Stroke Prediction
Group Jigglypuff

Gamze Aksu
171180005

Cansu Ayten
171180010

Ceren Umay Özten
181180060

ABSTRACT

In project stroke prediction dataset credentials are prepared for deep learning layered structure and we get the critic raw data for realizing train process of layered network nodes that changes weight and bias parameters. First we analyze the column information of stroke prediction dataset -gender, age, hypertension, heart disease, ever married, work type, residence type, average glucose level, BMI, smoking status and lastly stroke condition-, we set apart minority stroke positive conditioned patients to analyze columns qualifications critically identifying them. And we also set apart outliers of stroke prediction dataset that can cause wrong data depictions connected to training of neural data signals. Then we structure the our from scratch deep learning network layers and we assign own weight and bias parameters across backpropagation and forward propagation functioning mechanisms. Our deep learning network structure has two hidden layer and one input and output layers. And this deep learning network structure has training data that is consist of just train part, just 0.15 validation part and just 0.15 percentage test part. We train our neural network structure with training data and we experience parameter changings during data training in neural nodes. Our neural model structure can be controlled over inner workings and we can change parameters and biases.

Keywords

Deep Learning, Scratch, Interpretability, Stroke Prediction, ANN

1. INTRODUCTION

Stroke prediction dataset is consist of patients samples that demonstrate medical data of patients connected to patients stroke conditions and we want to predict stroke conditions of patients by patient medical data especially average glucose level, heart disease and hypertension. Prediction of stroke conditions of patients is critically important lifesaving situation that can save a patients life expectancy and can reduce life-threating, disability causing implications of stroke disease. Patient medical data and overall demographic data have potential to handle early diagnosis of stroke disease that can detect stroke condition of a patient or a patient group. Stroke prediction dataset's row samples follow a data pattern mechanism for findings of patients' stroke conditions as numerical, binary categorical data (1 numeric data for stroke positive condition and 0 numeric data for stroke negative condition). Therefore we construct a neural network structure that will simulate this data pattern mechanism to learn findings of patients' stroke conditions and predict an actual patient's or a patient group stroke condition. We search for a problem question that is "How we create a ANN neural network that can realize learning data pattern mechanism to extract patient stroke condition knowledge" and we analyse ANN neural network structure as together block with stroke prediction dataset. We also search for inner workings such as parameter metrics that are assigned by weight numerical - bias numerical, activation function -sigmoid-, total error function, parameter changings during ANN neural network functionality. Inner workings of ANN networks also encapsulates fundamental deep learning functions propagating data activation signals through ANN network layers and training raw features of stroke prediction dataset.

In our problem question of "How we create a ANN neural network that can realize learning data pattern mechanism to extract patient stroke condition knowledge", we experience deep learning training artificial forward propagation and back propagation assignment of weight numerical - bias numerical, activation function -sigmoid activation function-, parameter changings, behaviour of total error function integrated to back propagation training process and result of entire ANN neural network's inner workings through raw numerical stroke prediction dataset features. Especially during imitation of gradient descent process covers entire back propagation as a loop initiative updating weight and bias assignments according to a loss function derivative of connected weight and bias assignments in division factor. And during imitation of gradient descent process an activation function derivative contributes to optimization of whole weight and bias changings until gradient descent convergence. In gradient descent convergence end, ANN neural network reach its global minima over error score and we get ANN neural network actual learning gain that can be dedicated to real life practicality in our case prediction of real life patient stroke condition early diagnosis.

2. LITERATURE REVIEW

There are many academic literature papers that include prediction of patient stroke condition over passing stroke prediction dataset patient samples to an ANN neural network structure. We review academic literature domain and compare different results of ANN neural network structures, analyse the hidden nodes of supposed ANN neural network structures, training methods of supposed ANN neural network structures, partition ratios of stroke prediction dataset -train data, test data and validation data- and encoding of stroke prediction dataset for ANN neural network structure format. In academic literature review section we imply critic ANN neural network building layer nodes and stroke prediction dataset preparation. First academic literature review is Artificial Neural Network Application to the Stroke Prediction.[1]

In this proposed academic paper, an ANN structure is used in classification task for stroke prediction and supposed ANN structure is passed under condition of 1000 times cross validation. According to this paper, within scope of modern AI (Artificial Intelligence) applications, ANN structures are very suitable for medical applicability and ANN structures have successful results over cardiovascular disease prediction (CVD). Because of this situation, this paper apply ANN structure to build an artificial learning model on Kaggle's HealthCare Problem: Prediction Stroke Patients dataset to predict patients stroke conditions.

In preparation of Kaggle's HealthCare Problem : Prediction Stroke Patients dataset -including mainly the patient's basic physiological data, historical disease and living environment- for ANN structure, This data is divided into training data and testing data with 43400 and 18601 row records respectively. Just supposed training data is used for this paper. Before building ANN model, the dataset is randomly divided into three different subsets: 70% training set data, 15% test set data and 15% validation data. Then the dataset is encoded into data parameters that instruct dataset patients' sample column features such as id, gender, age, hypertension, heart disease, average glucose level, work type and categorical stroke condition [1].

Artificial Neural Network ANN structure is a computational heavy model that uses multiple artificial neuron nodes connected to simulate the neural structure and similar operation capacity of a real biological neural network. Neuron nodes transfer data signals from one type of nerve to another through same computational heavy connections. The widespread neural network ANN model is consisted of three parts: input layer, hidden layer and output layer. And first training process of a simple perceptron ANN structure is composed of neurons with adjustable synaptic weights and hard limiters (biases). Although this simple perceptron learning rule -regardless of the initial values of the weights- the correct, optimal weight vector values can be optimized in a generative, limited number of loops, this rule can only differentiate between linearly separable classes. Therefore this paper route us to activation functions that solve this differentiation problem [1].

ANN structure obtains correct results from a continuously data learning process of a loop cycle through adjusting weight parameters and optimizing errors to their minimal counterparts. ANN structure can provide solutions for complex, non-linear problems [1].

During creating of ANN structure model, this proposed academic paper is focused on training implementation of hidden neural nodes integrated to neural network ANN model and training algorithms is applied to hidden neural nodes for adding non-linearity to ANN structure training implementation. This paper applied several training algorithms to hidden neural nodes: Levenberg Marquardt Algorithm and Scaled Conjugate Gradient (SCG) Algorithm [1].

The Levenberg Marquardth (LM) algorithm is designed based on similarity to the quasi-Newton method, and in this similarity dimension the seconds order training speed can be achieved without calculation matrix production (Hessian matrix). Also when the connected performance function has the appropriate form of the sum of squares -a typical format value in the ANN training network-, the Hessian matrix can be approached instead of directly used for the ANN training network. Therefore Levenberg Marquardth (LM) algorithm uses approaching to Hessian matrix and calculation of relative gradients passing the training process [1].

Scaled Conjugate Gradient (SCG) algorithm, in standard backpropagation algorithm can adjust the weight in the largest negative value of gradient that is correspond to general steepest gradient descent rotation. Although in general sense the steepest gradient descent rotation is the fastest decline in performance capabilities, this decline type does not have the strictly the fastest convergence rate. Scaled Conjugate Gradient (SCG) algorithm have faster convergence rate than standard steepest gradient descent and this algorithm method has a performance function that is a more achieved direction. The SCG algorithm has a improvement over similar Hessian matrix approach of LM algorithm and total error function parameter. And SCG algorithm does not need necessarily to perform a linear search in each loop section by preventing time-consuming search practice of searching a direction calculation. Total computational cost of each loop section is reduced by SCG algorithm as an advantage [1].

The experimental results cover mainly number of hidden neural nodes and the difference in training algorithms. This academic paper's proposed ANN structure model has eight architectures, two different training algorithms and number of hidden neural nodes in a incremental order as 1, 2, 5, 10, 15, 20, 25 and 50, while supposed training algorithms are SCG and LM algorithms respectively. Proposed ANN structure model's each architecture, is run 1000 times with 10,000 epochs each time loop cycle. This proposed paper's experimental results suggest that both SCG and LM algorithms can achieve averaged classification accuracy rate of more than %98 by 1000-fold cross validation. In accordance of hidden neural node numbers, classification accuracy rate is not affected, whether it is 1 or 50 hidden neural node, classification accuracy rate is at above of %98 rate portion. However in 1 hidden neural node situation, the relative classification accuracies of SCG and LM algorithms training accuracy is weak. Also when the number of hidden neural nodes reach to 50, the SCG algorithm classification accuracy is more than 47% percentage and LM algorithm classification accuracy is up to %55 percentage. Especially findings of this proposed paper's study concludes that ANN neural structure models with fewer hidden neural nodes have generally weak performance however if number of hidden neural nodes are steadily increased, the training performance results can be enhanced while relative classification accuracy of testing and validation is not increased at all [1].

Prediction of Stroke using Deep Learning Model proposed paper, uses Heart Disease dataset that includes similar raw column features to our project's Stroke Prediction dataset -age, gender, blood pressure, type of chest pain, cigarettes, hypertension-. In this proposed paper's study, backpropagation algorithm is applied to main deep model as for multi-layered neural networks study. This proposed deep learning model -ANN neural network model- has a extractor mechanism for original dataset patient samples and in this proposed deep learning model, the backpropagation algorithm has also behavior of a multi-layered neural network with feedforwarding extension. Each hidden layer of multi-layered neural network can be considered as multiple perceptron outputs combined network structure. Linear combination of hidden units which are equivalent to the output perceptron unit [2].

The computation results of proposed deep learning model, shows Mean Squared Error (MSE) as 0.2596. This value indicates that the confidence has best performance for prediction of stroke condition. Also if proposed deep learning model is compared to generic Naive Bayes and Support Vector Machine classification techniques in terms of Mean Value and Standard Deviation scores, the mean values for prediction in deep learning is higher than these generic classification techniques. Deep learning training process also gives edge factors of weight values that sum some critic explanation for prediction of stroke condition parallel to deep learning training process [2].

3. PROJECT STUDY

In our Stroke Prediction project study, we used Stroke Prediction dataset and we construct a ANN neural network structure that is based on input layer-hidden layer-output standard ANN model proposal and we design over standard ANN model that is consisted of two hidden layers connected to one input layer and one output layer and we code the our ANN neural network structure design from scratch with its deep learning functional forward propagation and back propagation. However first we implement ready module version of our ANN neural network structure that is constructed from keras layer library.

3.1 Dataset

We used the Stroke Prediction Dataset, which we have accessed from the Kaggle platform, for the stroke prediction project. This dataset includes 11 features. These are id, gender, age, hypertension, heart disease, marriage status, work type, residence type, average glucose level, BMI (body mass index), and smoking status. id feature was removed from the dataset. Using these 10 features and using deep learning techniques, the probability of a person having a stroke will be estimated. There are 5110 records in the dataset. There are 201 null values in the BMI feature. These null values have been changed to be the median of the BMI feature. Then outlier values were found. Outlier values are also replaced with the median of the feature.

3.2 Model Building with Keras

In our Keras ANN structure we implement Dense layers that have sequentially 64-32-16-1 number of neural nodes from input layer -hidden layer- output layer. We assign “relu” activation function for each of hidden layers and assign “sigmoid” activation function as sigmoid probability distributions over binary classification of stroke condition. Then we compile our Keras model with “binary_crossentropy” loss and “Adam” optimization function and fit our Keras model with 50 epochs for loop cycle.

In Keras ANN structure, when we show the training results of Keras model we see that training accuracy result is 0.9492 and binary classification accuracy result is overall 0.96 with 0.96 precision, 1.0 recall and 0.98 f1-score for stroke negative condition. Also overall training loss is approximately 0.18. Our Training and Validation Loss graph and Reports for Accuracy are in Figure 1 and Figure 2.

3.3 Model Building from Scratch

In our from Scratch ANN structure imitating input layer- two hidden layer - output layer ANN model, first we define Scratch ANN structure parameters, learning rate, iterations loss array, sample size, layers and self X -training data- and y -training labels- variables. Then we initialize three different weight and bias parameters with layers variable numerical that are already assigned from Scratch NeuralNet ANN structure. Also we define activations function as ReLU and derivative of ReLU activation function, additional activation function as sigmoid. Lastly before we define forward propagation function, we define cross entropy loss and standard error loss functions.

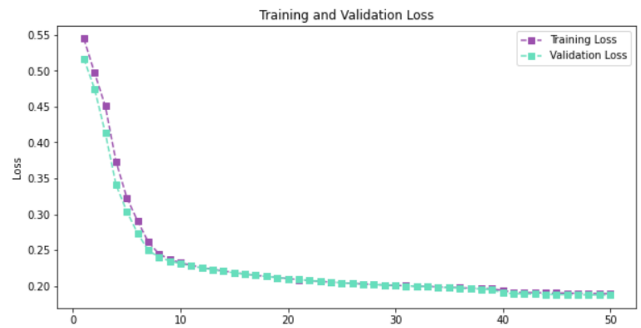

Figure 1: Training and Validation Loss

```
--------------------Accuracy Score---------------------
0.9621903520208605

---------------Classification Report -----------------
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       738
           1       0.00      0.00      0.00        29

    accuracy                           0.96       767
   macro avg       0.48      0.50      0.49       767
weighted avg       0.93      0.96      0.94       767


-----------Training Loss Accuracy---------------------
Training Loss: 0.18674415349960327
Training Accuracy: 0.9493706226348877

------------------Confusion Matrix--------------------
```

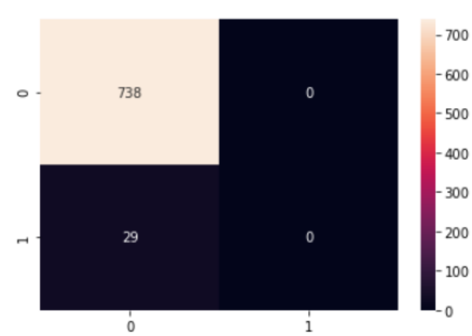

Figure 2: Reports for Accuracy

In forward propagation function, we assign our “neural node” variables with zeros for starting point of loop cycle iterations. We calculate dot multiplications of every self X data particles with each different weight parameters for each new assignment of neural node variables -three of neural node variables- and then we assign new value of neural node variables passed through activation functions to a new variable units. Therefore during loop iteration cycle, neural node variables and new variable units are always changing unit end of iteration minima as real prediction value candidate -and loss-.

In backpropagation function, we assign three different variables that are error function result of y data particle with forward propagation real prediction value candidate, dot multiplication of the error function result with third weight parameter transpose and lastly multiplication of the dot result with self-derivative of ReLU new neural node variables. Then we make dot multiplication of second weight parameter transpose with that self-derivative of ReLU multiplication and we multiply that second weight parameter transpose multiplication with self-derivative of ReLU new neural node variable. Finally we change inner weight and bias parameters through combination of these three different variables, X data transpose particles, new neural node variables and self-learning rate value. We multiply dot production of these combinations with self-learning value and we subtract the result from each three weight and bias parameters.

After implementation of forward propagation and backpropagation functions, we define fit, predict, accuracy and plot loss functions.

In fit function, loss and real prediction value candidate of forward propagation is assigned to loss array and backpropagation function. In predict function, prediction value is extracted from already trained three different weight parameter dot multiplications with X data particles and new neural node variables while these dot multiplications are addition with bias parameters. In accuracy function, accuracy value is extracted from self y label data with simple collection of comparison between y label data and prediction value as accuracy percentage. In plot loss function, connected loss array is plotted as a curve and the curve graphic of the loss array is demonstrated as a x and y legend axes.

When we finish entire construction of NeuralNet Scratch ANN structure, we fit NeuralNet to x_train and y_train tensor matrix array of Stroke Prediction dataset and we get NeuralNet prediction values from x validation data and we compare the prediction values to y validation data labels.

We get the classification accuracy performance of validation data and we plot the entire loss history of NeuralNet Scratch ANN structure. Therefore we get approximately %94 percentage. We get loss history is plotted in the Figure 3.

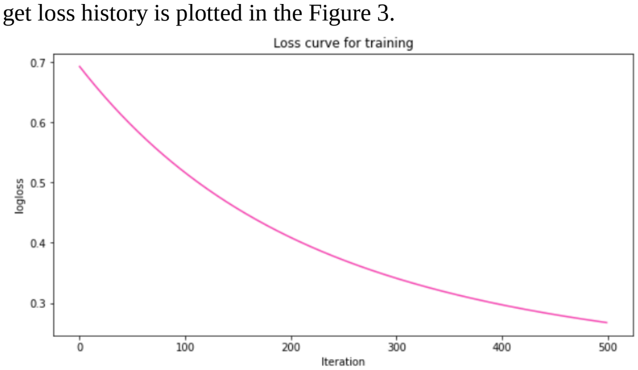

Figure 3: Loss Curve for Training

From loss history plot, we see that forward propagation loss is steadily declining over increasing of loop cycle iteration during training process. Also from classification accuracy of validation data, we see that Keras ANN structure has better classification accuracy performance than NeuralNet Scratch ANN structure with overall %96 rate.

3.4 Interpretability

In deep learning is important for finding "why deep learning ANN model acts for a certain decision points across its neural nodes" and we must realize interpretability for our relative deep learning ANN model. Since just a single metric, such as our Keras ANN structure classification accuracy is not adequate for fully description of most real-world tasks in our case stroke condition prediction. Interpretability is really helpful about why question of the prediction problem -why the prediction was made and probably pay for interpretability with a drop in predictive performance- instead of what question of the prediction problem -what is predicted-. In our approach we want to know that why the stroke condition prediction was made and how our NeuralNet Scratch ANN structure behaves against Stroke Prediction dataset data particles and reaches the final prediction value among various raw column features.

Because of this situation, we apply interpretability to our project study and we add evaluation of interpretability to our ANN structure implication. We interpret our ANN structure implication in accordance with Stroke Prediction dataset and we see whole picture of why questions through interpretability.

3.4.1 SHAP (SHapley Additive exPlanations)

Interpretability is used to get the models created with Deep Learning out of the Black Box state. Thus, the model's decision-making style can be easily understood by associating it with its outputs. Model interpretability makes deep learning or machine learning processes more effective in many fields such as medicine and finance. For this reason, many companies have started to use this method day by day and include it in their projects.

Interpretability can be examined under two headings, a global method and a local method. A global method is used to understand the decision-making process according to the general structure of the model. A local method is used to understand the decision-making process for a single instance of the model. There are many methods available for interpretability. It is the SHAP method used in the project. With SHAP Values, interpretability is enhanced by associating the model and data with outputs. SHAP contains both the local method and the global method.

We loaded the dataset and created our neural network. After the compile and fit processes, we made predictions. Finally, we plot the loss graph. After that, we moved on to interpretability methods. We have imported the SHAP library. We added interpretability to our model using both local and global methods.

3.4.1.1 SHAP Global Interpretation

3.4.1.1.1 ***Summary Plot***

In the summary plot, the impact of the features on the model is shown. The magnitude of these features is shown. Figure 4 shows the effect of the features in the dataset in the model. The Age feature contributed the most to the model. The Heart Disease feature has the least contribution to the model.

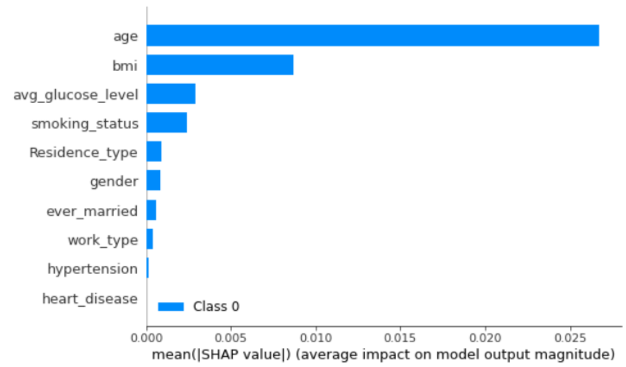

Figure 4: Summary Plot

3.4.1.2 SHAP Local Interpretation

3.4.1.2.1 Force Plot

The force graph is used to see the effect of each feature on the prediction for a given sample. In this graph, positive SHAP values appear on the left and negative values appear on the right. Base value is the model's average estimate over the training set. Output value is the model's estimate and is highlighted.

Figure 5: Force Plot

3.4.1.2.2 Decision Plot

The force graph and the decision graph are effective in explaining the prediction of the model. Decision charts are true representations of SHAP values, making them easy to interpret. Features are sorted in decreasing order of importance. Decision graphs are more effective than force graphs when it comes to many important features. The vertical line in the decision graph represents the base value of the model. The X axis shows the outputs of the model while the Y axis shows the features of the model. SHAP values of each feature are shown next to the line.

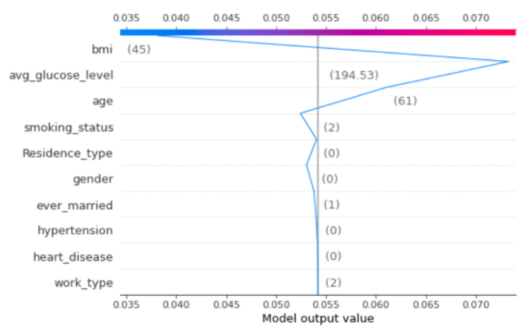

Figure 6: Decision Plot

3.4.1.2.3 Waterfall Plot

Waterfall Plot shows approximately the same information as Force Plot. The SHAP values of each attribute are visualized. At the bottom, it starts from the base value of the model. The positive or negative contributions of the features are then shown in each row. Positive values are shown in red and negative values in blue.

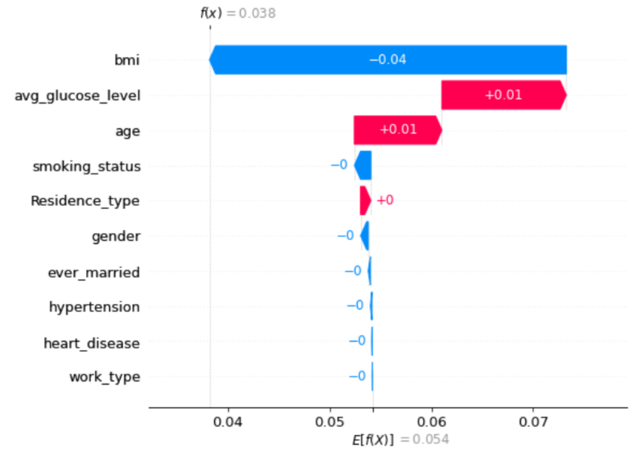

Figure 7: Waterfall Plot

4. EXPERIMENTS

We observed what kind of output the model created within the scope of the project gives when the hyper parameters are changed. First, the learning rate values, epoch and initial weights were changed. Learning rate values changed to 0.1, 0.01, 0.001, and 0.0001. Epoch values were tried as 50 and 100. The initial values of the weights were first chosen randomly over the normal distribution, and secondly all weights were chosen zero. Then the weights were assigned as initial values. Outcomes were observed at the end of these experiments. The best output is obtained in 50 epochs trained model with zero weights and a learning rate of 0.1.

It is seen that the age feature contributes the most to the model in the interpretability summary plot. Age feature affects the model a lot as it has much more contribution than others. Therefore, the age feature was removed and the effect of other features on the model was examined. After removing the age feature, it is seen that it is the average glucose level feature that contributes the most to the model. The average glucose level feature affects the model positively. After that BMI feature comes second. The BMI feature affects the model negatively.

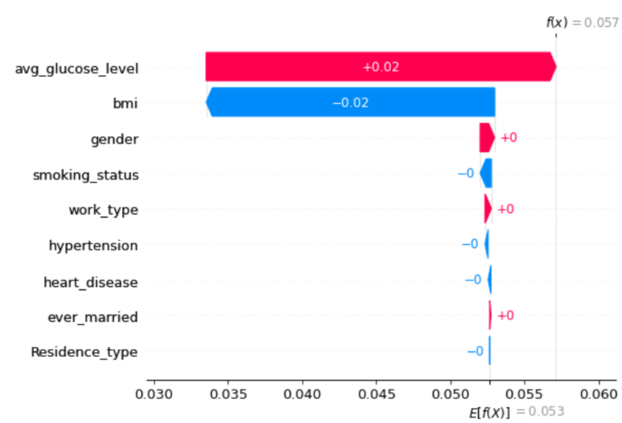

Figure 8: Waterfall plot after age removed

It is seen that the feature with the least contribution in the summary plot of the model is the heart disease feature. Therefore, the heart disease feature was removed and the effect of other features on the model was observed. After removing the heart disease feature, the age feature again contributes by far the most. Next, the BMI feature comes second. According to the waterfall chart, the BMI feature affects the feature negatively. Age feature affects the feature positively.

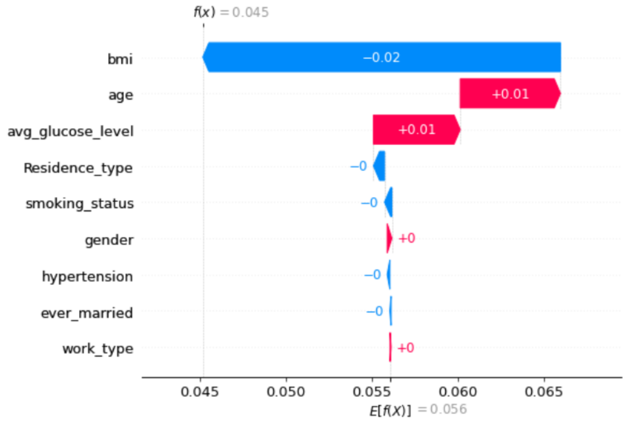

Figure 9: Waterfall plot after heart disease removed

5. DISCUSSION

Since there are not many records in the dataset we use, even if very good results are obtained with a complex model, the model may be overfitting. Therefore, the number of dense in the model is kept low. Compared to this, Peng et al. [1] used the number of dense as 1 and 50 in their studies. When the number of dense was determined as 50, they obtained 98% accuracy.

Chantamit-o-pas and Goyal (2017) used a heart disease dataset in their work. There are 76 features in this dataset. They made a stroke prediction with these values. In this study, in addition to the deep learning model, they also used the Naïve Bayes and Support Vector Machine technique. As a result, they got the best output with the deep learning model.

6. CONCLUSION AND RECOMMENDATION

Within the scope of the project, stroke prediction was made using the Stroke Prediction dataset. From the Stroke prediction dataset, 10 features - gender, age, hypertension, heart disease, ever married, work type, residence type, average glucose level, BMI, and smoking status- were used. Using these features, it is tried to predict whether someone will have a stroke. There are 5110 records in the dataset.

After applying the preprocessing to the data, the network was created. By adding interpretability to the created model, the contributions of the features in the model were observed.

After the outliers were removed in the dataset used within the scope of the project, 5109 records remained. This number is not sufficient for a complex model. Therefore, retraining can be done with more records. Similarly, the dataset used is unbalanced. Negative stroke values in the dataset are much higher than positive ones. Therefore, retraining can be done with a balanced dataset or the dataset can be balanced by using data augmentation methods in the preprocessing stage. In this way, better results can be obtained.

7. REFERENCES

[1] C. -C. Peng, S. -H. Wang, S. -J. Liu, Y. -K. Yang and B. -H. Liao, "Artificial Neural Network Application to the Stroke Prediction," *2020 IEEE 2nd Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS)*, 2020, pp. 130-133.

[2] Chantamit-o-pas, P., Goyal, M. (2017). Prediction of Stroke Using Deep Learning Model. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, ES. (eds) Neural Information Processing. ICONIP 2017. Lecture Notes in Computer Science(), vol 10638. Springer, Cham.