated approach, the data insufficiency problem is completely eliminated and the communication latency is reduced. Unlike traditional machine learning, federated learning enables us to collaboratively learn a global shared model. With the help of data present at local workers a shared global model is trained under the coordination of central server [22]. In federated learning, each worker is a node which performs computation in order to update the global shared model present at the central server. Every worker trains their model by using the local training data which is never uploaded to the central server, but the update of the model will be communicated. Initially, a common shared global model is shared with every worker and this model is trained by every worker on their respective data. Further, the updated local model are shared with the central server to update the global model by weighted aggregation of the weights of local models shared and the global model is shared with the local models and this goes on until T iterations. The structure of federated learning is illustrated in fig 2. The working of each component is as follows:

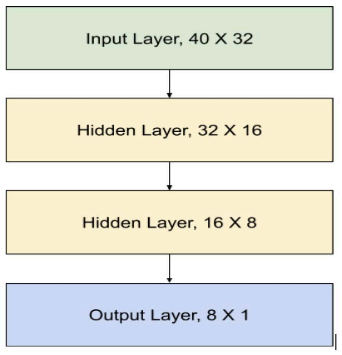


Fig. 1: Block diagram of common architecture model

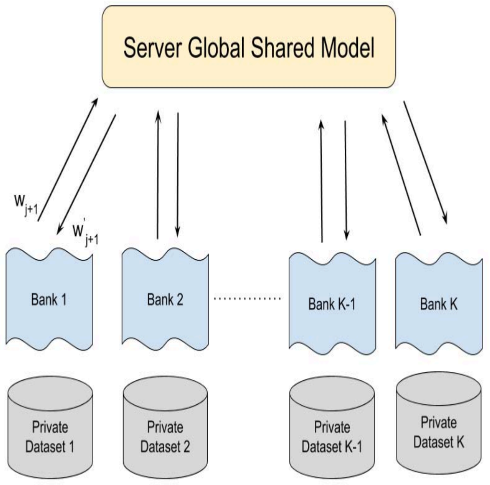


Fig. 2: Block diagram of the federated learning framework. w_{j+1} represents the banks weights that upload to the server, w_{j+1} represents the weights that are aggregated by server.

1) Private Dataset($D_1, D_2, \ldots, D_K$) :

With a fixed set of K banks(or banking organizations) as participants/workers and each worker possesses a fixed private dataset $D_i = \{x_i^k, y_i^k\}$ and $k = \{1, 2, 3, ..., K\}$. Here, x_i^k is the feature vector and y_i^k is the corresponding label and n_k is the size of the dataset associated with the worker k. To handle the class imbalance issue, SMOTE [18] method is selected for data balance at the local worker level for data D_i. SMOTE is easy to implement and does not lead to increased training time or resources.

2) Local models($Bank_1, Bank_2, \ldots, Bank_K$) :

There is a fixed set of K banks(or banking organizations) as participants/workers. Each worker(i) trains a model on the private data(D_i) available. The objective of the local neural network model is to minimize the loss i.e (1). But the local private data D_i is not substantial for the local model to perform the best. Therefore, to learn a better model one has to take into account the data from other workers as well.

3) Global model:

The global model is present at the central server and the central server communicates with all the workers. All the workers send their local model weights to the central server and a weighted aggregation of all local model weights takes place at the central server to create a shared global model. As the data is not present at the central server, the objective of the global model is:

$$\begin{aligned} \min_{w \in \mathbb{R}^d} \ell(w) \; where \; \ell(w) = \sum_{k=1}^{K} \frac{n_k}{n} \times F_k(w) \\ with \; F_k(w) = \frac{1}{n_k} \sum_{i=1}^{n_k} \ell_i(w) \; and, \\ \ell_i(w) \; is \; defined \; in \; (1) \end{aligned} \tag{2}$$

4) Aggregation:

At each communication round j = 1,2,..., all workers will be selected and these workers will communicate directly with the server. Initially, every model is initialized with the global model weights from the central server and the weights of the global model initially are random. Then, for every worker k which computes the f_k(the average gradient of the loss) on their respective private data with w_j(current model weights) with η(fixed learning rate) the weights of local models are updated as:

$$w_{j+1} = w_j - \eta F_k(w_j) \tag{3}$$

These workers update their models parallely and send the updated model to the central server. The central server then aggregates these updates and improves the default prediction shared model and this whole process goes on till J iterations.

FedAvg(Federated Averaging) algorithm is the generic algorithm used when more than one update is performed locally for each user. FedAvg aggregates all the local workers by averaging(mean) of the weights to build a global model. But averaging(mean) the weights of local workers is somewhat not feasible where a local worker(k) with very less training examples(n_k) and less performance contributes equally as other local workers. The FedAvg algorithm for updating weights of the global model(w_j) as follows:

$$w_{j+1} = \sum_{k=1}^{K} \frac{1}{K} w_{j+1}^k \tag{4}$$

To overcome the inefficiencies caused by the FedAvg algorithm in heterogeneous data distribution we have proposed an aggregating method based on the number of data samples and performance of the model. ROC score(α) is used as a factor to calculate the performance of the model. The algorithm for updating weights of the global model as follows:

$$w_{j+1} = \sum_{k=1}^{K} \frac{n_k}{n} \alpha_{j+1}^k w_{j+1}^k \tag{5}$$

We perform a weighted aggregation of the local models as increasing the weight of strong classifiers enables the formation of a better global shared model.

Algorithm 2: Federated learning model based on number of data samples and performance based weighted aggregation algorithm. The banks(K) indexed by n, local minibatch size(B), number of local epochs(E), and learning rate(η)

```
   Input: The private datasets of workers
   Output: A default prediction model
1  initialize w_0
2  for each round j =0,1,... do
3      for each client k ∈ $N_j$ in parallel do
4          $w_{j+1}^k, \alpha_{j+1}^k \leftarrow ClientUpdate(n, w_j)$
5      end
6      $w_{j+1} = \sum_{k=1}^{K} \frac{n_k}{n} \alpha_{j+1}^k w_{j+1}^n$
7  end
8  Function ClientUpdate($n, w$):
9      B ← split $D_n$ into baches of size B
10     for each local epoch i from 1 to E do
11         for batch b ∈ B do
12             w ← $w - \eta \nabla \ell(x, y; w)$
13         end
14     end
15     Return w and validation roc score $\alpha$ to server
16 end
```

IV. Dataset Description and Results

The proposed architecture is evaluated on the publicly available dataset on Kaggle released as a part of the LTFS Data Science Finhack ML Hackathon(2019) [19].

A. Dataset Description

The dataset consists of 40 features and 2 output labels namely repaid and defaulted. The dataset contains 6 categorical and 34 numerical data features. The dataset consists of 155754 data samples and the data is highly imbalanced as we can see in fig 3. Some of the dataset columns are UniqueID, disbursed amount(amount of loan disbursed), asset cost(cost of the asset), ltv(loan to value of asset), PERFORM CNS SCORE(bureau score), PRI NO OF ACCTS(count of total loans taken by the customer at the time of disbursement), PRI ACTIVE ACCTS(count of active loans taken by the customer at the time of disbursement), PRI OVERDUE ACCTS(count of default accounts at the time of disbursement), PRI CURRENT.BALANCE(total principal outstanding amount of the active loans at the time of disbursement), PRI SANCTIONED AMOUNT(total amount that was sanctioned for all the loans at the time of disbursement), etc. Further, we distributed the data heterogeneously among workers and applied a train-test split of 80% and 20% on each worker. We train our system with 2, 3, and 4 workers and the data is distributed as shown in table 1

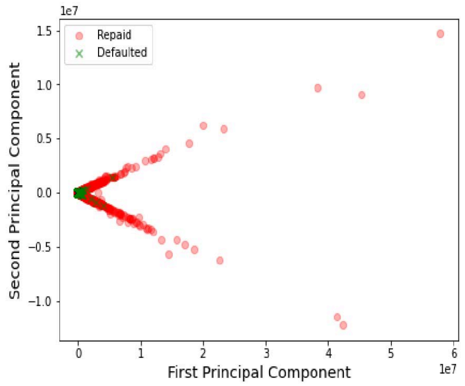


Fig. 3: Dataset visualization via PCA(1 = Defaulted and 0 = Repaid)

TABLE I: Data distribution among workers

No. of workers	Worker	Training Data	Testing Data
n = 2	Bank1	93452	23363
	Bank2	31151	7788
n = 3	Bank1	70960	17741
	Bank2	34952	8738
	Bank3	18690	4673
n = 4	Bank1	52956	13239
	Bank2	39717	9930
	Bank3	18690	4673
	Bank4	13239	3310

Each worker applies SMOTE to their private training data to overcome the concern of overfitting. The data distribution before SMOTE and the distribution achieved after SMOTE is shown in table 2. The local workers train their models on their respective data and are further aggregated at the central server and the process goes on.

TABLE II: Data distribution before and after SMOTE

No. of workers	Worker	Output	Before SMOTE	After SMOTE
n = 2	Bank1	Repaid	90231	90231
		Defaulted	3221	90231
	Bank2	Repaid	30003	30003
		Defaulted	1071	30003
n = 3	Bank1	Repaid	68253	68253
		Defaulted	2437	68253
	Bank2	Repaid	33746	33746
		Defaulted	1206	33746
	Bank3	Repaid	18046	18046
		Defaulted	644	18046
n = 4	Bank1	Repaid	51130	51130
		Defaulted	1826	51130
	Bank2	Repaid	38347	38347
		Defaulted	1370	38347
	Bank3	Repaid	18069	18069
		Defaulted	621	18069
	Bank4	Repaid	12783	12783
		Defaulted	456	12783

B. Performance Metric

As the majority of data is imbalanced, accuracy can often give a false inference by being biased towards majority class and thus cannot be an appropriate metric for the task. We thus resort to other performance metrics such as ROC score, precision, recall and F1 score.

Receiver Operating Characteristics(ROC) curve maps the true positive rate to the false positive rate and area under the same indicates the probability of model rating a random positive instance higher than a random negative instance. The formula for the same is given as:

$$ROC\ Score = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

Precision refers to how precisely a model categorizes a given sample into a particular class. We calculate the precision as:

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

Recall indicates how many of a particular class of samples was the model able to predict. We calculate the average recall as:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

The F1 score is the harmonic mean of precision and recall. The average F1 score is calculated as:

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{9}$$

In above equations,

- TP: True positive(outcome where the model correctly predicts the positive class)
- FP: False positive(outcome where the model incorrectly predicts the positive class)
- TN: True negative(outcome where the model correctly predicts the negative class)
- FN: False negative(outcome where the model incorrectly predicts the negative class)

Every local model was trained on a standard dual-core CPU. The trained model was then evaluated over each of the above-mentioned metrics.

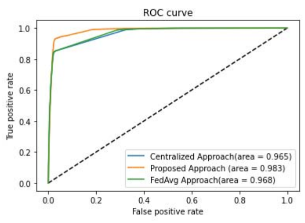
C. *Results obtained*

The results of the model on the test set are evaluated on each of the metrics. We focus more on the ROC score as it provides a better intuition of performance in case of class imbalance classification. The same metric has been used in the challenge as well. The implementation details for the comparison of the results between the proposed and centralized approach for each worker is as follows:

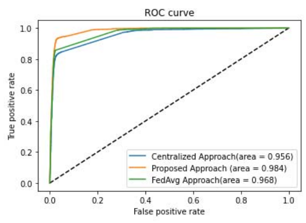
- **Proposed Approach:** Epochs: 50, Batch size: 64, Learning rate: 0.01, Aggregation steps: 8
- **Centralized Approach:** Epochs: 400, Batch size: 64, Learning rate: 0.01

Each worker is trained for same number of epochs but in the proposed approach the 8 aggregation steps takes place after every 50 epochs and the centralized is trained for 400 epochs straight. The results obtained when 2 workers(banks local models) through centralized and federated approaches are shown in fig 4.

It can be seen that the number of samples and performance-based averaging approach models outweigh the centralized approach models and FedAvg algorithm based averaging. Further, we compare results obtained by respective workers in various approaches across accuracy, precision, recall and f1 score metrics as tabulated in table 3.



(a) Bank1



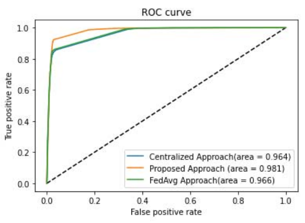
(b) Bank2

Fig. 4: ROC Curve for n = 2

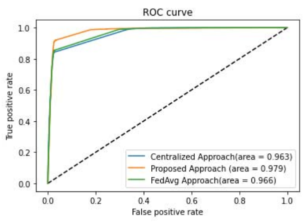
TABLE III: Comparison across multiple metrics(n = 2)

Approach	Worker	Accuracy	Precision	Recall	F1 Score
Centralised	Bank1	93.27	0.93	0.85	0.88
	Bank2	92.09	0.9	0.83	0.86
FedAvg	Bank1	93.41	0.94	0.84	0.88
	Bank2	93.75	0.94	0.85	0.89
Proposed	Bank1	**94.77**	**0.95**	**0.94**	**0.91**
	Bank2	**94.79**	**0.95**	**0.94**	**0.91**

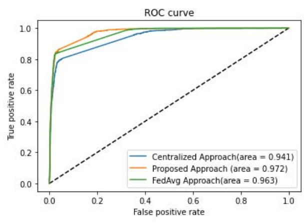
It can be seen that our proposed approach has outperformed other approaches on the majority of the metrics in the case of two workers. In the next tables and figures, we compare our results for three and four workers.



(a) Bank 1



(b) Bank 2

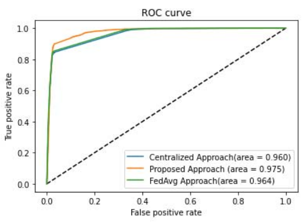


(c) Bank 3

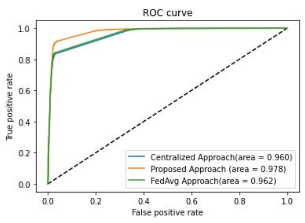
Fig. 5: ROC Curve for n = 3

TABLE IV: Comparison across multiple metrics(n = 3)

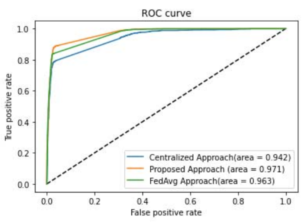
Approach	Worker	Accuracy	Precision	Recall	F1 Score
Centralised	Bank1	93.16	0.93	0.84	0.88
	Bank2	92.75	0.91	0.85	0.87
	Bank3	90.67	0.90	0.78	0.83
FedAvg	Bank1	93.27	0.93	0.85	0.88
	Bank2	93.3	0.93	0.85	0.88
	Bank3	92.6	0.91	0.83	0.86
Proposed	Bank1	**93.49**	**0.94**	**0.86**	**0.87**
	Bank2	**93.72**	**0.94**	**0.87**	**0.90**
	Bank3	**93.05**	**0.92**	**0.84**	**0.89**



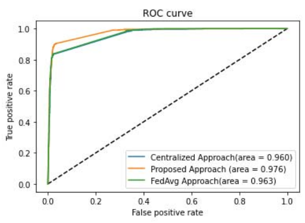
(a) Bank 1



(b) Bank 2



(c) Bank 3



(d) Bank 4

Fig. 6: ROC Curve for n = 4

TABLE V: Comparison across multiple metrics(n = 4)

Approach	Worker	Accuracy	Precision	Recall	F1 Score
Centralised	Bank1	92.91	0.92	0.85	0.88
	Bank2	92.34	0.92	0.84	0.87
	Bank3	91.57	0.91	0.84	0.87
	Bank4	90.28	0.92	0.82	0.85
FedAvg	Bank1	93.27	0.94	0.84	0.88
	Bank2	93.07	0.94	0.83	0.88
	Bank3	92.93	0.93	0.82	0.88
	Bank4	92.65	**0.93**	0.82	0.87
Proposed	Bank1	**94.81**	**0.95**	**0.91**	**0.91**
	Bank2	**94.62**	**0.95**	**0.89**	**0.91**
	Bank3	**94.55**	**0.94**	**0.89**	**0.91**
	Bank4	**93.90**	0.92	**0.88**	**0.89**

V. Analysis

Our main observations after performing out the study have been:

- Through the use of federated approach we can observe that the performance is better than the centralised approach for each worker while preserving the privacy of the data.
- Through the FedAvg approach we can obtain performance close to the performance of the local worker with highest performance in a centralized approach but through the proposed approach based on the number of samples and performance we can further increase the performance as that of the FedAvg approach.
- By increasing the number of workers, the performance of local workers is not compromised and the proposed approach outperforms other approaches.
- If all the data is to be present at one location, the model will tend to give better performance than the proposed architecture but having all data is not possible in every case and thus federated learning can be exploited in this case to develop a better model as compared to the centralized local models.

VI. Conclusion

Thus by making use of the federated approach, we have been able to achieve improved results in the area of default prediction. The Federated approach enables banks without sending their private data to a central server to train a default prediction system. This decentralized system can protect security, sensitivity, and alleviate the influence of unavailable data. While the generic FedAvg algorithm is found to be useful our proposed aggregation algorithm further improves the performance significantly. The improved results across various metrics are a testament to the power which the federated approach holds in sensitive data training. Further improvement in our work includes modifications for handling heterogeneous data to increase the scalability of the system. There are still minor privacy problems in federated default prediction systems which we can overcome by using a neutral third party aggregation for the system. Strong legislative rules have already begun limiting the centralized approaches for private-sensitive data and the idea of using centralized approaches has started becoming weak day by day. Federated systems have proven to be a better alternative and can be put to use in areas where the security of data is of prime importance.

References

[1] "Loan Growth Statistics", [Online] Available: https://tradingeconomics.com/india/loan-growth

[2] R. Merton, "On the pricing of corporate debt: The risk structure of interest rates", J. Finance, vol. 29, pp. 449-470, 1974.

[3] F. Longstaff, E. Schwartz, "A simple approach to valuing risky fixed and floating rate debt", J. Finance, vol. 50, pp. 789-819, 1995.

[4] Odom, Marcus & Sharda, Ramesh. (1990). A Neural Network Model for Bankruptcy Prediction. IEEE International Joint conference on Neural Networks. 2. 163 - 168 vol.2. 10.1109/IJCNN.1990.137710.

[5] A. F. Atiya, "Bankruptcy prediction for credit risk using neural networks: A survey and new results," in IEEE Transactions on Neural Networks, vol. 12, no. 4, pp. 929-935, July 2001.

[6] Hong Sik Kim, So Young Sohn, Support vector machines for default prediction of SMEs based on technology credit, European Journal of Operational Research, Volume 201, Issue 3, 2010, Pages 838-846, ISSN 0377-2217.

[7] Moula, F.E., Guotai, C. & Abedin, M.Z. Credit default prediction modeling: an application of support vector machine. Risk Manag 19, 158–187 (2017).

[8] G. Zhang, M. Hu, B. Patuwo, "Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis", European J. Oper. Res., vol. 116, pp. 16-32, 1999.

[9] P. Brockett, W. Cooper, L. Golden, U. Pitaktong, "A neural network model for obtaining an early warning for insurance insolvency", J. Risk and Insurance, vol. 61, pp. 402-424, 1994.

[10] T. Bell, G. Ribar, J. Verchio, "Neural nets vs. logistic regression: A comparison of each model's ability to predict commercial bank failures", Proc. 1990 Deloitte Touche/Univ. Kansas Symp. on Auditing Problems, pp. 29-53, 1990.

[11] R. Barniv, A. Agarwal, R. Leach, "Predicting the outcome following bankruptcy filing: A three-state classification using neural networks", Intelligent Syst. in Accounting Finance and Manag., vol. 6, 1997.

[12] K. Kiviluoto, "Predicting bankruptcies with the self-organizing map", Neurocomputing, vol. 21, pp. 191-201, 1998.

[13] Håvard Kvamme, Nikolai Sellereite, Kjersti Aas, Steffen Sjursen, Predicting mortgage default using convolutional neural networks, Expert Systems with Applications, Volume 102, 2018, Pages207-217, ISSN 0957-4174

[14] K. Tran, T. Duong and Q. Ho, "Credit scoring model: A combination of genetic programming and deep learning," 2016 Future Technologies Conference (FTC), San Francisco, CA, 2016, pp. 145-149.

[15] Moise, Adrian. (2014). Unsupervised Learning Based Neural System for Evaluating the Credit Risks. 1.

[16] Chen, Ya-Qi & Zhang, Jianjun & Ng, Wing. (2018). Loan Default Prediction Using Diversified Sensitivity Undersampling. 240-245. 10.1109/ICMLC.2018.8526936.

[17] J. Tian, X. Liu and M. Li, "An Incremental Learning Ensemble Method for Imbalanced Credit Scoring," 2019 IEEE Symposium Series on Computational Intelligence (SSCI), Xiamen, China, 2019, pp. 754-759.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique", Journal of Artificial Intelligence Research, vol. 16, no. 1, pp. 321-357, 2011.

[19] "LTFS Data Science FinHack (ML Hackathon) 2019 Leaderboard", [Online] Available: https://datahack.analyticsvidhya.com/contest/ltfs-datascience-finhack-an-online-hackathon/#LeaderBoard

[20] Y. Tang, Y. Zhang, N. V. Chawla and S. Krasser, "SVMs Modeling for Highly Imbalanced Classification," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 39, no. 1, pp. 281-288, Feb. 2009.

[21] McMahan, B., Ramage, D.: Federated learning: Collaborative machine learning without centralized training data. Google Research Blog (2017)

[22] Jakub Konecný and H. Brendan McMahan and Felix X. Yu and Peter Richtárik and Ananda Theertha Suresh and Dave Bacon, "Federated Learning: Strategies for Improving Communication Efficiency, arXiv:1610.05492 (2016)

[23] Jie Xu and Fei Wang, "Federated Learning for Healthcare Informatics", arXiv:1911.06270 (2019)