Determining Loan Repayment with Deep Learning Models

Group 4:
Eric Chaves, Jordyn Dolly, Kwame Sefa-Boateng & Han Zhang
University of North Carolina at Charlotte
DSBA 6162: Knowledge Discovery in Databases
Dr. Xi (Sunshine) Niu
December 14th, 2023

Introduction

Financial institutions all over the world rely on loan disbursement portfolios to generate profits (*Bank Profitability: Consider the Source*, n.d.) Banks, in particular, issue loans to customers, both individuals and businesses, to support them in their various needs; however, they charge some regulated interest rates (*Banks: At the Heart of the Matter*, n.d.). Proper management of such loans is prudent for the sustainability of financial institutions. A high loan default rate means a loss of revenue and poses quite a stressful situation for businesses and investors (of Governors of the Federal Reserve System, 2019) Banks, therefore, must initiate pragmatic policies to ensure that such risks are mitigated. To that extent, the issuance of loans normally comes with a predetermined set of criteria that must be met before eligibility. Over the years, the amount of data collected on loans acquired becomes valuable data to learn from as it gives rise to some certainty about how acquired loans may perform. Although several financial institutions across the globe deploy different criteria for loan eligibility, there are some common or ideal features that are common to each of them (Fdic, n.d.)

In this project, we explore the possibility of using deep learning models to enable the prediction of the outcome of a loan disbursement based on several key features of past data. The data consists of 395,932 customer loan applications and their current state from 2014 to 2017, with a set of thirty-one variables. The aim is to provide a model that has high accuracy in predicting whether a loan is likely to be good or bad. Two deep learning models are deployed for this task, which include the multilayer perceptron and the wide and deep models. The two models largely provided a high accuracy rate of over 89% based on the test data. In the subsequent sections, we provide justifications for the chosen models and provide implementation methods using scikit-learn, TensorFlow, Keras, and other libraries to provide the model for prediction. Furthermore, an evaluation of the two models was provided to show how well the model performed with test data and the accuracy rate expected.

Outline and Justification for Chosen Learning Models

Deep Neural Network Model

Deep Neural Networks (DNN) prove invaluable in assessing loan repayment likelihood, leveraging their ability to discern complex, non-linear relationships within high-dimensional data (Borisov et al., 2021). This flexibility is particularly advantageous in financial contexts where capturing intricate patterns is essential for accurate predictions. DNNs offer practitioners the flexibility to experiment with architecture, adjusting layers, neurons, and activation functions to optimize performance (Alzubaidi et al., 2021). Their end-to-end learning capability further streamlines the modeling process, eliminating the need for manual feature engineering. However, challenges such as interpretability and the potential for the vanishing gradient problem must be carefully addressed. While DNNs empower accurate loan repayment predictions, practitioners must employ effective model evaluation, interpretation techniques, and optimization strategies to navigate these challenges and fully unlock the predictive potential of these sophisticated neural networks.

When preparing to build a model there are many different considerations to take into account. Time to market, development cost, model performance, and support all are important when considering which model to pursue. Taking these components into account, we wanted our first model to be straight-forward so that we would have a baseline model to compare against more complex models. If we were to find that the initial model does not perform well, we have

the opportunity to use the DNN's scalable approach that will allow us to produce more accurate results.

Multilayer Perceptron Model

The multilayer perceptron is a feed-forwarding artificial neural network that has many important use cases, such as complex data analysis, predictive analysis, automatic feature extraction, and improving decision-making. MLPs are useful for developing predictive models that are able to forecast future trends based on historical information (Fajar & Nurfalah, 2021). While an MLP is a specific type of deep neural network, by adding additional layers to neural networks, it makes them more useful than shallow networks for capturing complex patterns and representations in data. In the task of predicting whether a loan is likely to be in default or not, using this model for prediction was vital.

After choosing to use a DNN model, including a simple MLP model would help incorporate more layers in the hopes of improving accuracy. We chose to use an MLP as it has a great ability to learn from complex patterns and extract useful relationships in data, each node in the hidden layer receives inputs from the previous layer and computes a weighted average while adding a bias. The model composed of three main layers: the input, the hidden, and the output layers. The computed weights are then passed through an activation function, which introduces non-linearities into the model. An output is produced from the application of the activation function. An important feature of the model is the use of the backpropagation algorithm, where it refeeds the hidden layer with computed weights that did not make it through the output layer after going through the activation function.

Deep and Wide Model

A "deep and wide" neural network combines these two concepts. It has multiple hidden layers and many neurons in at least one of these layers. When implemented effectively the model will benefit from the capacity to identify hierarchical representations through depth and learn diverse features simultaneously through width. The advantages of a deep and wide neural network include the potential to handle complex datasets with many features and capture intricate patterns. A deep and wide model can require careful consideration of factors like computational resources, overfitting, and appropriate regularization techniques to prevent issues such as vanishing/exploding gradients to work appropriately. As suggested by Chen et al. (2016), the combination of trained wide linear models and deep neural networks brings to light the benefits of memorization and generalizations for recommender systems.

This model was relevant to the task of predicting the possibility of loan default due to its characteristics and performance. It has three key components, which are the wide component, useful for memorization, the deep component for generalization, and the integration of both the wide and deep components to generate the outputs. This model performs well with recommender systems and is thus key to the task.

Implementation of Deep Learning Models

Implementing a neural network involves several steps, and the specifics may vary depending on the model being discussed. To ensure that these models are working appropriately to help predict if a loan will be repaid or not, we focus on some key details. Each of these deep learning models requires a specific architecture. This includes the number of layers, neurons in each layer, activation functions, and the type of output (Mats Forssell). After determining these

aspects of the model, an optimization algorithm is implemented to minimize the error between the predicted and actual outputs during training. Initialize the weights and biases of the neural network randomly. Proper initialization is crucial for effective training. While the steps for each of the models may vary, computing error between the predicted and actual outputs and analyzing the models through classification reports and other measures helped us to determine the more effective model.

Deep Neural Network Model

After cleaning and preprocessing our data, we worked to properly implement the DNN. To gain a basic understanding of the model and to not overfit, a simple three layer model was used. When building the model, the first layer introduces a dense layer with 128 neurons and reshapes the training data. Also in the first layer, the activation function chosen is a rectified linear unit (ReLU). The activation function introduces nonlinearity to the data and helps lower the chances of encountering a vanishing gradient problem. The second layer is another Dense Layer with 64 neurons to continue transforming the data and also includes a ReLU activation function. It is also shown that we are decreasing the number of neurons in each layer. The final layer has a single neuron, indicating that the network is designed for binary classification. The activation function, 'sigmoid', is used for the output layer and places the output between 0 and 1 so that it matches our problem output.

Figure 1

Excerpt of DNN Code: Building the Model

```
# Build the DNN model
model = Sequential()
model.add(Dense(128, input_dim=X_train.shape[1], activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

When compiling the model for training, the optimizer is responsible for updating the weights of the neural network during training as seen in figure 2. We used 'adam' as it has an adaptive learning rate and is efficient. The loss function is a measure of how well the model is performing. We used 'binary_crossentropy' as it calculates the binary cross-entropy between predicted probabilities and true labels. We used 'accuracy' to measure the proportion of correctly predicted instances as it is a simple metric to compare across multiple models. Also added to improve the comparison of models was a classification report and confusion matrix.

Figure 2

Excerpt of DNN Code: Compiling the Model

```
# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Multilayer Perceptron Model

As discussed there are a wide variety of model setting that can impact the accuracy and efficiency of a model. To achieve the highest accuracy score within the MLP model, when building the model itself, we started with a dense layer of 64 neurons and combined that with a ReLU activation function. In this model we incorporated a dropout layer with a dropout rate of 0.5 to help prevent overfitting of that data. This is important as less than 20% of the target variable were 0 or that the loan would not be repaid. In the second dense layer, we included

another dense layer of 64 neurons and a ReLU activation function. Continuing to keep the concern for overfitting of the data a second dropout layer was included with a dropout rate of 0.5. The final dense layer with 1 neuron uses a sigmoid activation function suitable for binary classification of our target variable. To keep consistent between the DNN and MLP models, the MLP model was also compiled with the adam optimizer.

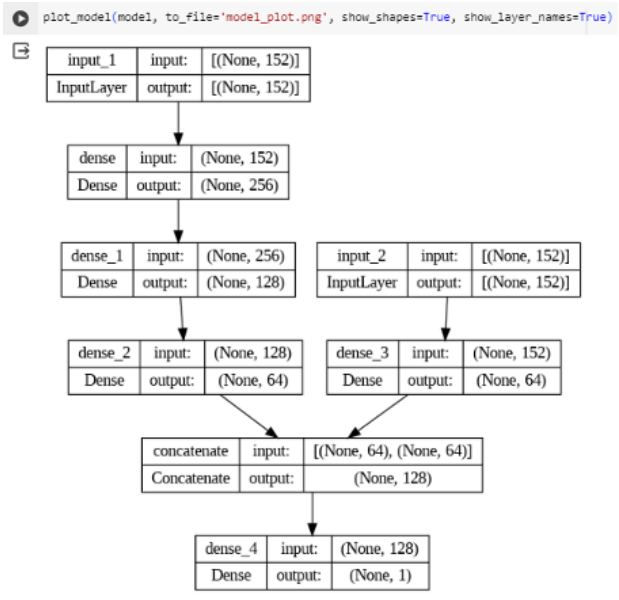
To give the model the best opportunity to train on the data, it was split between 80 % into training and 20% into testing. This ratio is not only commonly used but would also give our model enough training data to train on. This is a large data set with close to 400,000 samples of loan applications before the data was processed. After processing, we utilized over 50,000 loan samples. The model was then trained for 10 epochs with a batch size of 32. Additionally, a validation split of 20% is used during training to monitor the model's performance.

Deep and Wide Neural Network Model

The deep and wide model's articulates are the key idea of combining the memorization capabilities of a wide model with the generalization abilities of a deep model. This combination allows for both simple, rule-based information processing and complex pattern recognition. The text outlines the construction of the deep component with dense layers and the ReLU activation function, which is known for helping neural networks to learn and generalize better. The wide component is not depicted but is described as a linear model that memorizes rule-based patterns. This part of the model is crucial for capturing interactions that are explicit in the data.

Figure 3

A Visualization of the Deep and Wide Neural Network



The network with a 'wide' component and a 'deep' component. The deep part consists of multiple dense layers, which are fully connected neural network layers. These layers are commonly used in deep learning to enable the model to learn complex patterns through successive transformations. The concatenation layer merges the outputs of the last deep dense layer and the wide component, indicating an integration of memorization of rule-based patterns

from the wide part with the abstraction capabilities of the deep part. The final dense layer suggests a binary output, likely using a sigmoid activation function for binary classification.

Figure 4

Performance Classification Methods Used

```
# Print the metrics
print(f'Accuracy: {accuracy}')
print(f'Precision: {precision}')
print(f'Recall: {recall}')
print(f'F1 Score: {f1}')
print(f'ROC-AUC: {roc_auc}')
```

```
Accuracy: 0.896902269311882
Precision: 0.9047282720361619
Recall: 0.9895362229276103
F1 Score: 0.9452337827866103
ROC-AUC: 0.7865294953810738
```

The accuracy, precision, recall, F1 score, and ROC-AUC (Receiver Operating Characteristic - Area Under Curve) are all metrics used to evaluate the performance of classification models. The provided values are quite high, indicating that the model performs well on the data it was tested on. The precision metric signifies the model's accuracy in predicting positive classes, while recall measures the model's ability to find all the positive instances. The F1 score is the harmonic mean of precision and recall, providing a single metric for evaluation when the balance between precision and recall is important. Lastly, the ROC-AUC value gives an aggregate measure of performance across all possible classification thresholds.

Figure 5

Wide and Deep Model: Feature Importance Measurements

```
print(feature_importances.head(10))
```

```
                      feature  importance
15             last_pymnt_amnt    0.083772
3                     int_rate    0.046812
6                          dti    0.046254
20                tot_cur_bal    0.045740
11                  revol_bal    0.044980
12                 revol_util    0.044684
23  mths_since_earliest_cr_line    0.044535
21            total_rev_hi_lim    0.044284
5                   annual_inc    0.042046
4                  installment    0.040425
```

The feature selection and model training, suggesting that the model's architecture is complemented by careful feature engineering and selection, which are critical steps in building effective machine learning models. The list of features and their corresponding importance scores suggests that last_pymnt_amnt is the most significant predictor, followed by features like int_rate, dti (debt-to-income ratio), and tot_cur_bal (total current balance). These features could be related to financial data, possibly from a credit or lending domain, where the last payment amount, interest rate, and debt-to-income ratio are critical factors in predicting an outcome, such as loan default.

Model Evaluation

Table 1.

Summary of Model Performance

	Deep Neural Network (DNN)	**MultiLayer Perceptron (MLP)**	**Wide & Deep Model (WDM)**
Accuracy	89.63%	89.97%	89.69%
Precision	Class 0: 41% Class 1: 90%	Class 0: 58% Class 1: 100%	Class 0: 58% Class 1: 90%
Recall	Class 0: 6% Class 1: 99%	Class 0: 2% Class 1: 90%	Class 0: 2% Class 1: 100%
f1-Score	Class 0: 11% Class 1: 94%	Class 0: 4% Class 1: 95%	Class 0: 4% Class 1: 95%
ROC-AUC		80%	74.5%

Properly analyzing model performance is essential to determine the most accurate model, but also allows for improvements to be made on the model. With various performance metrics each calculating something slightly different, we utilized a variety of different measures to help us determine the best fit model. The first metric utilized is accuracy. Since accuracy is the calculation of the number of correct predictions divided by the total number of predictions, a high accuracy score generally indicates good model performance. Using different models accuracy is easily interpreted, however accuracy can tend to predict the majority class in imbalanced data sets. For this reason, additional metrics like precision, recall, f1-score, and ROC-AUC were included.

Beginning with our accuracy scores, we find that all three models appear to perform well. With only very minor details between each of the three models, the additional metrics became important. The precision score calculates the ratio of true positive predictions to the total predicted positives, and is very useful when trying to minimize the number of false positives (Jeni,… 2013). The MLP model performs ten percent greater than the DNN or WDM models at predicting true positives, indicating when people will repay their loans. The MLP model ties for the highest precision score with the WDM model at 58% when prediction class 0, when a loan is not repaid. Using recall, we can calculate the ratio of true positive predictions to the total number of actual positives, where the cost of false negatives is high. While the DNN and WDM models have near perfect recall, the models are extremely susceptible to overfitting. These high recall values for class 1, help indicate overfitting and with the cost of false negatives being high in this calculation this is not the best indication of model performance. The f1-score provides the combination of precision and recall. With the MLP and WDM models having the same balance of f1-score between class 0 and class 1, we utilize a final performance metric.

Area under the receiver operating characteristic curve (AUC-ROC), is commonly used in binary classification models, especially when a dataset is imbalanced and sensitive to performance of class 1 (Jeni, et al., 2013). It was the MLP model that had a higher ROC-AUC than the next most competitive model, the WDM. Having a value closer to 1.0 or 100% indicates better discrimination between loans being repaid and not. The MLP's AUC-ROC of

80% indicates that there is an 80% chance that the model will rank a randomly chosen positive case higher than a negative case.

Conclusion

Using deep learning, we created three different models to help predict whether an individual would repay their loan or not based on various criteria. Through building a simple deep neural network, a multilayer perceptron model, and a deep and wide neural network model, we were able to compare model implementation and performance to determine the best model for the dataset. Through all the performance metrics, the MLP model was found to be the best overall model. With not only the highest accuracy score but also a higher AUC-ROC, the model was most appropriate for determining loan repayment. Developing these models allowed for machine and deep learning to be used practically to find a model that is the most appropriate to work towards solving a real-world problem.

References

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00444-8

Bank Profitability: Consider the Source. (n.d.). Retrieved December 3, 2023, from https://www.imf.org/en/Blogs/Articles/2019/04/30/blog-bank-profitability-consider-the-source

Banks: At the Heart of the Matter. (n.d.). Retrieved December 3, 2023, from https://www.imf.org/en/Publications/fandd/issues/Series/Back-to-Basics/Banks

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., & Kasneci, G. (2021). *Deep Neural Networks and Tabular Data: A Survey*. https://doi.org/10.1109/TNNLS.2022.3229161

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., & Shah, H. (2016). *Wide & Deep Learning for Recommender Systems*. http://arxiv.org/abs/1606.07792

Fajar, M., & Nurfalah, Z. (2021). Hybrid Fourier Regression - Multilayer Perceptrons Neural Network for forecasting. *Statistical Journal of the IAOS*, *37*(4), 1199–1204. https://doi.org/10.3233/SJI-210876

Fdic. (n.d.). *LOANS Section 3.2 RMS Manual of Examination Policies 3.2-1*.

of Governors of the Federal Reserve System, B. (2019). *Financial Stability Report, May 2019*.

Forssell, Mats (n.d.). Hardware Implementation of Artificial Neural Networks. *Information Flow in Networks*. https://users.ece.cmu.edu/~pgrover/teaching/files/NeuromorphicComputing.pdf

Jeni, L. A., Cohn, J. F., & De La Torre, F. (2013). Facing Imbalanced Data Recommendations for the Use of Performance Metrics. *International Conference on Affective Computing and Intelligent Interaction and workshops : [proceedings]. ACII (Conference)*, *2013*, 245–251. https://doi.org/10.1109/ACII.2013.47