MiSA - A System for a Microlending Service to Assist Edge Communities

Yash Mahajan
Virginia Tech
Blacksburg, Virginia, USA
yashmahajan@vt.edu

Dilip Krishnaswamy
Reliance Industries Ltd
Fellow (C4IR), World Economic Forum
Navi Mumbai, Maharastra, India
dilip@ieee.org

Pethuru Raj Chelliah
Reliance Industries Ltd
Bangalore, Karnataka, India
peterindia@gmail.com

***Abstract*—In this paper, we propose a distributed edge+cloud system to assist with microlending services to communities, with machine learning catered to that specific community. A combination of technologies including microservices-based architecture and blockchain technology coupled with machine learning is utilized to provide microfinancing services to help sustain businesses in a local community, and to enable the community to grow into a thriving economy. To minimize the widespread expressed risk, in our prototype, the prediction of whether a loan will default or not is based on the various decision-enabling parameters and on any available information about the borrowers' past transaction as well as aggregate metrics related to the community that the borrower resides in. The authors hope that the suggested distributed edge+cloud architecture in the paper can be leveraged for other emerging sustainable edge applications as well.**

***Index Terms*—Microfinance, Distributed Machine Learning, Cloud Computing, Machine learning, Containerization, Blockchain, Edge Learning, Service Orchestration, Edge-Enabled Application**

I. Introduction

In the past, most edge communities have functioned in an isolated manner, and microfinancing services have been provided but with minimal support from a technology perspective. In this paper, we utilize new technologies to assist with microfinancing to better serve such communities. In particular, machine-learning is performed with respect to data available from a community to learn and provide better services to such a community. In addition, with global connectivity and cloud-based services, the learned model across different communities can be aggregated to leverage information across communities. Blockchain-based technology is utilized to provide a secure tamper-proof trusted platform for sharing and utilizing learned information as well. In addition, the micro-services based paradigm is utilized to combine different technologies such as to execute a microservice for machine learning based on available data, or to execute a microservice to predict the risk associated with a loan, or to invoke a microservice to interact with a blockchain system, or to invoke a edge microservice to interact with a remote cloud service for distributed edge+cloud processing.

Microfinancing enables borrowers to get credits in time from lenders and embark on a specific activity for their livelihood. Micofinance not only provides income to those who need it, but also creates real jobs and creates a possibility for future investments. Thus the ability to support microfinance for edge and remote communities can help in sustaining such communities by providing financial support for the needs of such communities and can also help with in the economic development and growth of such communities. So it is useful to create an edge model to better serve a specific community by learning that community better, and such an edge model can be hosted at either a remote cloud server or an edge server (if the latter is available), as suggested in the paper.

II. Literature Survey

Abhishek et al. [7], in Decentralized Edge Clouds, gives insight into the benefits and tradeoff of distributed and data intensive computing application using distributed and decentralized clouds. Lopez et al. [8] introduces the concepts of edge computing and the need for decentralization. Byanjankar et al.[1], uses a neural network model to predict P2P credit ratings, for screening applications and classify the model based on default and non-default ratings, using financial variables. The credit risk associated with P2P loan credits is high, and this has been supported by Kumar et al.[2], by using decision trees, random forest and bagging, using the parameters accuracy and precision. A model is proposed for blockchain based on IPFS in the paper by Benet [4], where the security concerns, storage issues and download speed are synchronized. The paper proposed by Nalic et al.[3], uses actual datasets from microfinance centres in Bosnia and Herzegovina. The preprocessing and filtering of the data involves selecting certain parameters, such as the credit history, the date of loan credit and current credit request. The proposed algorithm provides the performance matrix for logistic regression method used in this paper, with high accuracy. In this paper, we compare three different models - Logistic Regression, Neural Network and Light Gradient Boosting Machine and combine the result with our different rating approaches which is then provided as a service to be used by anyone and anywhere.

III. The Proposed MiSA System

The proposed MiSA system (Fig. 1) is comprised of a Microservices-based architecture to provide support for a machine learning-based lending service coupled with blockchain and IPFS capabilities.

978-1-7281-3108-5/20/$31.00 ©2020 IEEE

A. Machine Learning

We implement three approaches for calculating the credit rating of the borrower and the data used for these approaches are different :

- Directly based on the borrower's loan details: Credit grading is assigned based on the current loan parameters using Machine Learning.
- Based on the borrower's past transactions: Borrower's past transactions are taken into account for calculating credit grading.
- Combining both the above approaches: We combine both the approaches mentioned above and assign weights dynamically to arrive at an holistic credit rating to help the lender make a assured investment.

To implement the machine learning models, we make use of the Tensorflow library [10] by Google. Tensorflow is an open source software library, which makes machine learning faster and easier.

1) Data Definition:
Data for the study and implementation has been retrieved from the publicly available data set provided by Lending Club containing a total of 887379 entries with 74 different parameters from the year 2007-2016. The data set can be used and downloaded by anyone from [9] which is a link for accessing the data set. Furthermore, the data set has been worked on and modified to accurately represent borrower data, which is not publicly available due to privacy concerns. Lending club is an online platform which offers peer-to-peer lending services to both personal and business ventures and is in fact world's largest P2P lending platform. The target variable in our dataset is 'Loan Status 'which has the following six values -

- Default: Borrower was unable to repay and hence the loan was defaulted. Value = 1
- Charged - off: There is no reasonable expectation of further payment by the borrower. Value = 2
- Fully paid: The amount of the loan taken has been fully repaid. Value = 5
- Current: The loan is up to date on all outstanding payments. Value = 0
- Issued: The loan has passed all initial checks and has been funded by an investor. Value = 0
- In Grace Period: Loan is past due date and within the 15-day grace period. Value = 4
- Late (16-30 days): Loan has not been current for 16 to 30 days. Value = 3
- Late (31-120 days): Loan has not been current for 31 to 120 days. Value = 2

2) Data Set Preparation and Feature Selection:
Since the data was imbalanced, huge and had multiple features with multiple values and different data types, feature engineering was an essential step for building an unbiased predictor. Using feature engineering, relevant features were selected which have a high positive correlation with the target variable, thus increasing the predictive power of the intelligent system.

We first converted our target variable which was a string feature into numeric values by categorizing and assigning weights to the values (as mentioned while defining Loan Status). Next, we dropped all the columns (features) in the dataset with more than 70% missing data due to the lack of information available in those columns. We then manually removed all the columns which had no relation to our target variable, features such as 'id', 'member_id', 'zip_code', 'url', 'policy_code'. Several features were then again removed, including featues which were continuously updated or features that were valid only once the loan has defaulted or charged off, thereby creating an unnecessary bias for the classifier. From the remaining dataset, we located all the features with 'object'data-type (which are the categorical features) and one-hot encoded them. After one-hot encoding, for each feature there were 0 to N-1 columns (N columns in total), where N is the number of distinct values in the column. This exponentially increased the number of columns in our dataset. To reduce the number of columns, we calculated the feature importance using the Extra Trees Classifier and we set the threshold value to 0.015 which meant all the features with feature importance less than 0.015 were dropped. Moreover, all the missing values in any column will be filled with the mean of the columns. The result of this dataset cleaning and feature selection was that we are left with 887379 entries and 15 features excluding our target variable.

Since we had a class imbalance in this problem (more number of completed loans than default), for splitting and validating the data we used K-Fold Cross Validation with repetition wherein K=3 and Shuffle = True along with normal train-test split. As a result, we got a training and testing set with adequate representation of all the classes. So finally, we have training set of 544482 loan entries and testing set of 272240 loan entries.

3) Method 1: Rating Based on Current Loan Parameters:
In this approach we utilized the loan parameters as input and applied machine learning to predict the chances of the loan defaulting and provide the loan a grading from 0 to 5, to help the lender in making an investment. We provided three machine learning models as a service - Logistic Regression, Light GBM and Neural Network, which were used in predicting the probability of the loan defaulting. The above obtained training and testing dataset was used for training the models and predicting the accuracy and the performance of the model. In the next section we detail our model and then compare their performance in result section later.

1) Logistic Regression
   Logistic regression model is a linear model, in that the logic transformed prediction probability is a linear function of the target variable values. The input data (X) are combined linearly using the coefficient values to predict the target variable (Y) which loan status = default in our case. We implement logistic regression using the sklearn.linear_models package. Our model uses $C = 0.0001$, where C is the Inverse of Regularization

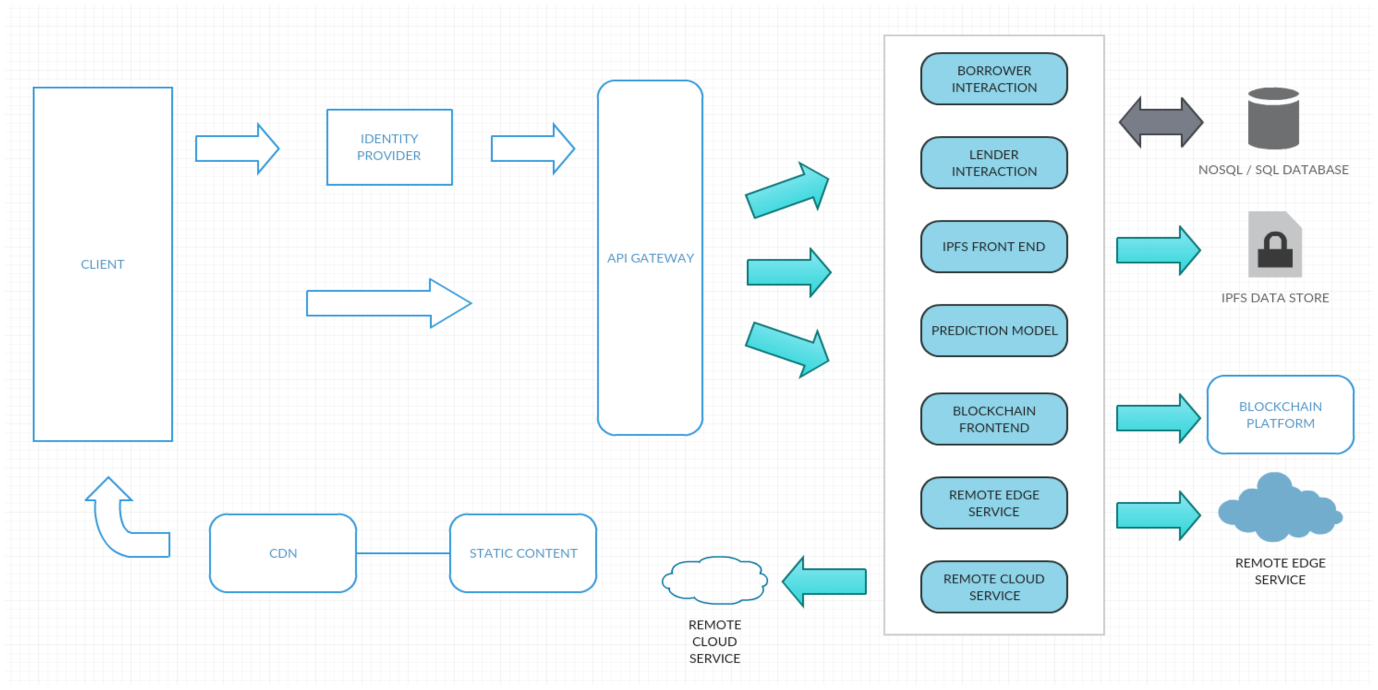

Fig. 1. MiSA System Architecture Diagram

Strength, $Tolerance = 1e{-}4$, with a $LibLinearSolver$ and $ClassWeight = none$.

2) Neural Network
The basic idea behind a neural network is to simulate a lot of densely connected neurons (brain cells) artificially so that one can get it to learn things, draw inferences, make patterns and take an informed decision in a humanlike way. We used the TensorFlow library for modelling our neural network. The neural network has a single input layer with 15 neurons corresponding to the number of independent variables in the dataset, input dimensions as 15 and activation function = sigmoid, whereas our output layer is also single layers consisting of 1 neuron with softmax activation function wherein there is one node per class label. Finally, there are two hidden layers of made up of 10 neurons each, since it is a general consensus to keep the size of the hidden layer in between the size of the output layer and the input layer, with a sigmoid activation function.

3) Light Gradient Boosting Machine
Light GBM is a fast, distributed as well as high performance gradient boosting machine, developed by Microsoft that makes use of a learning algorithm that is tree based. LGBM is known for its higher efficiency, less usage memory, higher accuracy as well as faster training speed. We used the API officially provided by LightGBM.
Light Gradient Boosting Machine uses $n\_estimator = 1000$, $learningrate = 0.02$, $max\_depth = 8$, $reg\_alpha = 0.04$, $nthreads = 4$, $min\_child\_weight = 40$ and $min\_split\_gain = 0.0222415$

The resulting output of all the models is a probability between 0 to 1, of the loan defaulting which is then converted to a rating from 0-5 by using the given formula -

$$Rating = Round((1 - Probability) * 5) \qquad (1)$$

So if the probability of the loan defaulting is 0.8 then the rating of the borrower loan will be 1. Similarly if the chances of the the loan defaulting is 0.2, then the rating of the borrower loan will be 4.

4) Method 2: Rating Based on All Past Transactions of a Borrower:
In this approach, we made a more informed decision on the grading of the borrower and instead of rating the borrower's loan, we rate the borrower itself, based on all his/her past transaction. For this purpose, we clubbed the data set randomly, assigning dispersed number of loan entries to different borrowers to test our approach. It is possible that a borrower may have multiple ongoing loans and hence in our credit rating calculation, we ignore those 'Current' and 'Issued' loans and only consider the 'Fully Paid' , 'Defaulted' , 'Charged off' and all 'Late' loans and assign weights to according to Data Definition section.

For calculating the rating for the borrower, we used our own mathematical model which gives the amount repaid and the rate of interest of a particular loan the most importance. The model normalizes the summation of all the loan entries of a particular borrower to get a value between 0 and 5.The summation of rating for all the individual loan is :

$$R = \sum (W * (\frac{A_{Repaid}}{A_{Repaid} + A_{Outstanding}}) + \frac{IR}{MIR}) \qquad (2)$$

Where W is the weight of the loan from Section 3.1.1, IR is the Interest Rate of the individual loan, MIR is the mean of all the interest rates in the data set, A_{Repaid} is the amount paid back in full at the time of credit rating calculation and $A_{Outstanding}$ is the total loan amount minus the Amount Repaid.

The Aggregated and Normalized Rating for the borrower is:

$$FinalRating = Round((\frac{\sum Rating}{6 * N}) * 5) \qquad (3)$$

Where N is the number of valid loan entries and Round function rounds the value to the nearest integer.

5) Method 3: Rating Based on Current Loan Parameters as well as Past Transactions of a Borrower:
In our third approach, we took a more comprehensive approach wherein we include both the approaches mentioned above. Both the approaches are assigned weights according to their importance in calculating the credit of the borrower. This way the lenders have a complete picture of the borrowers and their capability of repaying the loan listed by them. The lender can then weigh the risks and benefit to make a decision regarding the loan funding. In this approach we assigned dynamic weights to the borrower's past transaction and its effect on the rating. The borrower's current loan parameters and its likeness is given a slightly lesser weightage. Here, the weight assigned to both approaches depends on the borrower's past transactions and their capability. If the borrower is a frequent user, more weightage was given to the past transactions as it is best reflective of his/her repaying capabilities. Based on the number of past transactions up to a threshold $\delta = 100$, weights are assigned, above which past transactions are assigned a weight of 0.9. Once rating is obtained by using both the approaches, we combine then using the following formula:

$$Rating\ (R) = W_1 * R_1 + W_2 * R_2 \qquad (4)$$

Where R_1 and R_2 are the ratings from Method 1 and Method 2 respectively.

B. Distributed Machine Learning on the Edge

Emerging network infrastructure with 5G Networks is expected to be supported by distributed virtualized infrastructure between the cloud and the edge. Edge communities such as in villages or semi-urban areas can be well supported by edge network functions, services, and applications. Also, the expected behavior of borrowers could vary based on the edge community being serviced at an edge node. In this regard, distributed information processing between the cloud and the edge could be desirable for processing a microfinancing request. A grading based on a generic anonymized model, which consists of all the transactions in the network, stored in the cloud can be combined with a grading based on past transactions for a specific borrower at the edge to determine an overall grading for the borrower at the edge. Thus, distributed processing with combined execution of microservices at the cloud and the edge could be utilized to determine an overall grading for a borrower in a community of users being serviced at the edge. It should be noted that one could use a value of modified version α of given by α'=$\gamma * \alpha$ where $\gamma \in$ [0,1], so that some weightage to the grading obtained from the cloud is always utilized (when $\gamma \in$ [0,1)) even if α = 1, in the weighted grade estimation. In this case,

$$g\ = (1 - \alpha^{'}) * g_1 + \alpha * g_2 \qquad (5)$$

It should be noted that in this paper, only $\gamma = 1$ was utilized. Further optimizations could be possible, where a generic grading estimate g_1 across all users across the entire network is used (method 1), combined with a grading estimate g_2 based on past transactions of the specific borrower (method 2), further coupled with an edge-specific grading estimate g_3 for the group of users in the edge community (not implemented in the paper). The overall grade estimate g can then take the form

$$g\ = (1 - \alpha - \beta) * g_1 + \alpha * g_2 + \beta * g_3 \qquad (6)$$

where $\alpha \in$ [0,1], $\beta \in$ [0,1], $\alpha + \beta \in$ [0,1]. Thus distributed intelligence across a distributed network can be utilized to create an edge specific estimate for a borrower in an edge community, to best service the people in that community.

On the platform if a large number of new training vectors are mis-predicted by the current edge and cloud model, then the model is retrained to accurately predict and represent the borrowers in the community. Moreover if a user migrates to a different edge to get loans then the local edge model is used for loan approval. It is possible that the edge model could communicate with each other to create an aggregated model for an expanded edge network, however, that has not been implemented in this work.

For disconnected operations, having an edge model maintained at an edge small data center is useful. For use-cases where connectivity to a remote cloud server is not a problem, then the edge-optimized model could be hosted directly in the remote cloud, so that loan processing for users at an edge occurs in the remote cloud using an edge-optimized model for that edge location.

C. Microservices

Microservices architecture (MSA) is a versatile application design technique, which innately supports horizontal scalability, isolation, higher availability, and interoperability. Microservices are being positioned as the most optimal building block and deployment unit for producing and sustaining next-generation modular applications. These are enabling design, integration, composition and deployment patterns and platforms for speeding up the realization of enterprise-grade microservices-based applications. We have used the popular Kubernetes microservices-based platform to implement the MiSA system. Multi-container applications are being derived through Kubernetes, which is a key service and container orchestration platform. We use Flask, which is a Python library that makes it easy to set up Python functions that can be invoked via the web.

D. Blockchain + InterPlanetary File System

MiSA is hosted on a blockchain platform to inject security and trust into the system. We leveraged Quorum combined with IPFS (InterPlanetary File System) to realize our system. IPFS is used to store large amounts of data off-chain and place the immutable, permanent IPFS links as well as the cryptographic hash of our prediction model into the Quorum network. To provide security, we can encrypt the model with the public key of the user or user group with whom the model is to be shared with, who in turn can decrypt it using the user / user group's private PSK (preshared key) private key. The set-up Quorum network was then tested for transaction speed. Using the Raft Consensus algorithm, which provides fast block times(on the order of milliseconds instead of seconds), we get a high average transaction per second(TPS) of 99.2 using a blocking, non-async and single thread set-up on our local system. For microfinancing, the required TPS is typically expected to be reasonably low (say $<$ 10 transactions per second) so the TPS obtained appears to be adequate.

The smart contract basically implements three main functionalities - requestIPFS, sendIPFS and retrieveIPFS. The requestIPFS stores the address of the calling node which is then served one by one by owner of the model using the sendIPFS method of the smart contract. SendIPFS then maps the IPFS hash generated, on to the address retrieved from the requestIPFS. The user executing the requestIPFS functionality can then check 'retrieveIPFS' to see if it has received the cryptographic hash on its address, and if yes then retrieve it. Once the transaction is complete, the hash returned is then recorded in the Quorum network. This way we provide a secure base for the service which cannot be tampered.

IV. Results

We divide the result section according to our approaches and their performance.

A. Method 1: Based on borrower's loan details

In our first method we take the loan parameters as inputs and base our decision as well as our grading of the borrower based on it. All the three models described in the Methodology successfully classify the default and non-default loans with very high levels of accuracy. Thus, the P2P lenders can minimize their risk of investment failure by selecting an appropriate borrower by processing the loan applications through our service. We compare the performances of the three one by one.

1) Logistic Regression
The logistic regression model was accurate in classifying the default and non-default loans. The overall accuracy achieved by the model was 98.10%.
From the Table I, our logistic regression model was able to correctly predict 99.95% of the Non-Default loans and 70% of the default loans from our test dataset. Fig. 2. is the combined ROC-AUC curve of all our models. The ROC curve is trade-off between sensitivity (or TPR) and specificity (1 - TFR). Any classifier with its curve closer to the top left corner indicates better performance.

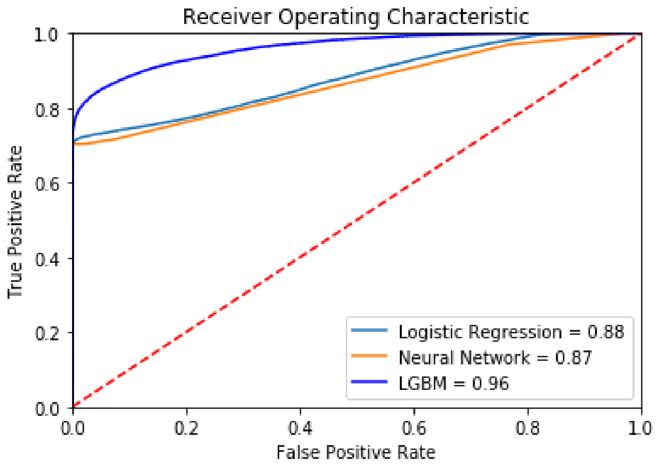

Fig. 2. ROC Curve of all the models

The ROC curve indicates the performance of our model with its AUC value being 0.88 which indicates a model with very good performance (Fig. 2).

TABLE I
CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Actual: Fully Paid	*Actual: Default*
Predicted: Fully Paid	255376	5039
Predicted: Default	132	11693
Percentage Correct	99.95%	70%

2) Neural Network
Our Neural Network model's performance was very similar to the performance of the Logistic Regression model with the accuracy of the model being slight better than the later (Table II). The overall accuracy score of the model is 98.13%.
The ROC curve of the neural network model is also very similar to that of the logistic regression model indicating a model with very good performance rate and an Area Under the Curve (AUC) value of 0.87 which suggests the same.

TABLE II
CONFUSION MATRIX FOR NEURAL NETWORK

	Actual: Fully Paid	*Actual: Default*
Predicted: Fully Paid	255508	5087
Predicted: Default	0	11645
Percentage Correct	100%	69.5%

3) Light GBM
The performance of Light GBM was the best out of the three machine learning models. It achieved an overall accuracy score of 98.23% which is better than both - Logistic Regression and Neural Network.
The ROC curve also indicates that the Light Gradient Boosting Machine is better than the other two models. LGBM grows trees vertically whereas other algorithms grow trees horizontally which means that LGBM grows trees leaf-wise whereas other algorithms grow trees level-wise. This approach appears to have helped LGBM to perform better than the other two models.

TABLE III
CONFUSION MATRIX FOR LGBM

	Actual: Fully Paid	*Actual: Default*
Predicted: Fully Paid	255430	4729
Predicted: Default	90	11992
Percentage Correct	99.9%	71.7%

This model produces an Area Under the Curve value of 0.96, which is more than both of the previous models. (Fig. 2.)

B. Method 2: Based on borrower's past transactions

In our second method, we focus only on the borrower and the past history of transactions on the borrower. By combining random rows as one borrower, from a dataset of 816722 we build a data frame which consists of 71673 borrowers in our first iteration. We repeat this step multiple times to obtain multiple dataset to work on and test our algorithm.
Once the data is ready to be processed, we apply our algorithm explained in the methodology to the given dataset. In the algorithm, we use two different techniques - one is where the principal amount paid back to the lender by the borrower is of the most importance and the grade is dependent only on it, the second is where along with principal amount paid back, interest rate is also equally important. We simulate the algorithm and get the following result which is a graph displaying the first 27 borrower's rating.

We can see from the figure (Fig. 3.) that the grading is pretty consistent and high across borrowers in the dataset. The median and mean both being 4.4 indicates that almost all past transactions of borrower's are quite successful which makes sense because our original dataset was imbalanced with completed transactions being more than defaulted transaction. In additional, current ongoing transactions aren't considered in the credit grade calculation since in this approach we just based it on the past transactions.

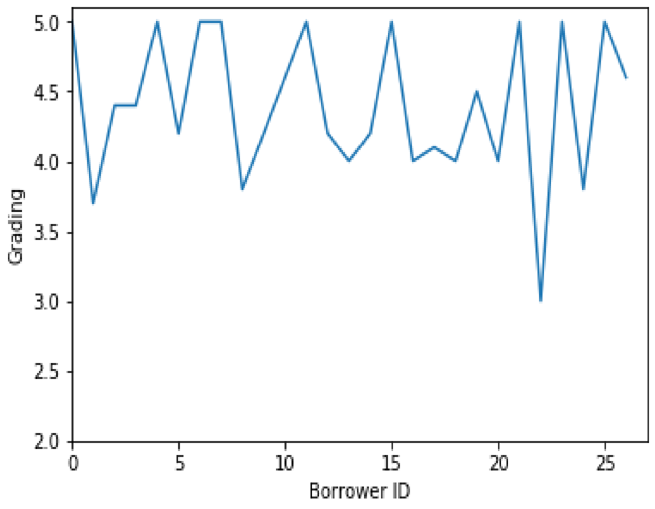

Fig. 3. Plot of Subset of Grading using Method 2

C. Method 3: Hybrid Approach

In our third method we combined both our previous approaches and come up with a holistic result. Combining the borrower's past transaction with the current loan parameters gives us the complete picture. In this approach we assigned the weightage to both approaches dynamically and is different from borrower to borrower. Since past transactions are best reflective of the borrower's capacity to pay back, relative importance is given more to past transactions when such information is available. However, if the number of completed transactions are very low, then a model derived purely based on past transactions may not full represent the borrower, as it can be statistically unreliable. It is possible that a borrower who has performed poorly with a few past transactions may have had to default due to unforeseen circumstances. In some cases, no past history of transactions may be available. Hence it is desirable to pursue a weighted combination of the grading g2 based on past transactions of the specific borrower under consideration (based on method 2) along with the grading g1 based on anonymized borrower loan data (based on method 1) As more information about past transactions related to a borrower become available, the system can give a larger weight to the grading based on the past transactions of a specific borrower. Here $\alpha = n/n_{\max}$ if $n \leq n_{\max}$ and $\alpha = 1$ if $n \geq n_{\max}$. Here n is the number of past transactions for which information is available, and nmax is a max threshold for weight determination. Thus an adaptive system approach with an estimated grade is given by -

$$g = (1 - \alpha) * g_1 + \alpha * g_2 \tag{7}$$

is utilized, where $\alpha \in$ [0,1], with increased weightage to past transactions for a borrower when such information is available. Additionally, we put a threshold value on the past If the borrower's number of past transactions exceeds $n_{\max}$ ($n_{\max}$ is set at 50 in our system) then, the a grading based on past transactions is best reflective of the users'behavior pattern and their likeliness of repaying the loan back. We simulated our third model and obtained a pretty consistent results as seen in the Fig. 4.

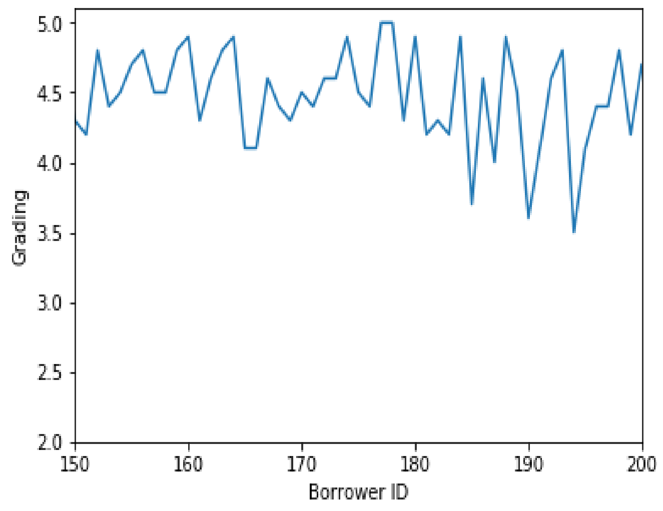

Fig. 4. Plot of Subset of Grading using Method 3

The grading is consistently high which is consistent with the patterns in the imbalanced dataset. The grading is then tested against our test vector which gives us an interesting insight into the performance of the approach. We infer that borrower's with a rating of 4.5 and above are 95% likely to repay and complete the loan in the stipulated time, whereas borrower's with a rating of less than 3.75 have a 60% chance of defaulting because of the imbalance in the dataset, rating the borrower relatively higher as a result. We thus conclude that the third model is the best of the three model for general cases.

D. Distributed Machine Learning In Emerging Infrastructure

In this distributed cloud+edge approach, edge communities with varying loan default percentages were considered. The performance of an edge-community-optimized model was studied relative to the performance of a generic cloud model across communities as the loan default percentage was varied. In the early phase as n < n_{max}, where n is the number of transactions at the edge and n_{max} is the threshold for the number of transaction after which the model starts to become consistent and more accurate, initially a purely cloud model is collapsed at the edge to take decision because of the inaccuracies associated with an early edge model. As the transactions are processed at the edge, the accuracy of the edge model gradually starts increasing from 89.45% with 50 samples to 99.16% with 200 samples (Table 5). As the accuracy of the model increases, a weighted combination of the cloud + edge model is used, with a greater weightage to the cloud model initially. The weightage to the cloud model is inversely proportional to the gradual increase in accuracy, which eventually transitions into a purely edge model or a weighted combination of an edge and cloud model with a greater weightage to the edge model.

TABLE IV
ACCURACY SCORE FOR VALUES OF N

n	Accuracy Score
50	89.48%
75	91.63%
100	94.01%
125	95.36%
150	97.83%
175	99.1%
200	99.16%

The cloud model being discussed is a generalized model trained over the complete dataset across all borrowers and communities in the network whereas the edge models are community specific models trained over a distinct community thereby making more accurate predictions for the future loans for that community. To support this claim, different datasets with different ratios of defaulted loans and fully paid loans are created and tested against the two models. Percentage of defaulted loans in the dataset increases from 25% to 99% to get a more holistic view and in doing so, an interesting trend is observed.

When both the generalised cloud model and the community/dataset specific edge model are tested against the created test dataset and the prediction normalized between the values 0 to 5, with 0 being a fully paid loan and 5 being a defaulted loan, the performance of the cloud model decreases whereas the performance of the edge model increases, as the percentage of default loans in the dataset increases from low to high. Initially when the default percentage is 25%, the cloud model performs slightly better than the edge model, with a higher accuracy of 92.5% and lower mean of 0.97 indicating that more of the loan entries are fully paid in the dataset. As the percentage of defaulted loans increase to 40%, the cloud and the edge model have a very similar performance, with the cloud model performing marginally better than the edge model because of its high variance and low mean. But as the default percentage increases even more, the edge model performs much better than the cloud, with the edge model having an accuracy of 89.33% whereas the cloud model having an accuracy of 82%, at 65% defaulted loan dataset. The performance further improves as the percentage of default loans increases in the dataset as seen from the Table 4.

$$KL\left(p,q\right) = \log\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2} \qquad (8)$$

Both the cloud model and the edge model are assumed to be a normal (gaussian) distribution, and hence the K-L (Kullback-Leibler) distance is calculated using the following equation, to measure how far apart both the models are, thereby giving us useful insights and confirming our previous prediction. We calculate the Symmetric K-L divergence using the following formula :

$$SymmetricKL = KL\left(p,q\right) + KL\left(q,p\right) \qquad (9)$$

TABLE V
CLOUD VS EDGE FOR DIFFERENT DEFAULT PERCENTAGES

DEFAULT %	CLOUD			EDGE			SYMMETRIC KL DIVERGENCE VALUES
	Accuracy	Variance	Mean	Accuracy	Variance	Mean	
25	92.5%	2.98	0.97	92.08%	2.85	1.41	0.066
40	88%	4.09	1.42	88%	3.09	2.04	0.149
65	82%	5.35	3.09	89.33%	2.36	4.02	0.623
80	73.7%	5.47	2.43	88.33%	1.85	3.96	1.49
90	78%	4.95	3.34	90.67%	0.96	4.49	2.49
99	64.7%	5.11	3.24	98.68%	0.12	4.86	31.49

The K-L divergence values of the edge model relative to the cloud model can be seen in Table V and Fig. 5. From the table, initially at 25% default loans, the KL Divergence value of the edge model is 0.066 whereas for 40%, the value increases to 0.149. The distance between the two models increases as the percentage of the defaulted loans in the dataset increases, with the symmetric K-L divergence increasing to 0.623 for a loan default percentage of 65%, and further increases to 31.49 for a loan default percentage of 99%. Thus the cloud and edge models for a given value of the loan default percentage become increasingly different relative to each other as the loan default percentage increases. It can also be seen from Table V that the accuracy of the edge model improves relative to the cloud model as the loan default percentage increases.

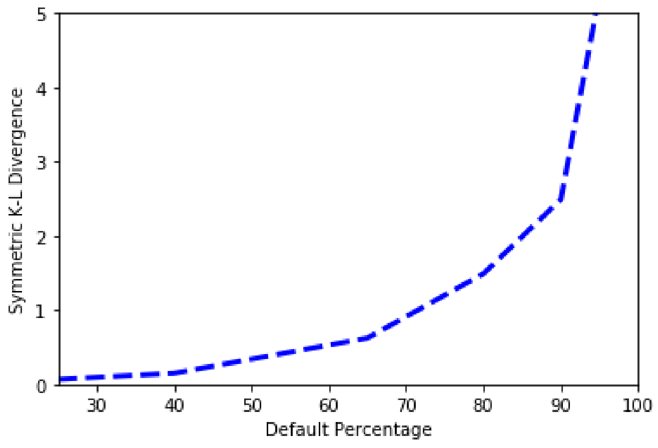

Fig. 5. Plot of the Symmetric KL Divergence Values

From the above results, to increase the reliability, accuracy and predictability of the system, the prediction for the bad community with the default rate to be anywhere above 40%, should be done at the edge providing much easier, quicker and hassle-free micro-loans to everyone in need of daily money in third-world countries.

V. CONCLUSION

To summarize, in this paper, we have leveraged a combination of emerging technologies such as microservices, blockchain technology, and machine learning at the edge to explore a microfinancing services application at the edge. An edge model could be hosted in the cloud or at an edge-server (if available) to cater to the needs of an edge community. It was observed that a machine-learned model for a local edge community can diverge from a more-generic machine-learned model in the cloud that is learned for a larger population that can include many edge communities. Depending on the nature of an edge community, such a divergence can be small or large, and in general, it would be advantageous to utilize edge learning for more accurate predictions at the edge. Blockchain technology based on Quorum was utilized to record transactions to enable provenance, immutability, and trust in the recorded information at the edge, with transactions recorded in IPFS, and a hash of each transaction recorded on the blockchain. This allows different lenders to leverage trusted shared information across each other to provide micro-financing services to needy borrowers at the edge to enable edge economies to flourish. It is hoped that the approach presented in this paper can be leveraged for other sustainable edge applications as well.

VI. REFERENCES

[1] A. Byanjankar, M. Heikkilä, J. Mezei, ”Predicting credit risk in peer-to-peer lending: A neural network approach”, 2015 IEEE Symposium Series on Computational Intelligence (pp. 719-725). IEEE.
[2] V. Kumar, S. Natarajan, S. Keerthana, K. M. Chinmayi, N. Lakshmi, ”Credit risk analysis in peer-to-peer lending system”, 2016 IEEE International Conference on Knowledge Engineering and Applications (ICKEA) (pp. 193-196). IEEE.
[3] J. Nalic, A. Svraka, ”Using data mining approaches to build credit scoring model: Case study—Implementation of credit scoring model in microfinance institution”, 2018 17th International Symposium Infoteh-Jahorina (pp. 1-5). IEEE.
[4] J. Benet, ”Ipfs-content addressed, versioned, p2p file system”, arXiv preprint arXiv:1407.3561.
[5] P. Zheng, J. You, J. Wang, X. Zeng,”A distributed and virtualized infrastructure for networked multimedia services”, 2013 IEEE China Summit and International Conference on Signal and Information Processing (pp. 556-560). IEEE.
[6] F. Slim, F. Guillemin, Y. Hadjadj-Aoul, ”CLOSE: A costless service offloading strategy for distributed edge cloud”, 2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC) (pp. 1-6). IEEE.
[7] A. Chandra, J. Weissman, B. Heintz, ”Decentralized Edge Clouds”, IEEE Internet Computing 17, 5 (September 2013), 70-73.
[8] P. G. Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, ”Edge-centric Computing: Vision and Challenges”. SIGCOMM Comput. Commun. Rev. 45, 5 (September 2015), 37-42.
[9] ”Lending Club”, Lending Club Loan Data 2007-2016 [Dataset]. Retrieved from https://www.lendingclub.com/info/download-data.action.
[10] M. Abadi et al, ”TensorFlow: Large-scale machine learning on heterogeneous systems”, 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). Software available from tensorflow.org.