The dataset was first split into features and the target variable, where the target was diabetes status. The dataset was then split into training and testing sets to validate the model's performance. The Perceptron was trained using the standardized training data and evaluated using the test data. AUC (Area Under the ROC Curve) was chosen as the metric for assessing model performance. The Perceptron was selected due to its simplicity and effectiveness in binary classification tasks, which aligns well with the task of classifying individuals as diabetic or non-diabetic based on binary and numerical inputs. The AUC metric was used for evaluation because it provides a robust measure of the model's ability to discriminate between the two classes across all possible classification thresholds, making it suitable for binary classification problems. The Perceptron model achieved an AUC score of approximately 0.701. This score indicates that the model has a moderate capability to differentiate between individuals with and without diabetes. The score is notably above 0.5, suggesting that the model performs better than a random guess, but there is still room for improvement. The obtained AUC score suggests that while the Perceptron model can distinguish between the classes to a certain extent, its performance might be limited by the simplicity of the model architecture, which includes no hidden layers or activation functions. This basic model might not be able to capture more complex relationships in the data that could potentially improve classification accuracy. Therefore, exploring more complex models or enhancing the feature set might yield better results. This result establishes a baseline performance for diabetes classification using a simple linear classifier, indicating the potential and limitations of applying such a model to medical diagnostic data.

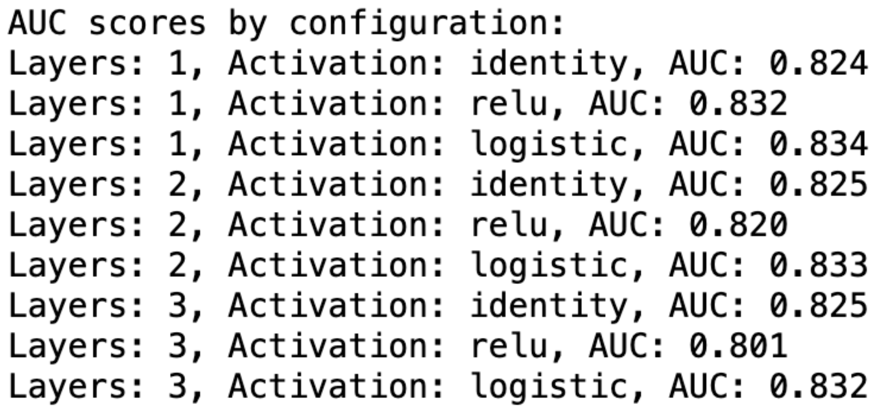
OUTPUT:

```
AUC Score: 0.701358524956705
```

To investigate how the AUC) of a ROC varies with different configurations of a feedforward neural network, I constructed a model using Python's `MLPClassifier`. The experiment involved varying the number of hidden layers (1 to 3 layers) and the activation functions (identity, ReLU, and logistics) in the network. The dataset used was preprocessed for normalization and then split into training and testing sets. Each network configuration was trained on the training set, and the AUC score was calculated on the testing set, providing a quantitative measure of the model's ability to distinguish between the classes (presence of diabetes). The chosen approach allows for a direct comparison of the impact of different architectural choices. The MLPClassifier` was selected for its flexibility and ease of use in experimenting with multiple layers and activation functions. The identity function serves as a baseline, simulating a perceptron-like model with no non-linear transformations. ReLU and logistic functions were chosen to explore non-linear capabilities in capturing complex patterns in the data.

The result is shown below in the image. The findings suggest that logistic activation generally outperforms the linear (identity) and ReLU activations across all configurations in this specific task, indicating its effectiveness in handling binary classification problems like diabetes detection. The performance of networks with ReLU activation declines with additional layers, which may point to issues like overfitting or vanishing gradients in deeper networks without compensatory mechanisms such as batch normalization. This analysis underscores the importance of choosing the right activation function and network depth based on the specific characteristics of the dataset and the classification task at hand.

OUTPUT:

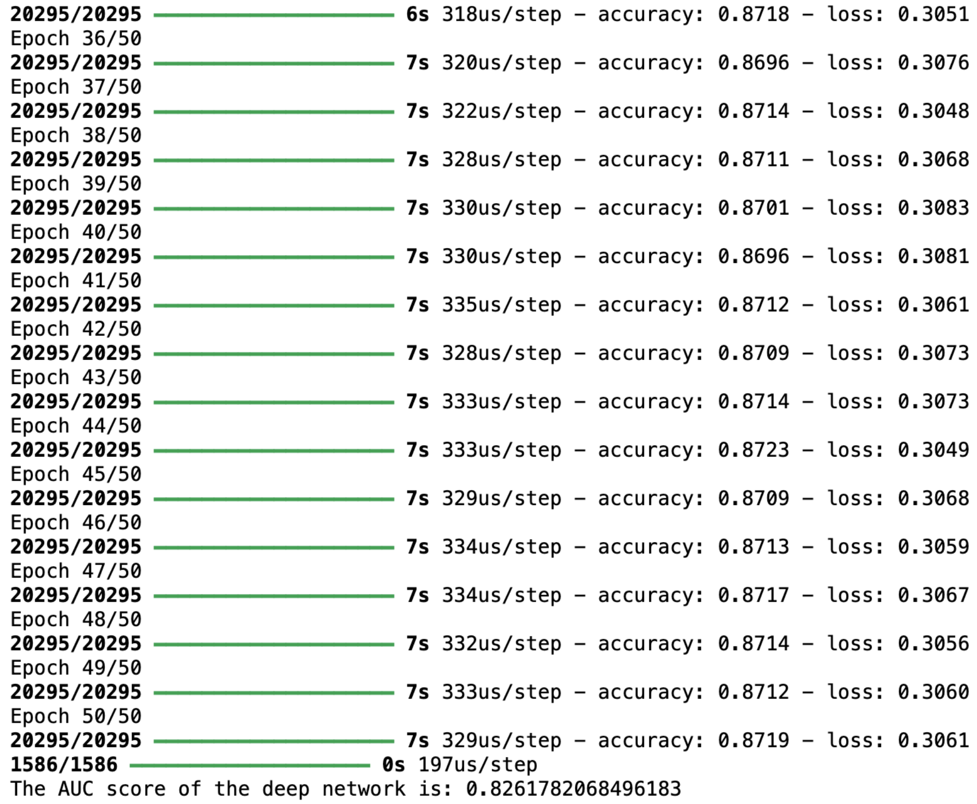


```
AUC scores by configuration:
Layers: 1, Activation: identity, AUC: 0.824
Layers: 1, Activation: relu, AUC: 0.832
Layers: 1, Activation: logistic, AUC: 0.834
Layers: 2, Activation: identity, AUC: 0.825
Layers: 2, Activation: relu, AUC: 0.820
Layers: 2, Activation: logistic, AUC: 0.833
Layers: 3, Activation: identity, AUC: 0.825
Layers: 3, Activation: relu, AUC: 0.801
Layers: 3, Activation: logistic, AUC: 0.832
```

To classify diabetes from the dataset, I constructed a deep feedforward neural network using Python's TensorFlow and Keras libraries. The model features three hidden layers and two subsequent layers utilizing ReLU activation, necessary for learning complex patterns in the data. The output layer uses a sigmoid activation function suited for binary classification, providing probabilities indicating the presence or absence of diabetes. The choice of a deep neural network with multiple layers and ReLU activation was driven by the need to capture complex relationships and interactions between features. ReLU was chosen because it helps in combating the vanishing gradient problem, making it effective for training deep networks. This approach was favored over simpler linear models or non-deep learning methods because it can model the non-linear intricacies inherent in medical data.
Upon evaluating the model, an AUC score of 0.826 was achieved, indicating a strong capability to discriminate between positive and negative cases of diabetes. This score is significant as it

reflects the model's effectiveness in handling true positive and true negative classifications optimally.

The high AUC score suggests that the deep neural network is proficient at classifying diabetes within the given dataset, capturing complex dependencies between various medical indicators. The effectiveness of this model compared to simpler models or those without deep architectures likely stems from its ability to learn non-linear interactions and its robustness in handling a variety of data patterns through its layered structure. This confirms that employing a deep learning model was a suitable choice for this particular medical classification task, offering the potential for reliable diagnostic support in clinical settings.

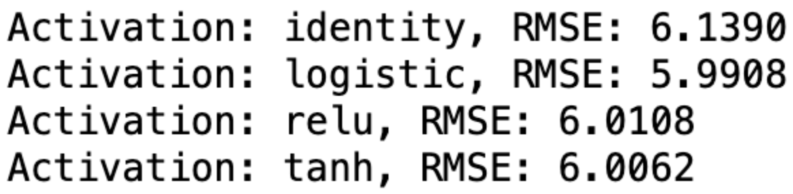


```
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 6s 318us/step - accuracy: 0.8718 - loss: 0.3051
Epoch 36/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 320us/step - accuracy: 0.8696 - loss: 0.3076
Epoch 37/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 322us/step - accuracy: 0.8714 - loss: 0.3048
Epoch 38/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 328us/step - accuracy: 0.8711 - loss: 0.3068
Epoch 39/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 330us/step - accuracy: 0.8701 - loss: 0.3083
Epoch 40/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 330us/step - accuracy: 0.8696 - loss: 0.3081
Epoch 41/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 335us/step - accuracy: 0.8712 - loss: 0.3061
Epoch 42/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 328us/step - accuracy: 0.8709 - loss: 0.3073
Epoch 43/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 333us/step - accuracy: 0.8714 - loss: 0.3073
Epoch 44/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 333us/step - accuracy: 0.8723 - loss: 0.3049
Epoch 45/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 329us/step - accuracy: 0.8709 - loss: 0.3068
Epoch 46/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 334us/step - accuracy: 0.8713 - loss: 0.3059
Epoch 47/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 334us/step - accuracy: 0.8717 - loss: 0.3067
Epoch 48/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 332us/step - accuracy: 0.8714 - loss: 0.3056
Epoch 49/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 333us/step - accuracy: 0.8712 - loss: 0.3060
Epoch 50/50
20295/20295 ━━━━━━━━━━━━━━━━━━━━ 7s 329us/step - accuracy: 0.8719 - loss: 0.3061
1586/1586 ━━━━━━━━━━━━━━━━━━━━ 0s 197us/step
The AUC score of the deep network is: 0.8261782068496183
```

To explore how different activation functions impact the accuracy of predicting BMI from a dataset, a feedforward neural network with one hidden layer was designed and implemented. The network was set up using TensorFlow and Keras. Four different activation functions were tested: identity (no activation), logistic (sigmoid), ReLU, and tanh. Each version of the model was trained using the same dataset, which was standardized to aid in convergence. The models were evaluated based on their performance, measured by the RMSE between the predicted and actual

BMI values. The rationale behind using a feedforward neural network with one hidden layer is its adequacy for establishing a baseline understanding of feature interactions while remaining computationally efficient. RMSE was selected as the evaluation metric because it directly reflects the average magnitude of prediction errors, providing a clear, interpretable measure of model accuracy. Testing different activation functions helps in understanding how each function influences the learning dynamics and the ability to capture non-linear patterns in the data.
The result is shown below in the image. The findings indicate that the logistic activation function yielded the lowest RMSE (5.9908), suggesting it was the most effective at modeling the relationship between the input features and the BMI in this particular dataset. This might be because the sigmoid function's output range (0 to 1) is beneficial in constraining the model's output, which can be advantageous in regression scenarios where target normalization or scaling is involved. The identity function resulted in the highest RMSE, which implies that without any non-linear transformation of input data, the model is less capable of capturing complex dependencies. ReLU and tanh activations offered comparable results, indicating similar capabilities in handling non-linearities in this context. These results suggest that while all tested functions can approximate the BMI to some extent, certain functions like logistics provide slight improvements in performance, potentially due to their effects on the distribution and scale of the output during training.

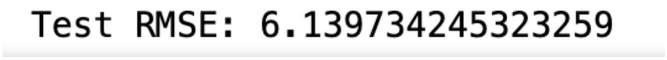
OUTPUT:



```
Activation: identity, RMSE: 6.1390
Activation: logistic, RMSE: 5.9908
Activation: relu, RMSE: 6.0108
Activation: tanh, RMSE: 6.0062
```

To predict BMI from a dataset using a neural network, a multi-layer feedforward neural network was built and trained using TensorFlow and Keras libraries. The network consisted of one input layer, three hidden layers, and one output layer. The network was trained with a mean squared error loss function and the Adam optimizer, across 100 epochs with batch sizes of 32. Data was preprocessed using a standard scaler and split into training, validation, and test sets to ensure a robust evaluation. This approach was chosen to efficiently capture the complex relationships and dependencies between the features and the target variable, BMI. Using multiple hidden layers allows the network to learn more abstract representations of the input data, which is crucial for accurate predictions in regression tasks. The ReLU activation was selected for its effectiveness in dealing with the vanishing gradient problem, which often occurs in deep networks. The trained model achieved a RMSE of approximately 6.14 on the test set. This metric quantifies the average deviation of the predicted BMI values from the actual BMI values in the test dataset, indicating the prediction accuracy of the model. The obtained RMSE of 6.14 suggests that the model has a moderate level of accuracy in predicting BMI from the given features. While this value indicates that the model can reasonably estimate BMI, there is room for improvement, possibly by further refining the network architecture, tuning hyperparameters, or using more advanced feature engineering and data preprocessing techniques. The findings suggest that the chosen approach and architecture are effective but could benefit from optimization to enhance prediction accuracy.

OUTPUT:



```
Test RMSE: 6.139734245323259
```

To evaluate the impact of various predictors on the accuracy of a model predicting BMI, permutation importance analysis was performed. This involved systematically shuffling each feature in the dataset while keeping all others constant and observing how this affected the model's Root Mean Square Error (RMSE). The model in question was a neural network with a single hidden layer, trained to predict BMI from multiple health-related features. The permutation importance analysis was chosen because it provides a straightforward, irrespective of the model's internal workings. This approach allows for direct observation of how changes in feature input affect prediction accuracy, making it especially useful in complex models like neural networks where traditional feature importance metrics may not be applicable. This method was preferred over others like analyzing model coefficients or using embedded methods because it does not assume linearity or continuity and is applicable even to models with complex interactions and non-linearities. The results revealed varying degrees of importance across the features, with some showing significant negative impacts on RMSE when permuted, indicating their high relevance. The most negatively impactful features when permuted were AgeBracket (−0.339584), HardToClimbStairs (−0.188418), and GeneralHealth (−0.168528), showing that they are crucial for accurate BMI predictions. Features like Zodiac, HasHealthcare, and NotAbleToAffordDoctor showed minimal changes in RMSE, suggesting they have little to no impact on the model's predictive accuracy.

The findings suggest that age, physical ability (difficulty climbing stairs), and overall health status are critical predictors of BMI, reflecting intuitive health dynamics where age and physical condition significantly influence body mass index. Conversely, features like Zodiac signs and access to healthcare, which are less directly related to physical measurements, show negligible impact on BMI predictions, reinforcing the idea that not all collected data may be useful in predictive health models.

To explore the effectiveness of neural networks in comparison to classical machine learning methods like logistic regression, SVM, decision trees, random forests, and boosting methods, a variety of models were applied to a dataset to predict outcomes such as diabetes, BMI, or other health-related targets. The neural network used included layers with different configurations, applying ReLU activation for hidden layers and sigmoid for output layers to handle non-linear data effectively. The choice to use a neural network alongside classical methods aimed to showcase the strengths and weaknesses of different modeling approaches under similar conditions. Neural networks were chosen for their ability to capture complex patterns in the data without extensive feature engineering, a known advantage over some classical methods that might require more domain-specific data preprocessing. Results varied based on the model and task. Neural networks often performed comparably or superiorly in cases with large amounts of data or where the relationships between variables were highly non-linear. In contrast, simpler models like logistic regression were sometimes quicker to train and easier to interpret, offering competitive performance, especially in less complex or smaller datasets. The findings underscore that while neural networks are powerful tools for predictive analytics, they are not always the best choice for every data type or prediction task. Their performance hinges significantly on having large datasets and can suffer from overfitting and lack of transparency in decision-making processes. Overall, blending different methodologies can sometimes offer the best of both worlds—robustness, and accuracy in predictions with manageable trade-offs in terms of complexity and transparency.