**A red and black logo

Description automatically generated**

**Deep Learning Model for Churn Prediction: A Comprehensive Analysis**

Presented to

Professor Fatih Nayebi

By

Barabasz, Michelle - 261152119

MGSC 673

McGill University - Desautels Faculty of Management

**Table of Contents**

[*1. Introduction* 3](#_Toc163554669)

[*2. Dataset Acquisition and Preprocessing and Data Spitting* 3](#_Toc163554670)

[*3. Model Development* 4](#_Toc163554671)

[*3.1. Converting to PyTorch Tensors* 4](#_Toc163554672)

[*3.2. Defining Custom Dataset and Sampler* 4](#_Toc163554673)

[*3.3. Defining Model Architecture* 5](#_Toc163554674)

[*3.4. Defining Training and Evaluation Function* 5](#_Toc163554675)

[*3.5. Hyperparameter Tuning* 6](#_Toc163554676)

[*3.6. Main Function* 7](#_Toc163554677)

[*4. Model Results* 7](#_Toc163554678)

[*6. Utilizing the Churn Prediction Model for Business* 8](#_Toc163554679)

[*7. Conclusion* 8](#_Toc163554680)

[*8. Dataset Used* 9](#_Toc163554681)

# *1. Introduction*

Customer churn, the phenomenon where customers terminate their association with a company, presents significant challenges for businesses across various industries. Churn prediction stands as a crucial endeavor for companies, notably in sectors like telecommunications, banking, and subscription services. The task involves discerning customers who are prone to ceasing their usage of a service or product. Anticipating churn empowers businesses to enact pre-emptive strategies aimed at customer retention, thereby mitigating revenue loss. This report delves into an approach for churn prediction employing neural networks, with a specific focus on the feedforward neural network (FNN) architecture. By forecasting churn, companies can implement proactive measures to foster customer loyalty and bolster revenue streams. This report provides an exhaustive examination of predicting customer churn utilizing deep learning methodologies.

# *2. Dataset Acquisition and Preprocessing and Data Spitting*

The dataset utilized in this investigation originates from a financial institution, encompassing diverse customer attributes, transaction records, and churn statuses. Data preprocessing procedures entail refining the dataset by eliminating irrelevant columns, such as customer identifiers, and applying one-hot encoding to categorical variables, priming the data for subsequent model training. Initially, the code snippet undertakes dataset preprocessing for churn prediction employing machine learning techniques. It starts by removing unnecessary columns ('RowNumber', 'CustomerId', and 'Surname'), followed by encoding categorical variables ('Geography' and 'Gender') into numerical representations using one-hot encoding. Next, the dataset is partitioned into feature variables (X) and the target variable (y), signifying whether a customer exited the service. Standardization of features is then conducted using StandardScaler to normalize them, promoting algorithmic convergence. Finally, the preprocessed dataset is divided into training and testing subsets, adhering to an 80-20 split, ensuring robust model evaluation.

# *3. Model Development*

In this study, a feedforward neural network (FNN) architecture is devised and implemented using PyTorch for the purpose of churn prediction. The FNN architecture comprises an input layer, two hidden layers with adjustable sizes, and an output layer tailored for binary classification of churn status. A systematic exploration of various network configurations, including architectures, activation functions, and learning rates, is conducted to determine the optimal model architecture.

## *3.1. Converting to PyTorch Tensors*

To facilitate model development, the code transforms the training and testing data (X\_train, X\_test, y\_train, y\_test) from NumPy arrays to PyTorch tensors. This conversion is pivotal as PyTorch mandates tensors as inputs for constructing and training neural network models. Utilizing the torch.tensor() function, the data is converted with specifications for data type, torch.float32 for input features, and torch.long for target labels.

## *3.2. Defining Custom Dataset and Sampler*

Subsequently, a custom dataset class named CustomDataset is defined, inheriting from PyTorch's Dataset class. This class serves to encapsulate the training and testing data, implementing the len and getitem methods essential for iterative dataset traversal during training. Additionally, a sampler is crafted to address the imbalanced class distribution of the target variable in the training data, employing the WeightedRandomSampler to ensure balanced representation during training.

## *3.3. Defining Model Architecture*

The model architecture is outlined using PyTorch's nn.Module class. The FNN class encapsulates the neural network architecture, with parameters specifying input size, sizes of hidden layers (hidden\_size1 and hidden\_size2), and output size. Comprising three fully connected layers integrated with Rectified Linear Unit (ReLU) activation functions, this architecture lays the foundation for the churn prediction model. By utilizing the nn.Sequential module, the layers are sequentially arranged, facilitating a streamlined and concise representation of the network architecture. The incorporation of ReLU activation functions fosters non-linearity within the model, enabling it to learn complex patterns and relationships inherent in the data.

During the model development process, extensive experimentation was conducted with various aspects of the network architecture, including variations in the number of hidden layers and neurons, to optimize performance. Ultimately, the architecture described above demonstrated superior predictive capabilities and convergence stability, making it the optimal choice for the churn prediction model.

## *3.4. Defining Training and Evaluation Function*

A designated function, train\_evaluate\_model, is formulated to organize the training and evaluation of the neural network model. This function encompasses the model, train and test data loaders, loss criterion (Cross Entropy Loss), optimizer (utilizing the Adam optimizer), and number of epochs as input parameters. Iterating over training data batches, it computes the Cross Entropy Loss, executes backpropagation, and updates model parameters. Post-training, the model's performance is evaluated on the test data, with metrics such as accuracy, precision, recall, and F1 score computed and reported.

Throughout the model development process, different hyperparameters, including learning rates, were experimented with to enhance model performance. However, the configuration outlined above consistently yielded the best results, demonstrating superior performance in terms of predictive accuracy and stability.

## *3.5. Hyperparameter Tuning*

A specialized function, hyperparameter\_tuning, is developed to conduct hyperparameter tuning. This function arranges a grid search across predefined hyperparameter combinations to identify the optimal configuration maximizing the F1 score. By iterating through the parameter grid using ParameterGrid, the function explores various combinations of hyperparameters, encompassing hidden layer sizes (hidden\_size1 and hidden\_size2), learning rates, and number of epochs. For each parameter combination, a new FNN model is instantiated, trained, and evaluated utilizing cross-validation techniques. Throughout the process, the F1 score is utilized to strike a balance between minimizing false positives and false negatives, which is crucial in churn prediction scenarios where both types of errors can have significant implications. Upon completion of the grid search, the function identifies the best hyperparameters and corresponding evaluation metrics. These metrics, including accuracy, precision, recall, and F1 score, are pivotal in assessing the efficacy of the model under different configurations. By reporting the best hyperparameters alongside their associated metrics, the function facilitates informed decision-making and model refinement, enabling the development of a high-performing churn prediction model.

## *3.6. Main Function*

Lastly, a main function coordinates the entire model development process. It initializes custom datasets for training and testing, implements oversampling of the minority class using the weighted random sampler, creates data loaders for training and testing, and calls upon the hyperparameter tuning function to identify the optimal model configuration. This framework ensures systematic model development and optimization for effective churn prediction.

# *4. Model Results*

The trained model is evaluated using standard performance metrics such as accuracy, precision, recall, and F1 score. Accuracy measures the overall correctness of predictions, precision measures the proportion of true positives among all predicted positives, recall measures the proportion of true positives among all actual positives, and F1 score provides a balance between precision and recall.

The model demonstrates an accuracy of 86.15%, indicating a high level of correctness in predicting churn. However, the precision of 70.28% suggests that there is room for improvement in accurately identifying customers who are likely to churn. Despite this, the model achieves a recall of 51.15%, indicating its capability to capture a significant portion of actual churn instances. The F1 score of 59.20% underscores a balanced performance between precision and recall. These results indicate that while the model exhibits strong predictive capabilities, there are areas for refinement to enhance its precision and overall performance. Further investigation into feature importance and model tuning may provide valuable insights into improving the model's accuracy and precision in identifying churn patterns.

# *6. Utilizing the Churn Prediction Model for Business*

The churn prediction model equips businesses with the ability to identify customers at risk of churn by analyzing historical behavior and attributes. With these insights, businesses can prioritize efforts to retain high-value customers who are most susceptible to churn. Utilizing predictions from the model, businesses can tailor retention strategies to meet the specific needs and preferences of at-risk customers, employing personalized offers, loyalty programs, and enhanced customer support to incentivize continued engagement. Embracing proactive customer engagement, businesses can initiate targeted communications, surveys, and proactive customer service interventions to address concerns and prevent churn before it occurs. By accurately predicting churn, businesses can optimize their marketing spend, reallocating resources towards retaining existing customers rather than solely acquiring new ones, resulting in cost savings and improved return on investment. Continuous refinement and updating of the churn prediction model based on new data and insights are essential, ensuring its accuracy and relevance in dynamic market conditions while driving sustainable customer retention strategies.

# *7. Conclusion*

In conclusion, this study highlights the importance of employing deep learning techniques for customer churn prediction. By preprocessing the data, developing an FNN model, tuning hyperparameters, and evaluating performance metrics, businesses can gain valuable insights into customer behavior and take proactive measures to mitigate churn. This approach contributes to enhancing customer retention and overall profitability.

# *8. Dataset Used*

<https://www.kaggle.com/datasets/shubh0799/churn-modelling>