GSJ: Volume 11, Issue 7, July 2023, Online: ISSN 2320-9186
www.globalscientificjournal.com

A DEEP LEARNING PREDICTION MODEL FOR LOAN DEFAULT

[1]Abiodun M. Abidemi, Mojeed D. Ajegbile[2], Yusuff B. Ajegbile[3], Joy Adedeji[4], Cecilia Dada[5]

[1]Department of Computer Science and Quantitative Methods Austin Peay State University, Clarksville, TN, USA
[2]Department of Computer Science and Quantitative Methods Austin Peay State University, Clarksville, TN, USA
[3]Department of Health Education University of Ibadan, Nigeria
[4] Department of Information Technology, University of the Cumberlands, KY, USA
[5]Department of Zoology, University of Ibadan, Nigeria

Abstract

Deep learning, a subfield of machine learning, has been making significant strides and achieving remarkable accomplishments in recent years. Powered by large-scale neural networks and massive amounts of data, deep learning has revolutionized various domains, including computer vision, natural language processing, speech recognition, and more.

One of the most notable achievements of deep learning in recent times is its remarkable progress in computer vision tasks. Deep convolutional neural networks (CNNs) have achieved groundbreaking results in image classification, object detection, and image segmentation. Models like AlexNet, VGGNet, GoogLeNet, and ResNet have demonstrated unprecedented accuracy on benchmark datasets such as ImageNet, significantly surpassing human-level performance.

Moreover, deep learning has also enabled impressive advancements in the field of natural language processing (NLP). Recurrent neural networks (RNNs) and transformer models, such as the famous GPT (Generative Pre-trained Transformer) series, have pushed the boundaries of language understanding and generation. These models have excelled in tasks like machine translation, text summarization, sentiment analysis, and even generating coherent and contextually relevant text. Deep learning can be understood as a subfield of machine learning concerned with efficiently training artificial neural networks (NNs) with many layers.

This project focused on using deep learning models to predict loan defaults of credit loan customers to minimize the outrageous risks for financial companies.

Loan default prediction plays a crucial role in the financial industry, aiding lenders in managing risk and making informed lending decisions. In recent years, deep learning models have gained attention for their ability to capture complex patterns and dependencies in loan data. This

abstract provides an overview of a deep learning prediction model for loan default, highlighting its methodology, performance, and key findings.
The model is deployed using Keras API via TensorFlow 2.0. It uses the rectified linear unit (reLU) and Sigmoid function as the activation layers and Adam optimizer for the model optimization.
The proposed deep learning model utilizes a feed-forward neural network (FNN) architecture. The model incorporates feature engineering techniques, such as principal component analysis (PCA) and oversampling/undersampling, to address imbalanced datasets and enhance predictive accuracy. Data preprocessing steps ensure data integrity, including normalization and missing value imputation. The deep learning prediction model is evaluated using metrics such as accuracy, precision, recall, and the area under the receiver operating characteristic curve (AUC-ROC). The results demonstrate superior performance compared to traditional machine learning algorithms, showcasing the model's ability to accurately predict loan default. The interpretability of the model is addressed through explainable AI techniques, providing insights into the factors influencing loan default.
The model evaluation shows how effective deep learning networks predict loan defaulters and help to curb loan and credit risk for financial firms. The proposed deep learning prediction model for loan default shows promise in enhancing risk assessment and decision-making for lenders. Future research directions may involve exploring alternative deep learning architectures, such as graph neural networks (GNNs), to capture complex relationships between borrowers and lenders. Additionally, incorporating transfer learning approaches and large-scale labeled datasets can further improve the model's performance and address challenges related to data scarcity.

INTRODUCTION

Problem Statements:

Predicting loan default is a critical task for financial institutions and lenders to assess the creditworthiness of potential borrowers. Deep learning techniques have proven to be effective in various domains, including finance, and can be leveraged to develop predictive models for loan default.
Loan default can have significant effects on both individual businesses and the overall American economy. Here are some key impacts:

Effects on Businesses:

Financial Distress: Businesses often experience financial distress when they default on loans. This can lead to cash flow problems, hampering their ability to meet operational expenses, pay employees, and invest in growth opportunities.

Limited Access to Credit: Defaulting on a loan can damage a business's creditworthiness and make it harder for them to secure future loans. Lenders may view them as higher risk, leading to higher interest rates or denial of credit altogether.
Reputation and Relationships: Loan default can harm a business's reputation within the financial community. This may strain relationships with lenders, suppliers, and other stakeholders, making it difficult to establish new partnerships or negotiate favorable terms.
Business Closure: In extreme cases, loan default can lead to business closure. If the financial burden becomes unsustainable, the business may have to shut down, resulting in job losses, economic instability, and potential ripple effects on related industries.

Economy: Reduced Investment and Economic Growth: When businesses default on loans, it decreases their ability to invest in expansion, research and development, and job creation. This can hinder economic growth as investments drive innovation and productivity.

Increased Unemployment: Defaulting businesses may be forced to downsize or close, leading to job losses. This increases unemployment rates, reduces consumer spending, and can lead to a decrease in overall economic activity.

Financial System Stability: Widespread loan defaults can pose risks to the stability of the financial system. If many businesses default, it can weaken banks and financial institutions that have exposure to these loans, potentially leading to a cascading effect of financial instability.

Impact on Supply Chain: Loan defaults can disrupt supply chains as businesses struggle to pay suppliers and fulfill their obligations. This can create a chain reaction, affecting other businesses and industries that rely on these suppliers, leading to a slowdown in economic activity.

Government Intervention: In severe cases, widespread loan defaults may require government intervention to stabilize the economy. This can involve implementing stimulus packages, and financial assistance to affected regulatory measures to prevent future defaults. It's important to note that the effects of loan default can vary depending on the scale and severity of the default, as well as the resilience and responsiveness of the businesses and the overall economic environment. However, overall, loan defaults have the potential to disrupt businesses, impact employment, and have broader implications on the American economy.

Sometimes in financial institutions, many lenders, banks, and financial investment companies have gone bankrupt due to many factors that have not been completely addressed, but one major one is a result of lending to non-credible and potential loan defaulters. This loan default occurs when a borrower fails to pay back a debt according to the initial arrangements. In most cases in customer loans, it happens that successive payments have been missed over a period of weeks or months. The period between missing a loan payment and having the loan default is known as ***delinquency***. In the recent development, Data analysts and machine learning engineers have taken the responsibility to help rescue lenders from this high risk of bankruptcy by analyzing the future behavior of a prospective borrower.

Objective and Scope of the Research:

As described in the title,

- The overall objective of this project is to develop a predictive model to predict whether a potential borrower will pay back a loan based on his/her historical data.
- To safeguard banks, and financial investment institutions through the more stringent policies and requirements for financial loans.
- The project would also predict and suggest pressing factors that have strong correlations with loan status.

Literature Review

Neural networks are a fundamental component of artificial intelligence and machine learning. They are computational models inspired by the structure and functioning of biological neural networks. Over the years, various types of neural networks have been developed, each with its unique architecture and applications. This literature review explores some of the prominent types of neural networks and their characteristics.

1. Feedforward Neural Networks (FNN): Feedforward neural networks are the most basic type of neural network. They consist of an input layer, one or more hidden layers, and an output layer. Information flows through the network in a unidirectional manner, from the input layer to the output layer. FNNs are primarily used for pattern recognition, classification, and regression tasks [7].
2. Convolutional Neural Networks (CNN): Convolutional neural networks are designed specifically for processing structured grid-like data, such as images. They employ specialized layers, including convolutional, pooling, and fully connected layers. CNNs excel at tasks such as image classification, object detection, and image segmentation. They leverage the local connectivity pattern and shared weights to efficiently extract meaningful features from input data [8].
3. Recurrent Neural Networks (RNN): Recurrent neural networks are well-suited for processing sequential data, where the order of inputs is crucial. RNNs have feedback connections, allowing information to persist and be processed across different time steps. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are popular variations of RNNs that address the vanishing gradient problem and improve information retention. RNNs find applications in natural language processing, speech recognition, and time series analysis [9].
4. Generative Adversarial Networks (GAN): Generative Adversarial Networks consist of two neural networks: a generator and a discriminator network. The generator network learns to generate synthetic data that resembles the training data, while the discriminator network learns to distinguish between real and fake data. GANs have revolutionized the field of generative modeling, enabling the creation of realistic images, videos, and audio. They have also been used for data augmentation, style transfer, and anomaly detection [10].
5. Self-Organizing Maps (SOM): Self-Organizing Maps, also known as Kohonen maps, are unsupervised learning neural networks that create a low-dimensional representation of high-dimensional data. SOMs organize input data into a grid of neurons, where each neuron represents a specific region of the input space. SOMs are commonly used for visualization, clustering, and data exploration tasks [11].
6. Radial Basis Function Networks (RBFN): Radial Basis Function Networks employ radial basis functions as activation functions. They consist of input nodes, hidden nodes with radial basis functions, and output nodes. RBFNs are particularly useful for function approximation and interpolation tasks. They are capable of learning complex relationships between input and output data [12].

Neural networks have witnessed significant advancements and diversification over the years, giving rise to various types of networks with specialized architectures and applications. Feedforward neural networks, convolutional neural networks, recurrent neural networks, generative adversarial networks, self-organizing maps, and radial basis function networks are just a few examples of the extensive range of neural network models available. Each type offers unique strengths and is suitable for different problem domains. Understanding the characteristics and capabilities of these neural network types can guide researchers and practitioners in selecting the most appropriate model for their specific tasks.

Method

The accurate prediction of loan default is a crucial task in the field of finance, as it enables lenders to assess the risk associated with granting loans. With the advent of deep learning techniques, researchers have explored the application of these models for predicting loan default. This literature review aims to provide an overview of the existing studies that have employed deep learning prediction models for loan default analysis, highlighting their methodologies, performance, and key findings.

Methodologies and Architectures:

Various deep learning architectures have been utilized in loan default prediction models, including recurrent neural networks (RNNs), convolutional neural networks (CNNs), and hybrid models. RNNs, particularly Long Short-Term Memory (LSTM) networks, have gained popularity due to their ability to capture sequential dependencies in loan payment data. CNNs have been employed to extract spatial features from structured loan datasets. Hybrid models combine different deep learning architectures to leverage the strengths of each model type. These models typically incorporate feature engineering techniques to enhance the predictive power of deep learning models.

Feature Engineering and Data Preprocessing:

Feature engineering plays a crucial role in loan default prediction models. Researchers have employed various techniques such as principal component analysis (PCA), autoencoders, and oversampling/undersampling to handle imbalanced datasets. Additionally, data preprocessing steps, such as normalization and missing value imputation, are employed to ensure the quality and integrity of the data. These preprocessing techniques are essential for optimizing the performance of deep learning models in loan default prediction.

Performance Evaluation:

The performance of deep learning prediction models for loan default is typically evaluated using metrics such as accuracy, precision, recall, and the area under the receiver operating characteristic curve (AUC-ROC). Studies have reported promising results, with deep learning models outperforming traditional machine learning algorithms in terms of predictive accuracy. The ability of deep learning models to capture complex patterns and dependencies in loan payment data has contributed to their superior performance.

Challenges and Limitations:

Despite the advancements in deep learning techniques for loan default prediction, there are several challenges and limitations. The interpretability of deep learning models remains a concern, as they are often considered black-box models. ensuring transparency and explainability in the decision-making process is crucial, especially in the financial domain. Furthermore, the requirement of large-scale labeled datasets poses a challenge, as obtaining such datasets with accurate default labels can be expensive and time-consuming.

Future Directions:

Future research in this field can focus on addressing the challenges associated with deep learning models for loan default prediction. Incorporating explainable AI techniques to enhance interpretability and transparency can provide insights into the factors driving loan default. Additionally, developing transfer learning approaches that leverage pre-trained models from related domains could help alleviate the data scarcity issue. Exploring alternative deep learning architectures, such as graph neural networks (GNNs), could also prove beneficial in capturing the complex relationships between borrowers and lenders.

Data Sources:

A diverse and comprehensive dataset is essential to build a loan default prediction model. This dataset should include a range of features related to the borrower's financial history, credit scores, employment details, loan terms, and any other relevant information. Historical loan data with labeled default/non-default instances is also required for training the model.

We used a subset of the LendingClub DataSet obtained from Kaggle: https://www.kaggle.com/wordsforthewise/lending-club, which includes 396030 entries and 28 feature variables, with loan status as the target variable we are predicting.

Tools and Techniques:

Several tools and techniques were employed in deep learning to make the problems of predictions from data easier, the following tools in Python 3.7 were used:

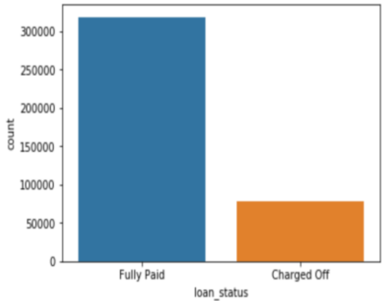
- Jupyter notebook IDE was used for Python programming and Microsoft Excel for data storage.
- Pandas pd, Numpy np, and Statistical summaries for numerical features in Python.
- Graphs (Histogram, Box Plots, ScatteredPlot, Bar Chart, HeatMap…) for data visualization with the seaborn, matplotlib libraries.
- Studied the variables for their relevance to the model and identified significant correlations.
- Using MinMaxScaler for data scaling and Deep Neural Network for Model Development.
- Used Deep Learning Neural Network (DNN), ReLU Activation Functions, Sigmoid Loss Functions, Optimization, BackPropagation, Gradient Descent.

EXPLORATORY DATA ANALYSIS.

Our aim here is to get a prior understanding of the data, which variables are important, a view of summary statistics, and proper data visualization for a better understanding of the data distribution and relationships.

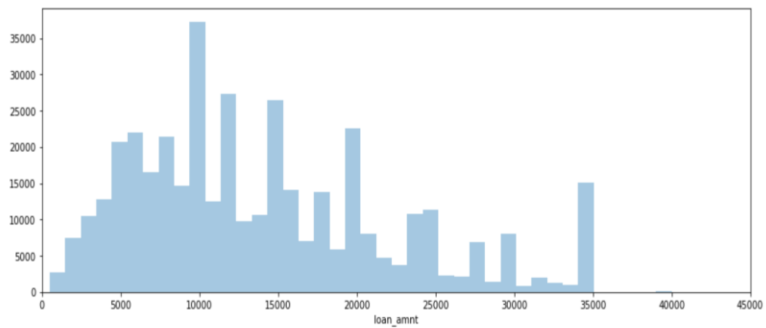
Exploring the Target Variable, Loan Status:

```
n [20]:  # We want to explore the counts for the loan status, to have a view of the number of entreis in the da
         sns.countplot(x='loan_status',data = data_lc)

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x17dc45c97f0>
```



The chart here shows the proportion of the data set that is either Fully Paid or Charged Off.

```
n [32]:  # Create a histogram/distribution plot of the loan_amnt column
         plt.figure(figsize=(15,5))
         plt.xlim(0,45000)
         sns.distplot(data_lc['loan_amnt'],kde=False,bins=40)

Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x17dbbd6a9e8>
```



The histogram indicates how the loan amount is distributed throughout the data set. It shows the amount that gathered more receivers. A loan of $10000 appears to have the highest receiver but in general, a loan amount below $15000 gathered more population from the data set.

Correlation between the Continuous Variables:

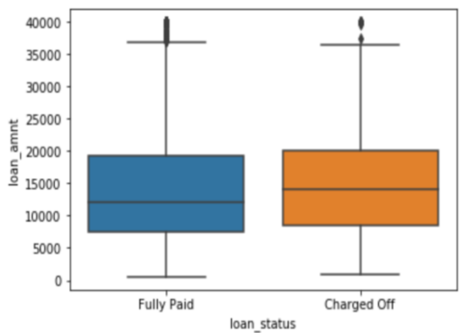
```
In [47]:  plt.figure(figsize=(15,7))
          sns.heatmap(data_lc.corr(),annot=True)
          #plt.ylim(10, 0)

Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x17dbd04eb00>
```

The HeatMap visualization for features correlation indicates a stronger correlation (0.95) between installment and loan amount, these two features were explored further.

Exploring relationship loan_status and the Loan Amount:

```
sns.boxplot(x='loan_status',y='loan_amnt',data=data_lc)
```

```
ut[51]: <matplotlib.axes._subplots.AxesSubplot at 0x17dbd915da0>
```



It is observed that the average loan amount for charged-off customers is higher. The summary statistics show 15126.300 for Charged-off and 13866.878 for Fully paid loan status.

DATA PREPROCESSING AND FEATURE ENGINEERING:

Deep learning models can often benefit from feature engineering techniques. Domain knowledge and insights can be utilized to create new meaningful features or transformations of existing features that might improve the predictive power of the model.
The collected data needs to be preprocessed to ensure its quality and compatibility with deep learning algorithms. This step involves handling missing values, removing outliers, normalizing numeric features, and encoding categorical variables.

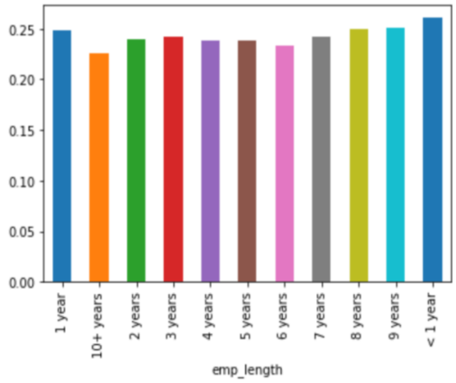
Dealing With the missing data:

```
In [77]: # we get the sum of the missing data in each column.
         data_lc.isnull().sum()
```

```
Out[77]: loan_amnt                  0
         term                       0
         int_rate                   0
         installment                0
         grade                      0
         sub_grade                  0
         emp_title              22927
         emp_length             18301
         home_ownership             0
         annual_inc                 0
         verification_status        0
         issue_d                    0
         loan_status                0
         purpose                    0
         title                   1755
         dti                        0
         earliest_cr_line           0
         open_acc                   0
         pub_rec                    0
         revol_bal                  0
         revol_util               276
         total_acc                  0
         initial_list_status        0
         application_type           0
         mort_acc               37795
         pub_rec_bankruptcies     535
         address                    0
         loan_repaid                0
         dtype: int64
```

Having explored the emp_title feature, we decided to remove that column since it appears there are too many unique job titles in the dataset and if we must convert this to a dummy variable feature, almost half of the people have unique job titles, which seems not very informative.

```
In [107]:  emp_lenProp.plot(kind='bar')
   Out[107]: <matplotlib.axes._subplots.AxesSubplot at 0x17dc6d91978>
```

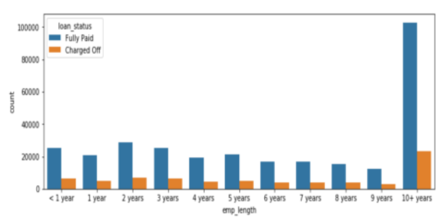


Charged-off rates are extremely similar across all employment lengths. So, we decided to drop the emp_length column. The 'mort_acc' feature has about the highest proportion of missing data, we fill in the missing data points. The correlation chart shows “total_acc” has the strongest correlation with 'mort_acc'. Then the missing points were replaced with the mean of the correlated column “total_acc” with a groupby function for the number of accounts.

Convert categorical string features to dummy variables:

Some of the categorical variables were converted to dummy variables, the "grade" feature was dropped for the "sub_grade", converted to dummy and concatenated to the original data frame.

```
n [101]:  #Plot out the countplot with a hue to separate Fully Paid vs Charged Off
          plt.figure(figsize=(12,4))
          sns.countplot(x='emp_length',data=data_lc,order=emp_length_order, hue = 'loan_status')
  Out[101]: <matplotlib.axes._subplots.AxesSubplot at 0x17dc6ce27f0>
```



Also converted the 'verification_status', 'application_type','initial_list_status','purpose' to dummies, dropped the initial variables to avoid duplicate information, and concatenated them to the original data frame. Feature extraction of the "address" column was done by creating a "zip_code" column that extracts the zip code from the address, the "address" column dropped and concatenated the "zip_code" column to the data.

CREATING & FITTING THE MODEL

Train_Test_Split:

This step involves assigning predictors and target variables.

We Imported train_test_split from the Sklearn library in python and performed a train/test split with test_size=0.2 and a random_state of 101. We used a MinMaxScaler to normalize the feature data training set and the test set, to avoid data leakage from the test set so we only fit on the training set.

Training Set:

The training set is a subset of the data that is used to train a model. It contains labeled examples where the input data and corresponding output (target variable) are known. The primary purpose of the training set is to allow the model to learn patterns and relationships between the input variables and the target variable. By exposing the model to a diverse range of examples, it can infer generalizable patterns and optimize its internal parameters to make accurate predictions.

Test Set:

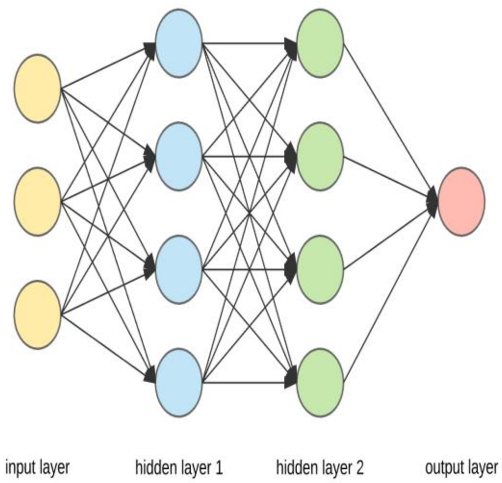
The test set, on the other hand, is a separate subset of data that is used to assess the performance of the trained model. It consists of examples that the model has not seen during training. The test set evaluates how well the model can generalize its learned knowledge to new, unseen data. By measuring the model's performance on the test set, we can estimate its effectiveness in making predictions and understand how it may perform in real-world scenarios.

The reasons for having distinct training and test sets are as follows:

1. Performance Evaluation: The test set provides an unbiased estimate of how well the model will perform on new, unseen data. It helps evaluate the model's generalization ability, ensuring that it can make accurate predictions beyond the examples it has been trained on.
2. Overfitting Detection: Overfitting occurs when a model becomes too complex and learns noise or irrelevant patterns from the training data. By evaluating the model on the test set, we can detect if it has overfitted by comparing its performance on the training set and test set. If the model performs significantly better on the training set but poorly on the test set, it indicates overfitting.
3. Hyperparameter Tuning: Machine learning models often have hyperparameters that need to be tuned to optimize their performance. The test set can be used to compare the performance of different models or parameter configurations and select the one that generalizes best to new data.
4. Model Selection: In some cases, multiple models or algorithms are compared to determine the best-performing one. The test set allows for a fair comparison between different models by evaluating their performance on the same data.

Creating Deep Neural Network Models:

Deep learning models for loan default prediction can be built using various architectures, such as feedforward neural networks, recurrent neural networks (RNNs), or convolutional neural networks (CNNs), depending on the nature of the data and the problem at hand.
We made the first layers have the same number of features, 78 (X_train.shape), Dropout layer will help prevent overfeeding. The algorithm used 4 layers deep learning neural network to model the prediction, an input layer with 78 nodes, a first hidden layer with 39 nodes, a second hidden layer with 19 nodes, and an output layer with 1 node as a classification model.



We used "Sigmoid" activation functions as the "output" layer for 0 or 1 for a classification model with the expected result for one neuron.

Mathematical Formulation of the Model

The three basic functions in the model are:
(1) Cost Function
(2) Gradient Learning algorithm
(3) Output units

Cost Function:

Cost function is a measure of how well a neural network performs on a given set of data. It is typically a single number that is calculated by comparing the network's predictions to the actual data labels.

The goal of training a neural network is to minimize the cost function. This is done by adjusting the network's weights and biases to make predictions as close to the actual labels as possible.

Cross-entropy: This is the most used cost function for classification problems. It calculates the difference between the network's predicted probability distribution and the true probability distribution of the labels. The formula for Binary cross entropy is given as:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Gradient Learning:

In a neural network, the learning algorithm calculates the gradient of the loss function with respect to the network's parameters, such as weights and biases, and updates these parameters in the direction that minimizes the loss. The gradient provides information about how the loss changes as the parameters vary, enabling the algorithm to search for the optimal set of parameters.

The gradient learning algorithm typically employs a technique called backpropagation to compute the gradients efficiently. Backpropagation calculates the gradient of the loss function by recursively applying the chain rule of calculus through the layers of the neural network. It starts from the output layer and propagates the error backward, updating the weights and biases at each layer.

Hyperparameter Tuning:

Deep learning models often have various hyperparameters that need to be optimized for better performance. Techniques like grid search, random search, or Bayesian optimization can be used to find the optimal combination of hyperparameters.

Performing binary classification, the model compiled uses "binary cross-entropy" as a loss function. we also deployed the Adam optimizer for model optimization.

Model Fitting & Data Validation:

We fit the model to the training data for at least 25 epochs, add in the validation data for later plotting, and a batch size of 256.

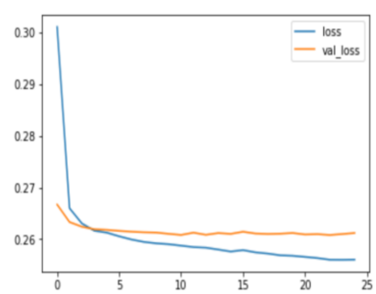
MODEL EVALUATION

Evaluating Model Performance:

Recall during model training, we passed in both the validation set and the training set, let's visualize these two sets:

```
In [101]:  #Let's Plot out the validation Loss versus the training Loss.

           losses = pd.DataFrame(model.history.history)
```

```
In [102]:  #Plot 'Loss' against 'val_Loss'
           losses[['loss','val_loss']].plot()

Out[102]:  <matplotlib.axes._subplots.AxesSubplot at 0x14552155748>
```



We noticed a similar behavior in the plot, it indicates the training loss(loss) and validation loss are both decreasing but there appears to be a slight improvement in our validation loss.

Predictions from Model:
Created predictions from the test set and displayed a classification report and confusion matrix for the test set.

```
In [105]:  print(classification_report(y_test,predictions))

              precision    recall  f1-score   support

           0       0.98      0.44      0.61     15658
           1       0.88      1.00      0.93     63386

   micro avg       0.89      0.89      0.89     79044
   macro avg       0.93      0.72      0.77     79044
weighted avg       0.90      0.89      0.87     79044


In [106]:  confusion_matrix(y_test,predictions)

Out[106]: array([[ 6859,  8799],
                 [  107, 63279]], dtype=int64)
```

Recommendation & Future Improvement:

For our model evaluation, we are more concerned about the ***Precision, Recall*** and ***F1-score***. A precision of 98% and a model accuracy of 89% looks very beautiful for our model, but F1-score on 0- class, 61% and 44% Recall can be improved further by playing around many of our neural networks model parameters like increasing the epochs, adding more layers, adjusting the Dropout function and adding early stopping callback to boost the model.

Conclusion
Going by the model accuracy of 89%, we would say the model will be able to predict correctly whether a potential credit loan borrower should be classified as Fully paid or Charged-off. Therefore, by leveraging the power of deep learning, financial institutions can develop robust loan default prediction models that help them make informed decisions, mitigate risks, and streamline the lending process. However, it's important to note that deep learning models require substantial computational resources and extensive data to achieve reliable results.
Deep learning prediction models have demonstrated promising performance in loan default prediction, surpassing traditional machine learning algorithms. The application of various deep learning architectures and feature engineering techniques has enhanced the predictive accuracy of these models. However, challenges related to interpretability and the availability of labeled datasets still exist. Future research should address these challenges and explore alternative deep-learning approaches to further improve loan default prediction models in the financial domain.

References

[1] Jiang, Li., William Rich and Donald Buhl-Brown., 2015, "Texture Analysis of Remote Sensing Imagery with Clustering and Bayesian Inference", Technical Report HTL-26, CFD-4, Austin Peay State Univ., Clarksville, TN., Vol. 1, pp. 1.

[2] Sebastian, Raschka, and Vahid, Mirjalili., 2019 "Machine Learning and Deep Learning with Python, Scikit-learn, and TensorFlow2", Textbook, Packt Publishing Ltd., Birmingham B3 2PB, UK., 3rd Edition, pp. 383-450.

[3] Karthikeyan, Gnanasekaran., Shrinivasabharathi, Balasubramanian., Sankaranarayanan, Mahadevan., and Nagesh, Shenoy M., 2016, “Identifying Patterns In New Delhi's Air Pollution - Final Report”, PGP-BABI 8 Bangalore, Indian.

[4] Wes McKinney., 2018, “Python for Data Analysis- Data Wrangling With Pandas, Numpy, and IPython”. Ph.D. Thesis, Published by O'REILY Media, Inc., 1005 Gravenstein Highway North, Sebastol, CA 95472.

[5] Rudolph, Russell., " Step-by-Step Guide To Implement Machine Learning Algorithms with Python," 2018

[6] Rob, Hyndman., “Stackexchange open source Website https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-network, Last Accessed May 5, 2020

[7] Murat H. et al, “A brief Review of Feed-Forward Neural Networks”, Commun. Fac. Sci. Univ. Ank. Series A2-A3, V.50(1) pp 11-17 (2006).

[8] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, Tsuhan Chen, Recent advances in convolutional neural networks, Pattern Recognition, Volume 77, 2018, Pages 354-377, ISSN 0031-3203.

[9] Salehinejad, H., Baarbe, J., Sankar, S., Barfett, J., Colak, E., & Valaee, S. (2017). Recent Advances in Recurrent Neural Networks. *ArXiv, abs/1801.01078*.

[10] Antonia Creswell et al, (2017), Generative Adversarial Networks: An Overview ArXiv:1710.07035v1

[11] Van Hulle, M.M. (2012). Self-organizing Maps. In: Rozenberg, G., Bäck, T., Kok, J.N. (eds) Handbook of Natural Computing. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-92910-9_19

[12] Ghosh, J., Nag, A. (2001). An Overview of Radial Basis Function Networks. In: Howlett, R.J., Jain, L.C. (eds) Radial Basis Function Networks 2. Studies in Fuzziness and Soft Computing, vol 67. Physica, Heidelberg. https://doi.org/10.1007/978-3-7908-1826-0_1