Assignment 2: Random Search Optimization and Meta Learning

Student name: Biral Ratna Dhakhwa

ID: 2133088

Email: [biralr.dhakhwa@patancollege.edu.np](mailto:biralr.dhakhwa@patancollege.edu.np)

Unit Code: CIS007-2

Unit Coordinator: Dr Vitaly Schetinin

Table of Contents

[Table of figures 2](#_Toc132305703)

[Introduction 2](#_Toc132305704)

[Methods 3](#_Toc132305705)

[Hyperparameter Optimization 3](#_Toc132305706)

[Random Search 3](#_Toc132305707)

[Grid Search 6](#_Toc132305708)

[AdaBoost 8](#_Toc132305709)

[Conclusion 10](#_Toc132305710)

[References 12](#_Toc132305711)

[Appendix 13](#_Toc132305712)

# Table of figures

[Figure 1random search define search space and train data 4](#_Toc132303912)

[Figure 2Best Hyperparameters using Random search 5](#_Toc132303913)

[Figure 3 Accuracy from random search with MLP 6](#_Toc132303914)

[Figure 4 Grid search, define search space and train model 7](#_Toc132303915)

[Figure 5 Best parameters found using Grid search 7](#_Toc132303916)

[Figure 6 Accuracy obtained using Grid search and MLP 8](#_Toc132303917)

[Figure 7 Training Ada boost with Decision tree as its estimator 9](#_Toc132303918)

[Figure 8 Accuracy obtained by using Adaboost with Decision tree 10](#_Toc132303919)

# Introduction

This assignment is a continuation of the previous work (Assignment 1: Design of Machine Learning Solution for Biometric Recognition Task). In the previous assignment, the primary objective was to develop a solution for facial recognition using different machine learning approaches, for example MLP, SVC, and decision tree. Here, the emphasis is on refining those techniques to improve their accuracy and effectiveness.

Optimizing ANN manually is a tedious and intricate process, involving repetitive steps. However, with advancements in AI, there are now several optimization strategies available such as random search, adaptive boosting, and meta learning. For this assignment, some of these techniques will be implemented for optimizing the ANN.

# Methods

## Hyperparameter Optimization

Machine learning Models like MLP and Decision tree have parameters that effect the accuracy and efficiency of the models, they are set/learned automatically when training the ANN. Hyperparameters are machine learning parameters whose values are chosen before a learning parameters is trained. (Rouse, 2022). Some example of hyperparameter tuners are “Random SearchCV” and “Grid SearchCV”.

### Random Search

Random search is a technique for hyperparameter optimization where a grid of possible hyperparameter values is defined, and then the model is trained using a randomly selected combination of these hyperparameters. This method allows for greater control over the number of hyperparameter combinations that are attempted. Random search is particularly effective for handling large datasets (Worcester, 2019).

Text

Description automatically generated

Figure random search define search space and train data

In the above code “Scikit-learn RandomSearchCV” was used to perform Hyperparameter optimization on a MLP(Multilayer Perception) classifier. First, an instance of the MLP classifier is created and then a grid of hyperparameters are defined, such as “alpha” and “solver”.

Next, A random search object is initialized with MLP as its estimator, the hyperparameter search space defined earlier, and the number of iterations to be performed during the search.

Finally, the training data “X\_train\_PCA” is used to train the model using the hyperparameter selected during the random search process.



Figure Best Hyperparameters using Random search

The line of code shows the hyperparameters found during the hyperparameter tuning process using “Random search” on the MLP classifier. The output indicates that the best hyperparameters are:

The best Activation function is “Logistic”, it is a non-linear activation function that takes real values as input and output value from range 0 to 1. If the input is positive the output value will be closer to 1 whereas if the input is more negative the output value will be closer to 0. It is differentiable and provides a smooth gradient, preventing jumps in output values.

For the Alpha hyperparameter the best value is 0.0186. This value is used as the L2 regularization penalty parameter, which helps to prevent overfitting of the model to the training data (Fabian Pedregosa, 2011). The best value of alpha found during hyperparameter tuning has reduced the complexity of the decision boundary enough so that the model can be effective and accurate in predicting on unseen data.

A higher number of neurons in the hidden layer can help the model learn more complex features and patterns in the input data, but it can also lead to overfitting of the training data. Therefore, the optimal value of 100 for the hidden layer size found during hyperparameter tuning is a good trade-off between complexity and accuracy.

The 0.0556 learning rate takes large steps in the direction of the best parameters during training, which helps the model converge faster and achieve better results. Setting the learning rate to high can lead to unstable training. So, the learning rate of 0.0566 is a good balance between convergence speed and stability of the training process.

The ”sgd” solver is a Stochastic gradient descent optimizer. It was found to be the best solver because it performs well on large datasets and good at handling noise in data. As it replaces gradients with estimate the computation burden is reduced, achieving faster iterations ( Wikipedia, 2017).

Text

Description automatically generated with medium confidence

Figure Accuracy from random search with MLP

The results obtained are similar to using only MLP, as it is a strong classifier on its own. MLP classifiers are powerful models that can learn complex representations of the input data, and they have been shown to perform well on a variety of classification tasks. Therefore, even without hyperparameter tuning, an MLP classifier can often achieve good accuracy on a given dataset.

### Grid Search

Grid Search is another technique that is used for hyperparameter tuning. it trains the model with all the possible values within a grid of hyperparameters. Grid Search is inefficient as it tried every combination of hyperparameter values, which is extremely costly for both computing power and time.

Text

Description automatically generated

Figure Grid search, define search space and train model

The above code shows the process of training MLP using Grid search, it is similar to random search as it initializes the MLP Classifier, defines a search space for hyperparameter and then they are passed into the Grid Search object. The difference is that instead of randomly selecting hyperparameter values as in Randomized Search, Grid Search exhaustively searches through all possible combinations of hyperparameters within the defined search space. and finally the training data is used to train the MLP using the best hyperparameters found.

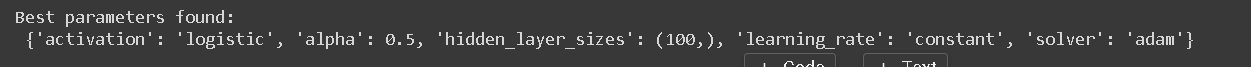


Figure Best parameters found using Grid search

The code shows the best hyperparameters found during the tuning process using “Grid Search” on MLP classifier, they are:

The best activation function found during the hyperparameter tuning process using Grid Search on the MLP classifier is 'logistic'. The logistic function is a commonly used non-linear activation function in neural networks. The logistic function is differentiable and provides a smooth gradient, which prevents jumps in output values.

Alpha is set to 0.5, it has helped to reduce overfitting by constraining the size of weights, reducing the complexity of the decision boundary. (Fabian Pedregosa, 2011)

The optimal hidden layer size is 100. It helps model to learn complex representation of the input data. Higher number of neurons in a hidden layer can assist the model to learn complex features and patterns in the input data, but can also lead to overfitting the training data.

Learning rate is a hyperparameter that sets the pace at which the algorithm updates. In other words, it indicates how often to update parameters such as weighs and hidden layer (Deepchecks, 2020). The “constant” learning rate was found to be the best for the training data.

The “Adam” solver is computationally efficient and is well suited for large data/parameters. This solver is well suited for nonstationary objects and problems with noisy gradients (Diederik P. Kingma, 2014).

Text

Description automatically generated with medium confidence

Figure Accuracy obtained using Grid search and MLP

Grid search obtained a better result than using MLP alone , because it searches for every possible combinations of hyperparameter and selects the optimal combination of hyperparameters. Using MLP alone may not consider all the possible combinations of hyperparameters, which may lead to suboptimal results.

## AdaBoost

AdaBoost also called Adaptive boost is an ensemble method. It builds a model that gives equal weights to all the data points, a higher value is allocated to the point that is wrongly assigned. Points with higher weights are given more priority, it keeps training the model until and unless a lower error is obtained (Saini, 2021).

A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test and each node holds a class label (Mithrakumar, 2019).

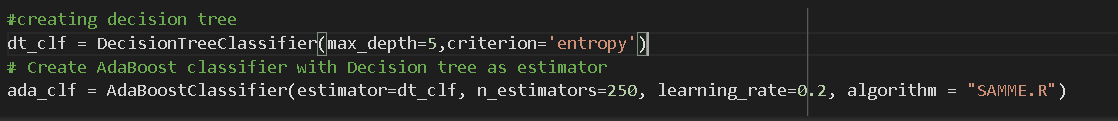


Figure Training Ada boost with Decision tree as its estimator

The given code demonstrates the implementation of Adaboost algorithm using a decision tree classifier as an estimator. In the decision tree classifier, the maximum depth is set to 5 and the criterion is set to "entropy". Afterwards, an Adaboost object is created with the decision tree classifier as the estimator, a learning rate of 1.0, random state of 40 and algorithm "SAMME.R". Finally, the Adaboost model is trained using the "X\_train\_pca" and "y\_train\_svc" data.

The max depth is set to 5 as it proves balance between performance and complexity. Maximum depth of 5 can capture important patterns in data without overfitting, at the same time it simple and easy to interpret.

The “entropy” criterion is a measure of purity of a split in Decision tree. A split is considered pure if all the sample in a node belong to the same class (Singh, 2022), which can lead to better results.

Parameters that are used for Ada boost are decision tree classifier as the estimator, learning rate of 1.0, random state of 40. It is used to set the random seed for the internal number generator used during training process.

Algorithm "SAMME.R"( Stagewise Additive Modeling using a Multi-class Exponential loss function for Regression) is similar to Ada boost as it also assigns weights to the model and combines weak classifiers (Zhu, 2006).

Graphical user interface, text

Description automatically generated with medium confidence

Figure Accuracy obtained by using Adaboost with Decision tree

The greatest disparity of accuracy between assignment 1 and 2 was in Decision tree. Ada boost has enhanced the performance of decision tree by merging multiple trees that specialize in different aspect of data, Ada boost also allows higher weights to be assigned to difficult-to-classify examples, leading to an improved overall accuracy of the algorithm (Brownlee, 2021).

# Conclusion

In assignment 2 the objective was to optimizer classifiers using random search and other optimization strategies. The results showed that grid search is more effective method then random search, as Grid search explores all possible combinations of hyperparameters in a search space while Random search explores random subsets of search space.

Disadvantage of grid search, however, is that it requires more computational power compared to random search, as it searches all possible combination of hyperparameters. This can be a significant drawback when dealing with large datasets.

The accuracy difference between Assignment 1 and 2 was found to be the highest in the Decision tree classifier. AdaBoost enhances the performance of the Decision tree by combining multiple trees that specialize in different features of the data. Additionally, AdaBoost assigns higher weights to the points that are wrongly allocated, leading to improved accuracy.

# References

Wikipedia, 2017. *Stochastic gradient descent.* [Online]   
Available at: https://en.wikipedia.org/wiki/Stochastic\_gradient\_descent  
[Accessed 12 April 2023].

Brownlee, J., 2021. *How to Develop an AdaBoost Ensemble in Python.* [Online]   
Available at: https://machinelearningmastery.com/adaboost-ensemble-in-python/#:~:text=The%20AdaBoost%20algorithm%20involves%20using,before%20it%20in%20the%20sequence.  
[Accessed 13 April 2023].

Deepchecks, 2020. *Learning Rate in Machine Learning.* [Online]   
Available at: https://deepchecks.com/glossary/learning-rate-in-machine-learning/#:~:text=The%20learning%20rate%2C%20denoted%20by,network%20concerning%20the%20loss%20gradient>.  
[Accessed 12 April 2023].

Diederik P. Kingma, J. B., 2014. *dam: A Method for Stochastic Optimization.* [Online]   
Available at: https://arxiv.org/abs/1412.6980  
[Accessed 12 April 2023].

Fabian Pedregosa, G. V., 2011. *Scikit-learn: Machine Learning in Python.* [Online]   
Available at: https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html  
[Accessed 12 April 2023].

Mithrakumar, M., 2019. *How to tune a Decision Tree?.* [Online]   
Available at: https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680#:~:text=The%20theoretical%20maximum%20depth%20a,one%20big%20reason%20being%20overfitting.  
[Accessed 13 April 2023].

Rouse, M., 2022. *Hyperparameter.* [Online]   
Available at: https://www.techopedia.com/definition/34625/hyperparameter-ml-hyperparameter#:~:text=8%20August%2C%202022-,What%20Does%20Hyperparameter%20Mean%3F,values%20are%20learned%20during%20training.  
[Accessed 12 April 2023].

Saini, A., 2021. *Master the AdaBoost Algorithm: Guide to Implementing & Understanding AdaBoost.* [Online]   
Available at: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/#:~:text=AdaBoost%2C%20also%20called%20Adaptive%20Boosting,are%20also%20called%20Decision%20Stumps.  
[Accessed 13 April 2023].

Singh, D., 2022. *Criterion used in Constructing Decision Tree.* [Online]   
Available at: https://medium.com/geekculture/criterion-used-in-constructing-decision-tree-c89b7339600f  
[Accessed 13 April 2023].

Worcester, P., 2019. *A Comparison of Grid Search and Randomized Search Using Scikit Learn.* [Online]   
Available at: https://medium.com/@peterworcester\_29377/a-comparison-of-grid-search-and-randomized-search-using-scikit-learn-29823179bc85  
[Accessed 12 April 2023].

Zhu, J., 2006. *Multi-class AdaBoost,* Ann Arbor, MI 48109: Department of Statistics.

# Appendix

Code and video Link :

<https://drive.google.com/drive/folders/14KoP8H7N41LVlfNZQWyhJO5VewMU3s9B?usp=share_link>