Using Neural Network ensemble method to enhance traditional Machine Learning Classification accuracy

PEIQIN LI

Content:

1 Introduction:

Classification is a very common task in nowadays machine learning application. With more and more complex classification models, like Random Forest, Gradient Boosting Decision Tree (GDBT) and Deep neural network, the accuracy of classification result is also improving. The mainstream machine algorithms regard training data as a whole, focusing on building a powerful enough model to handle all hidden information. However, in realistic different instances can be clustered into different subsets and the data structures in different subsets may not be the same.

For example, if we want to predict the people's income, the income distributions of Male and Female should be different, and the income distributions of people from developing and developed counties are also different. One natural question we should ask is that whether our model is complex enough to fully take these special data structures into consideration. And moreover, if the model cannot 'perfectly' analysis all special data structures (which is always the cases), how can we improve the existing algorithm.

There I introduce a supervise ensemble method that aims to solve the question mentioned above. By training models on hole trainset and its disjoint subsets, and using neural network (including linearly Connected Neural Network and Convolutional Neural Network) to aggregate the results of these algorithms, more data structure information among various subsets can be learned.

In addition, to give the data structure of these training subsets more variety, a feature selection method using genetic algorithm (GA) is applied for each subset. In Part 4, I will show how the genetic algorithm is applied in each subset and how the accuracy is improved when we drop some features in each subset.

Finally, to prove the generality of this ensemble method, I construct several empirical test using data sets from UCI repository. These results shows that by adding this ensemble method, the accuracy of classification is higher than the accuracy of pure traditional Machine Learning technique.

2 Motivation:

2.1 Dataset source:

To illustrate how the Neural Network can be applied in this ensemble method, there I choose a public dataset "Adult Census Income" in Kaggle *[15]* to construct numerical experiment. This data set, which extracted from 1994 Census bureau database, can construct typical binary classification task. The prediction task is to determine

whether a person makes over $50K a year. The number of categorical variables is 8 and the number of continuous variables is 6. There are totally 32561 instances, and I will choose 80% of them as training set and the rest 20% of them as testing set in later experiment.

8 categorical variables are:

1) Work class
2) Education
3) Marital Status
4) Occupation
5) Relationship
6) Race
7) Gender
8) Native Country

And Here I choose Gender (Male, Female) as the criterion to divide the dataset into 2 subsets.

2.2 Data structure analysis:

The number of Male respondents is 1 time more than the number of Female respondents (*figure 1*), and the income distribution (target value distribution) is shown in *figure 2*. We can intuitively see that the income distributions of Male and Female are different. By applying Chi-square fitness test *[1]*, with the Null Hypothesis that the income distribution of Female was the same as the income distribution of Male, we get p-value 2.2e-16, which means that there is significant evidence that the income distributions are different among Male and Female.

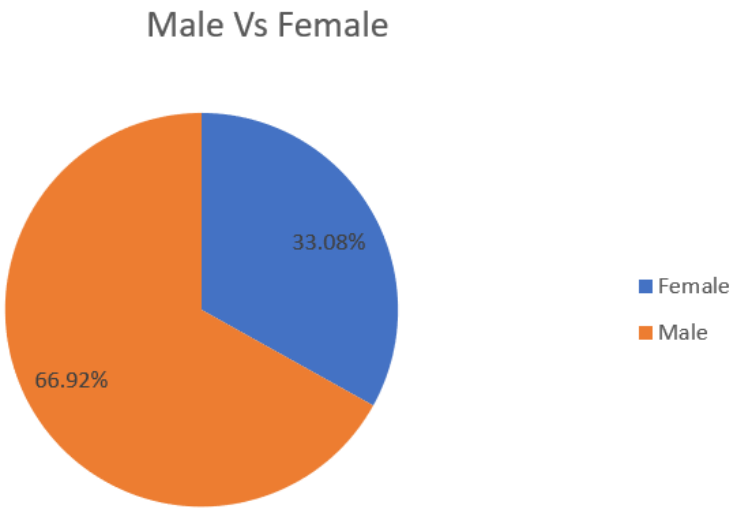

Figure 1: Male respondents are more than Female respondents

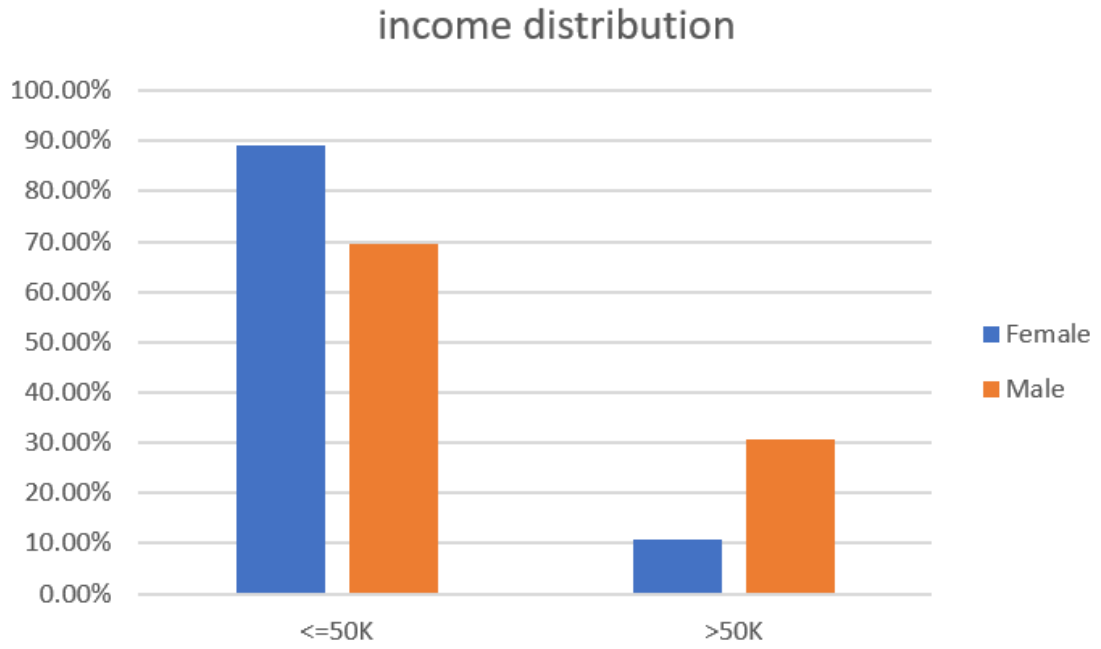

Figure 2: the income distribution of Female and Male are significantly different

There I use XGBoost *[3]* models trained on whole training dataset and its Male/Female subsets separately and evaluate their accuracy with testing set and its Male/Female subsets correspondingly:

	Precision for "<=50K"	Precision for "> 50K"	accuracy
XGBoost on Whole training set	0.8923	0.7732	0.8684
XGBoost on Female subset	0.9419	0.7284	0.9257
XGBoost on Male subset	0.8563	0.7747	0.8353

Table 1: XGBoost trained on whole training dataset and their Male/Female subsets and tested on whole testing set and its Male /Female subsets

Moreover, I also calculated the accuracy of XGBoost trained on whole training set while tested on Male/Female testing subsets (*table 2*):

	Precision for "<=50K"	Precision for "> 50K"	accuracy
Testing on Female subset	0.9506	0.7654	0.9367
Testing on Male subset	0.8537	0.7908	0.8374

Table 2: XGBoost trained on whole training set while tested on Male/Female testing subsets.

According to *table 1*, XGBoost can reach accuracy 0.8664 using whole training set. And XGBoost can also achieve good precision using different subsets (which accuracy 0.9257 and 0.8353 separately). Which suggested that Machine Learning models trained on different subsets can contribute to do classification work.

Unfortunately, comparing *table 2* with *table 1*, models trained on subsets don't have obvious advantages compared with model trained on whole training set:

1) For Female testing subset, the accuracy of model trained on whole training set is 0.9367, which is larger than the accuracy of model trained on Female training subsets -- 0.9257.
2) For Male testing subset, the accuracy of model trained on whole training set is 0.8374, which is larger than the accuracy of model trained on Male training subsets -- 0.8353.

These observations suggest that subsets models can not improve classification accuracy without proper transformation. we need to build a new model for aggerating the results of different models trained on various subsets, and then provide the final improved classification result. Neural Network is be a possible choice to solve this problem.

3 Methodology:

3.1 Algorithm design:

Firstly, a Machine Learning algorithm should be chosen as the footing stone. There I use XGBoost algorithm as the foundation model and later in chapter 5 more Machine Learning models will be tested.

The algorithm of Neural Network ensemble method is shown in *Figure 3*, where attribute Gender is chosen to be the criterion for dividing training and testing data.

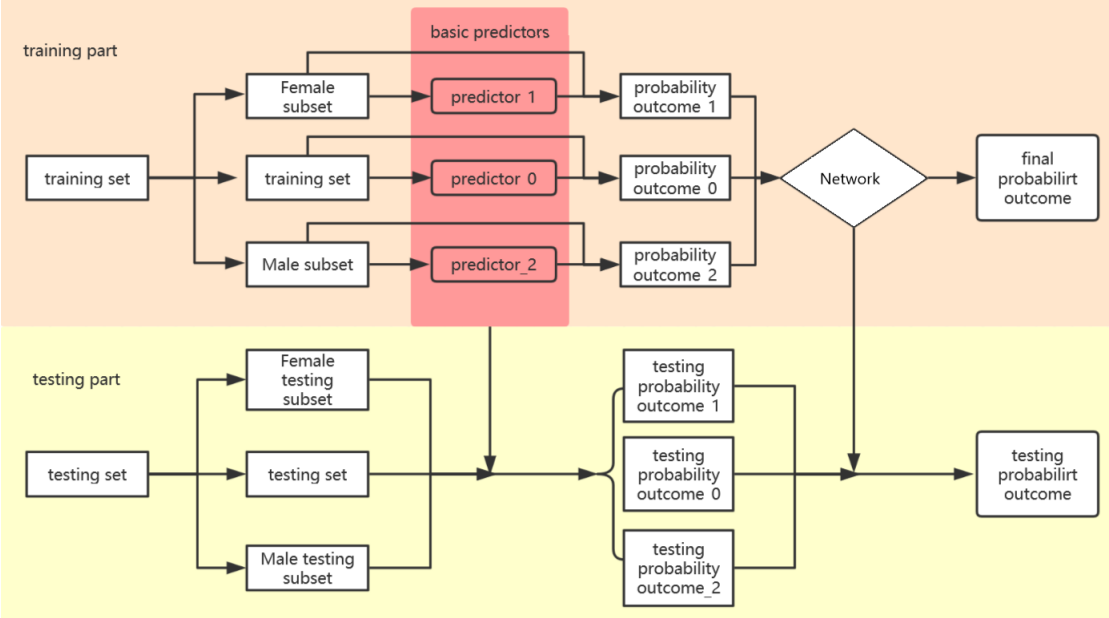

Figure 3: Neural Network ensemble method

The algorithm could be summarized as following:

1) Splitting whole training sets into several subsets (in *Figure 3*, they are illustrated as *training set*, *Female subset*, *Male subset*)
2) Training chosen foundation Machine Learning models using different subsets separately (in *Figure 3*, they are illustrated as *predictor_0*, *predictor_1* and *predictor_2*). Denote these models as basic predictors.
3) Let the training subsets used in step (2) be the input and applying models in basic predictors correspondingly to get the probability output (in *Figure 3*, it is denoted as *probability outcome_0*, *probability outcome_1* and *probability ourcome_2* separately)
4) Let the probability outcomes as the input of Convolution Neural Network (CNN) or linearly Connected Neural Network, and using the real income labels of training data to train its parameters.

When the training procedure is finished, the next step left is testing. For any input instance. Firstly, a set of probability outcomes is computed by pre-trained basic predictors in step (2). Secondly, the probability outcome will be passed to pretrained Neural Network in step (4). Finally, the probability output of Neural Network will be regarded as the final result for classifying testing instance.

In our case, we have 3 training subsets (whole training set, Female set, Male set). And generally, If the criterion of data splitting is changed, there may be more subsets. All probability vector is combined to form a single $(M+1)\times N$ matrix (Or a vector with $(M+1)\times N$ elements), where M is the number of strict subsets (In our case, M=2) and N is the number of target variable levels (In our case N=2 since it is a binary classification task).

The i^{th} row of matrix is the probability output of $predicotr\_(i-1)$, and all other position is 0. For example, if the input x labeled as Female, and the probability output of $predicotr\_0$ is [P_{01}, P_{02}] and the probability output of $predicotr\_1$ is [P_{11}, P_{12}]. Then the input matrix of Neural Network will be:

$$\begin{matrix} P_{01} & P_{02} \\ P_{11} & P_{11} \\ 0 & 0 \end{matrix}$$

You can image we have (M+1) predictors which can give their suggestions about how to classify input x, while only 2 of them are valid.

3.2 Numerical experiment:

All the Numerical tests in this research are constructed by Python (version 3.7). The foundation Machine Learning technique like XGBoost algorithm is applied with the help of *sklearn* package, while *Pytorch* is used for building and training Neural Network.

For XGBoost algorithm, all parameters are chosen as default value. However, there are multiple ways to design and optimize Neural Network. Since our input data for training Neural Network is usually small: the input data for Neural Network is a $(M+1)\times N$ matrix (or a numerical vector with $(M+1)\times N$ elements which is obtained by squeezing the input matrix). We should expect that the Neural Network having extremely simple structure since the input $(M+1)\times N$ matrix has several special properties:

1) Any input matrix has at most 2 non-zero rows and at least 1 no-zero rows. Any other row's elements are all zeros
2) The sum of any non-zero row is 1
3) Any non-zero row has small Euclidean distance with the target value. In other words, even we just randomly choose a non-zero row to be the final output without any additional transformation, we can still expect a reasonable result.

Because the task the Neural Network facing is very simple, and the input data is extremely clean. The Neural Net I used only contained one or two layers.

Convolution Neural Network contains 1 convolution layer and 1 linear layer. The convolution layer is a 1D convolution layer with one input channel, kernel Length N, stride N and output channel number K. And the following linear layer has a squeezed $(M+1)\times K$ vector as input and the output vector size is N. Here I choose Relu *[8]* has the active function for convolution layer and SoftMax as the active function for linear layer. In addition, I set the number of output channel of convolution layer as 12

For single-layer linear Neural Network, the linear layer is expected to have an input vector with $(M+1) \times N$ elements and output vector with N elements. There I choose SoftMax function as the active function. For double-layers Linear connected Neural Network, the first linear layer has an input $(M+1) \times N$ vector and output vector with size 16. The second linear layer has output vector size $N \times 1$. The active functions for first linear layer and second linear layer are Relu and SoftMax separately.

Since our Network is very simple and the input matrix/vector is extremely clean, the optimizer for backpropagation step I choose there is Stochastic Gradient Descent *[7]* without momentum validation. The batch size is 128 and the whole training processing contain 1000 epochs. Since our training data has 26048 instances, one epoch contains 204 times backpropagation. The Loss function applied for all Neural Network is Cross Entropy *[2]*. The learning rate for the first 10 epochs is 0.01 and for the last 990 epochs is 0.001.

Note that all 3 neural networks' foundation model are XGBoost and the *predicoter_0* is exactly the pretrained XGBoosting model in chapter 2. Thus we can directly compare the accuracy improvement of machine learning algorithm before and after Neural Network is applied. We regard the original Machine Learning model as the baseline and all numerical experiments shown later in this research will strictly follow the consistency of baseline model and *predictor_0*. This is the reason why cross-validation is not involved in numerical tests.

The test accuracy of 3 mention neural network structures is shown in *Figure 4.1*, and the average training loss of these 3 neural networks is illustrated as *Figure 4.2*. As we can see, the testing accuracy and training loss of convolution neural network and double linear layer are very similar (shown with blue and green color) while the training loss of single linear layer Neural Network is higher than others. This is mainly because the single linear layer Neural Network has simpler structure and less parameters than the other two Neural Networks (which have 2 layers).

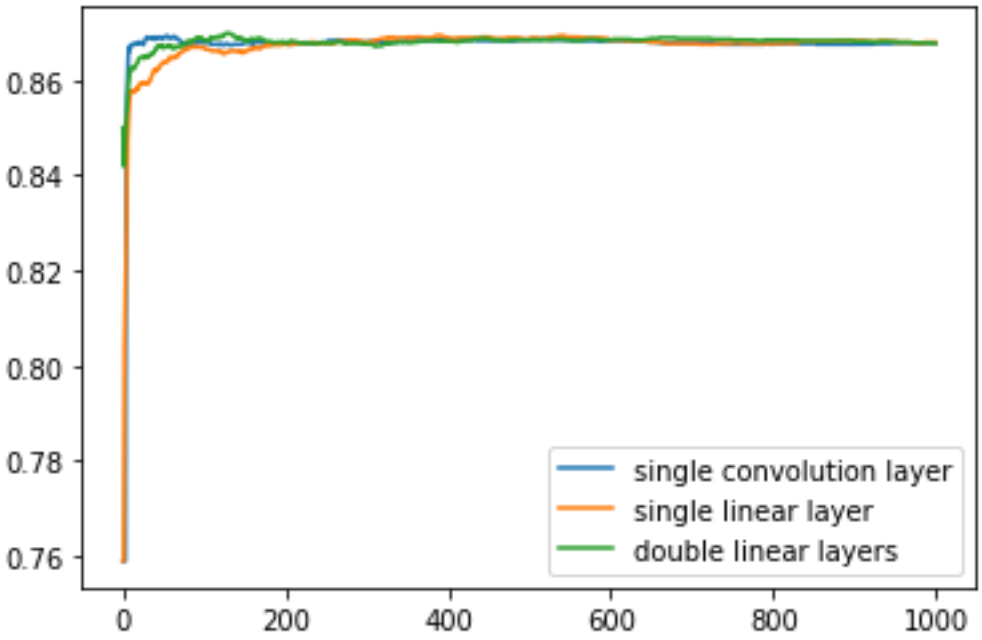

Figure 4.1: testing accuracy of 3 Networks

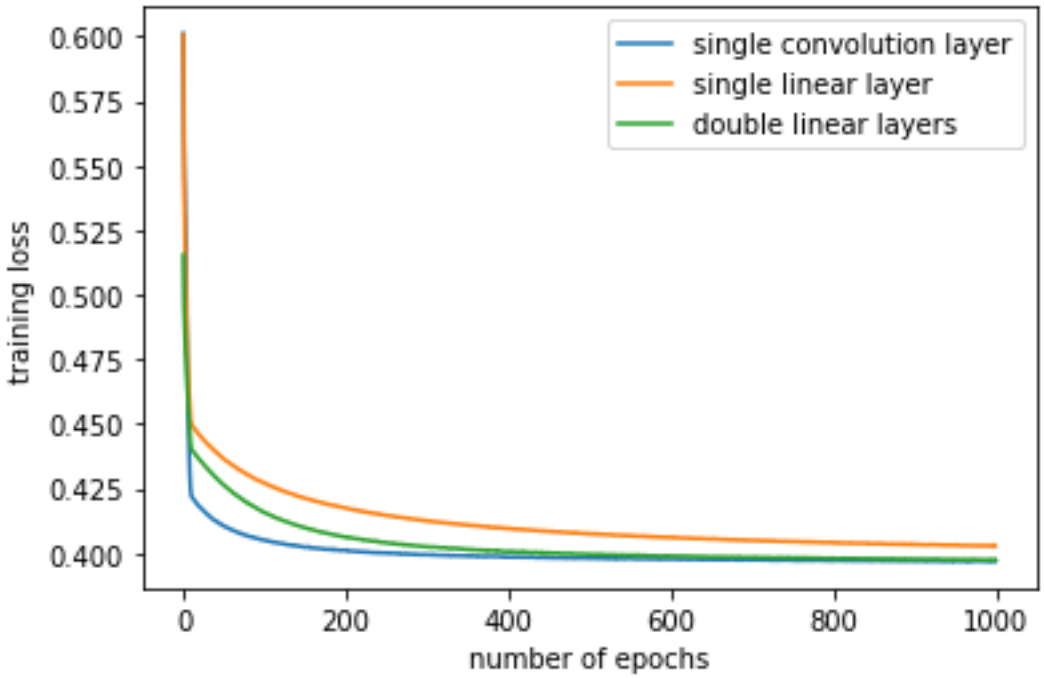

Figure 4.2: training loss of 3 Networks

Method	Accuracy in 1000 epochs	Maximum accuracy
Convolution Neural Net	0.8678	0.8693
Single linear layer	0.8680	0.8693
Double linear layers	0.8676	0.8693

Table 3.1: testing accuracy of 3 Networks

Method	Loss in 1000 epochs	Minimum Loss
Convolution Neural Net	0.3968	0.3967
Single linear layer	0.4027	0.4026
Double linear layers	0.3974	0.3971

Table 3.2: training loss of 3 Networks

Reminding that our baseline (accuracy achieved by XGBoost) is 0.8684. According to *Table 3.1 & 3.2*. the maximum accuracy achieved by 3 Neural Network is 0.1% percent higher than the baseline. However, all of 3 Neural Networks cannot provide satisfying testing accuracy after 1000 epochs. the overfitting issue become an important problem in our problem, and too many epochs may lead to overfitting.

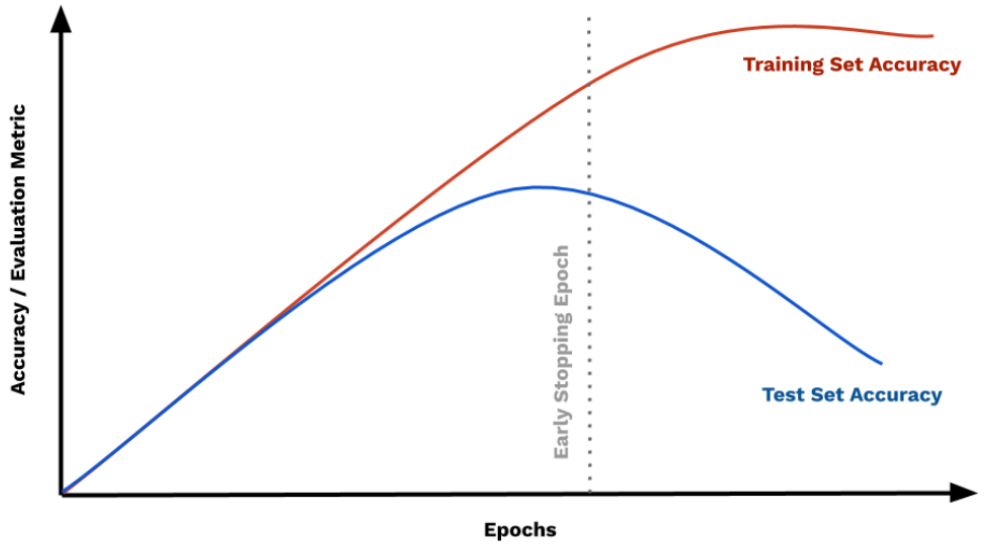

Figure 5: Early Stopping illustration [10]

The overfitting issue can be solved by early stopping, which means that we only use less epochs to train our neural network. As shown in *Figure 5*, when the number of epochs increases, the testing set accuracy may decrease. More detail will be discussed in Section 3.3.

3.3 Initialization method:

In the previous numerical test, all three Networks are initialized by standard normal distribution, which may not be the best initialization method for our ensemble method. Reminding that for single linear-layer neural network, the first N elements is the probability prediction of $predicotr\_0$, thus even the neural network do nothing but identical transformation, the over-all accuracy is still very high. Unlike

the classical normalization methods such as normal distribution initializer which regard these $M+1$ predictors with equally importance, here I introduce a specially designed sparse initialization method to emphasis the importance of $predicotr\_0$.

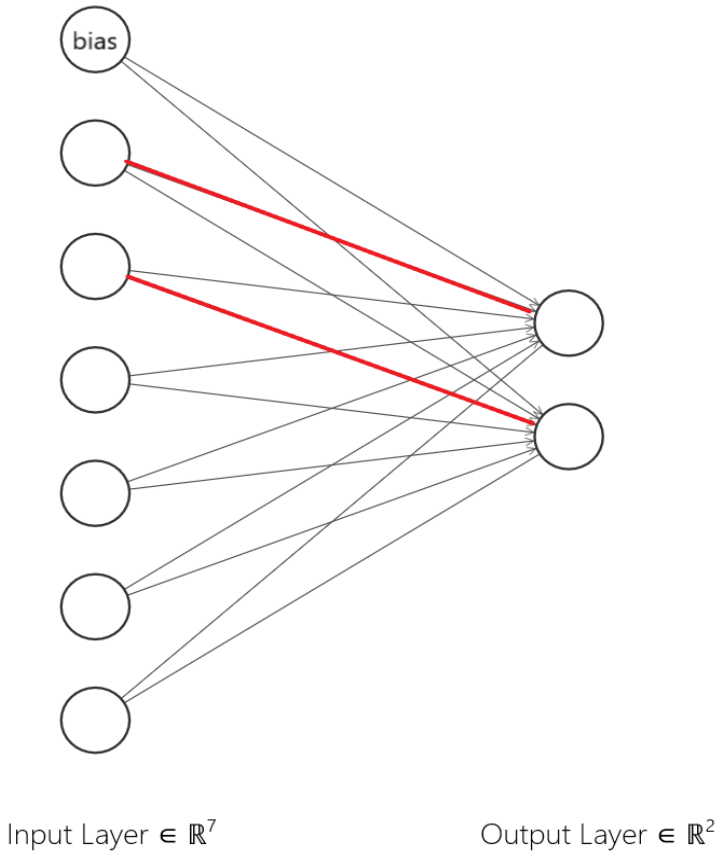

Figure 6: sparse initializer. The red lines have weight 1 while All other black lines have weight 0.

As *Figure 6* illustrating, the sparse initializer is designed for single-layer linear network. Only the red-color parameters are initialized as 1 and all other parameters including bias are 0. Thus the R^N output is exactly the same as the first N elements of input, which is the probability prediction result of $predicotr\_0$. Since the sum of R^N output equals to 0, the SoftMax active function won't change this R^N output vector.

By this special designed sparse initializer, the Neural network will be modified with $predicotr\_0$ as the ground, and using the information proved by $predicotr\_0$, $predicotr\_1 \dots predicotr\_M$ to enhance $predicotr\_0$. As shown in *Figure 7.1 &7.2*, it has better performance than normal initializer. Notice that except the initialization method, the numerical experiment is the same as what I did for normal initializer.

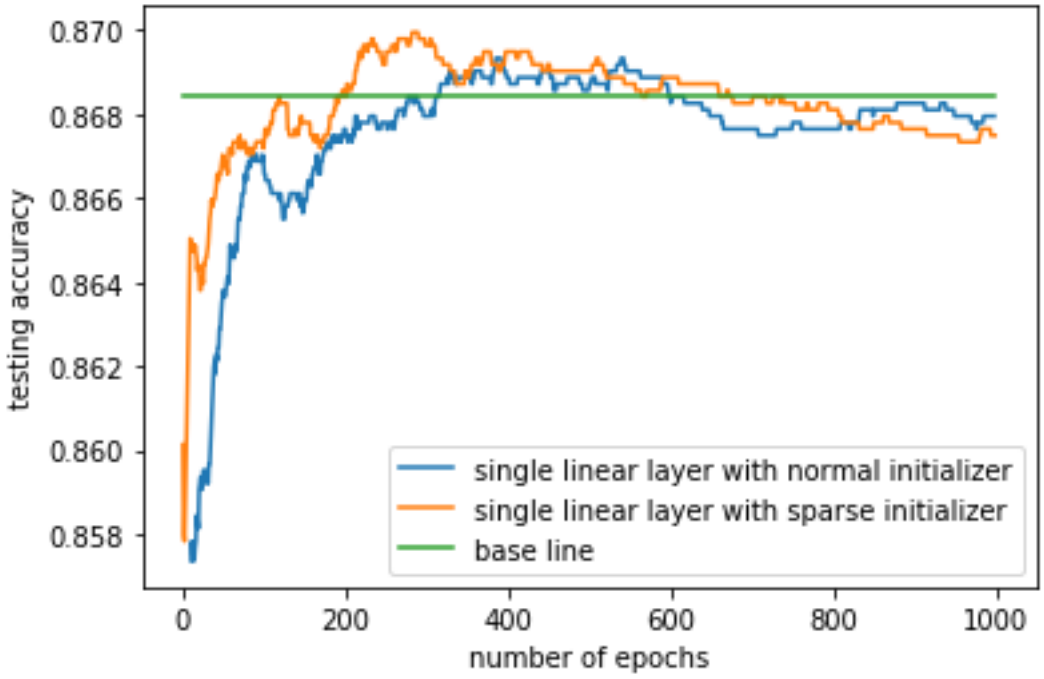

Figure 7.1: testing accuracy of normal initializer and sparse initializer. To make the graph clearer, the first 10 data points for single linear layer with normal initializer is dropped.

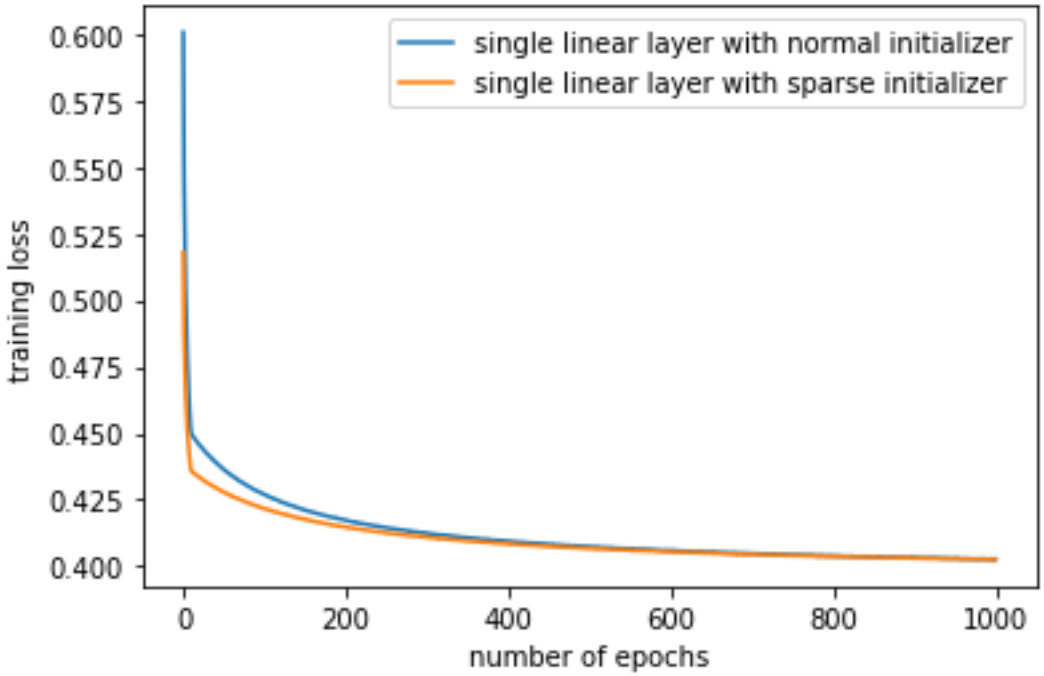

Figure 7.2: training loss of normal initializer and sparse initializer.

Method	Accuracy in 1000 epochs	Maximum accuracy
Sparse initializer	0.8675	0.87
Normal initializer	0.8680	0.8693

Table 4: testing accuracy of normal initializer and sparse initializer.

The overfitting issue still exists, and the maximum of single linear initializer shows in

280 epochs. I restrict the number of epochs to 280 and redo the test for single linear layer network with sparse initializer and the final result can be seen in *Figure 8 and Table 5*.

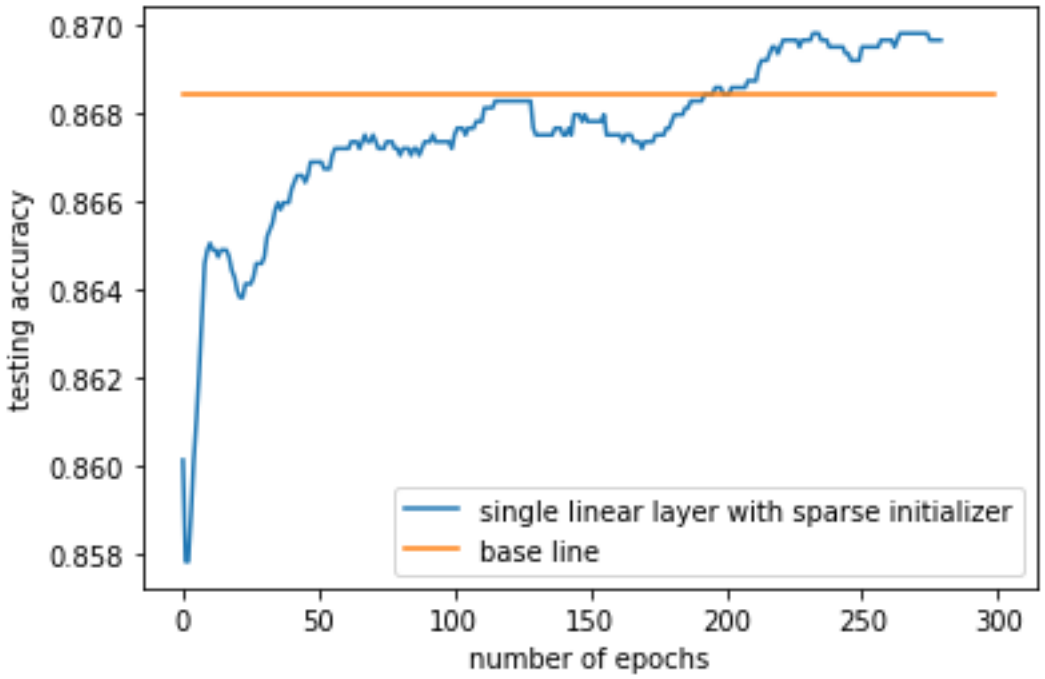

Figure 8:

single layer linear Network using sparse initializer and early stopping

VS

baseline

Method	Accuracy in 280 epochs	Maximum accuracy
Sparse initializer	0.86965	0.8698
Baseline	0.8684	0.8684

Table 5: single layer linear Network using sparse initializer and early stopping VS baseline

In conclusion, by applying a single linear neural network with spatialized sparse initializer, and with the help of early stopping, we can improve the accuracy of the original XGBoost. And there is an intuitively explanation for overfitting observation: since any non-zero row has small Euclidean distance with the target value, if we keep doing backpropagation without large loss reduction, we may introduce unnecessary transformation of non-zero rows, which will lead to lower validation score.

In the next chapter, feature selection method using genetic algorithm is applied to further improve the accuracy.

4, Feature selection using genetic algorithm

Feature selection is always an importance part in Machine Learning. By reducing the dimension of attributes and applying feature transformation, the accuracy of original Machine Learning model can be improved. There I use Genetic algorithm (GA) *[6][9]* to do feature selection and make a comparison of the Neural Network ensemble method before and after Genetic algorithm.

Genetic algorithm is an inductive learning strategy which have demonstrated substantial improvement over other commonly used feature selection method *[6]*. It can exploit accumulating information about an initially unknown problem *[6]*, which takes the interactive of different attributes into consideration. The genetic algorithm is an analogy of Darwin's natural selection theory: each feature selection is regarded as a gene and only the genes with the highest evaluation score can survive and pass to next generation.

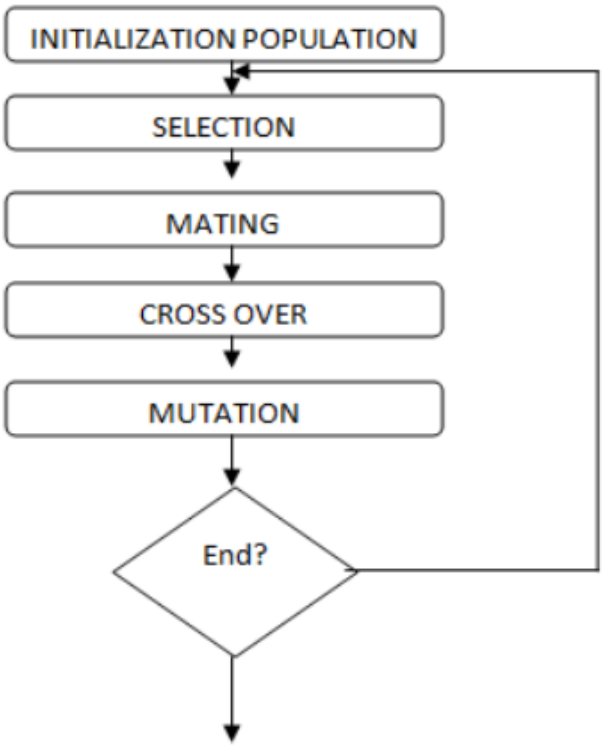

Figure 9: basic structure of genetic algorithm [4]

Here I set the population size as 40, the number of features to be selected in each generation as 5, the mutation probability as 0.05 for each gene segment and the total generation number as 10. 5-cross-validation is applied as the evaluation function.

In our case, the adult income classification has 14 attributes including real and nominal data, while after the genetic feature selection only 11 of them are selected.

As shown in the first row of *Table 6*, the testing accuracy increases after the genetic algorithm is applied.

	Accuracy before genetic algorithm	Accuracy after genetic algorithm
XGBoost on Whole training set	0.8684	0.8693
XGBoost on Female subset	0.9257	0.9360
XGBoost on Male subset	0.8353	0.8385

Table 6: testing accuracy of baseline model before and after feature selection.

Unlike traditional Machine Learning feature selection method, which do feature selection once and applied models on the new learning space, one advantage of this neural network ensemble method is that we can applied the generic algorithm several times separately on different training subsets. Taking the adult income classification problem as instance, there are totally 3 training subsets: ①whole training set ②Female subset ③Male subset. Thus 3 genetic algorithms are applied to find the most suitable learning space for these 3 subsets, and the result is : ①For whole training set, 11 of 14 attributes are selected ②For Female subset, 8 of 13(excluding Gender attribute) attributes are selected ③For Male subset, 10 of 13 attributes are selected. By combining neural network ensemble method and feature selection procedure, more carefully data analysis can be applied, and the feature selection procedure can also increase the variety of different training subsets.

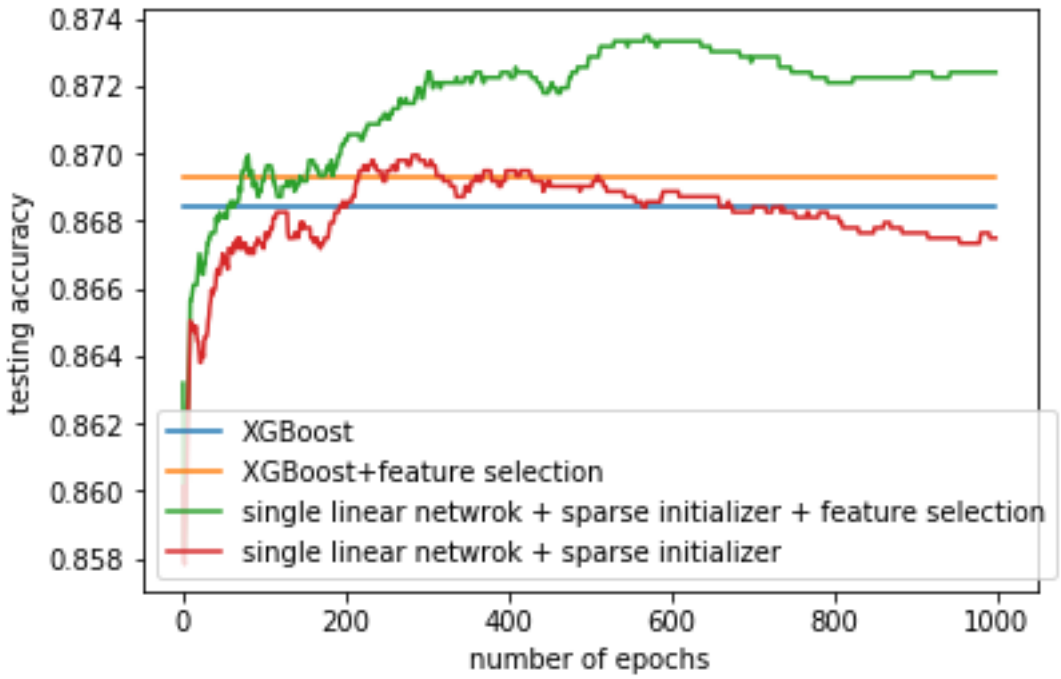

Figure 10: testing accuracy of Neural Network ensemble method + feature selection

Method	Accuracy in 1000 epochs	Maximum accuracy
XGBoost +feature selection	0.8693	0.8693
Single linear layer +feature selection	0.8724	0.8735

Table 7: testing accuracy of Neural Network ensemble method + feature selection

As we can see in *Figure 10 & Table 7*, the ensemble method + feature selection can provide 0.3-0.4% accuracy improvement compared to using XGBoosting + feature selection, and 0.7% accuracy improvement compared to using XGBoosting without feature selection.

5 More numerical experiment:

In previous discussion, it can be shown that the neural network ensemble method works for adult income classification task based on XGBoost model. In this Chapter, I am going to show the generality of this ensemble method. Here I applied various classification method including Random Forest, SVM, logistic regression, MLP, Gradient Boosting and XGBoost on different dataset, then I select the model with the highest testing accuracy and see how the ensemble method can affect it.

All the Machine Learning method are implemented by sklearn package with default parameter set. the random seed for all Machine learning technique is 42 and the random seed for train-test splitting is 10. One-Hot-Encoding pre-processing method is used for categorical attributes and if there is any missing-value, it will be deal with a new level called ‘Missing Value’, so they can be treated as legal discrete values. For all the missing values of numerical attributes, they are just ignored in the training and testing procedure.

Note that the default MLP in sklearn is a single layer neural network with weight 100. Here I set the maximum iteration for training MLP as 1000. And the default kernel for SVM is gaussian kernel.

For the adult income classification problem we discuss in previous 4 Chapters, XGBoost is already the best performance model we have. 1000 epochs are applied with learning rate 0.01 for first 10 epochs and 0.001 for remining epochs.

Method	Accuracy	
Random Forest	0.8524	
SVM	0.795	
Logistic regression	0.7958	
MLP	0.8419	
Gradient Boosting	0.8643	
XGBoost	0.8684	
XGBoost+ feature selection	0.8693	
	End Accuracy	Maximum Accuracy
Single-layer linear network + feature selection	0.8724	0.8735

Table 8: empirical testing for adult incomes dataset.

Besides the adult income datasets, there I choose another 4 datasets from UCI repository *[5]*. For numerical experiments implemented on these datasets, the feature selection part is skipped since GA has too much randomness and is hard to reproduce.

Data Set	instance	Input attribute	attribute type	Classes
Statlog (Australian Credit Approval) *[11]*	666	15	Categorical, Integer, continuous	2
Soybean(large) *[12]*	306	35	Categorical	19
Contraceptive Method Choice *[13]*	1473	9	Categorical, Integer	3
Covertype *[14]*	581012	54	Categorical, Integer	7

Table 9: dataset summary

Among these four additional datasets, the smallest dataset contains only 306 instances while the largest dataset contains 581012 instances. The neural network structure is chosen as the single linear layer Network with sparse initializer in Chapter 3, and the input and output size are automatically fitted. The only two differences between these four additional datasets and the adult income dataset is: ①the attribute chosen to split data ②Network learning rate and epoch number.

Different choice of criterion attribute (using to split data) can lead to different testing accuracy, and there is no simple way to select the best one. Exhaustive method can be a reliable method to select the best criterion, but it is extremely time-consuming for dataset with too many categorical variables and large size. For example, the "Soybean(large)" dataset, there are 35 categorical variables and the "Covertype" dataset has 581012 instances. There I subjectively choose one categorical attributes as the criterion attribute for each dataset:

1) Statlog (Australian Credit Approval): choose the first categorical attribute as the

criterion attribute.

2) Soybean(large): choose the 5^{th} categorical attribute as the criterion attribute.
3) Contraceptive Method Choice: choose the 4^{th} categorical attribute as the criterion attribute.
4) Covertype: choose the “Wilderness_Area” categorical attribute as the criterion of splitting data into subsets.

The learning rate and epoch number also varies for different datasets.

1) Statlog (Australian Credit Approval): 2000 epochs with learning rate 0.01 in the first half and 0.001 in the second half.
2) Soybean(large): 1000 epochs are using for training and the learning rate is 0.001.
3) Contraceptive Method Choice: 2000 epochs with learning rate 0.01 in the first half and 0.001 in the second half.
4) Covertype: 4000 epochs with learning rate 0.01 is used.

Dataset： Statlog (Australian Credit Approval)		
Random Forest	0.8806	
SVM	0.6418	
Logistic regression	0.8731	
MLP	0.806	
Gradient Boosting	0.8955	
XGBoost	0.8955	
	End Accuracy	Maximum Accuracy
Single-layer linear network + Gradient Boosting	0.8955	0.8955
Dataset: Soybean(large)		
Random Forest	0.8225	
SVM	0.8548	
Logistic regression	0.8065	
MLP	0.7097	
Gradient Boosting	0.8065	
XGBoost	0.77419	
	End Accuracy	Maximum Accuracy
Single-layer linear network + Gradient Boosting	0.8871	0.8871
Dataset: Contraceptive Method Choice		
Random Forest	0.5186	
SVM	0.4814	
Logistic regression	0.5119	
MLP	0.5559	
Gradient Boosting	0.5559	
XGBoost	0.522	
	End Accuracy	Maximum Accuracy
Single-layer linear network + Gradient Boosting	0.5695	0.5525
Dataset: Covertype		
Random Forest	0.9556	
SVM	N/A	
Logistic regression	0.6866	
MLP	0.7648	
Gradient Boosting	0.7739	
XGBoost	0.8699	
	End Accuracy	Maximum Accuracy
Single-layer linear network + Gradient Boosting	0.9567	0.9567

Table 10: numerical experiment for 4 additional set. The highest traditional machine learning accuracy and the neural network ensemble method accuracy is highlighted as red color.

6 Conclusion and Future research

Neural Network ensemble method is an inspiring method for classification task. Based on the assumption that the data can be spitted into disjoint subsets with various hidden structure, Neural Network ensemble method aims to aggregate the learning results of models trained on different subsets.

This method can be an extension of existing data analysis frame: besides normal feature selection, transformation and model building, we can repeat this procedure several times for different subsets and further analysis the hidden data structure behind different subsets. For the Neural Network training part, overfitting is an important issue need to be considered. By the specially designed sparse initialization strategy, we assign the model trained on whole training set 100% initial weight, which gives the Neural Network stronger physical meaning.

For neural Network design, both linear Network and convolutional Network can be applied to aggregate foundation model results, and there are lots of optimizers and loss functions to be chosen, which gives the Neural Network ensemble method strong flexibility.

However, there is one importance question need to be solved: how to split data. You may discover that all 5 datasets in used in this research contain at least one categorical variable, which is regarded as data splitting criterion. However, there are more datasets in realistic which only contain integer and continues data. Moreover, even for datasets contain categorical variables, the choice of splitting criterion is limited, and we don't have an effective method to choose the best one. I think it is a good chance to combine supervise learning (the Neural Network ensemble method) and un-supervise learning (for example clustering). For any dataset, we can firstly apply un-supervise learning technique to divide data into several subsets and apply the Neural Network ensemble method.

Reference:

[1] Agresti, A. (2007). *An Introduction to Categorical Data Analysis*, 2nd ed. New York: John Wiley & Sons. Page 38.

[2] Boer, P. & Kroese, D. & Mannor, S. & Rubinstein, R.. (2002). A Tutorial on the Cross-Entropy Method. Ann Oper Res. 134.

[3] Chen, Tianqi & Guestrin, Carlos. (2016). XGBoost: A Scalable Tree Boosting System.

[4] De Jong, K. "Learning with Genetic Algorithms : An overview," Machine Learning Vol. 3, Kluwer Academic publishers, 1988.
Yadav, S. L. and Sohal, A. (2018) 'Comparative Study of Different Selection Techniques in Genetic Algorithm'

[5] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[6] H. Vafai and K. Dejong (1992). "Genetic algorithms as a tool for feature selection in Machine Learning ". in proceedings of TAI'92.

[7] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

[8] Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: International Conference on Machine Learning. pp. 807–814 (2010)

[9] Thesis (Ph. D.)--University of Michigan, 1975. Includes bibliographical references (leaves 253-256). Photocopy.

[10] https://deeplearning4j.org/docs/latest/deeplearning4j-nn-early-stopping

[11] http://archive.ics.uci.edu/ml/datasets/Statlog+%28Australian+Credit+Approval%29

[12] http://archive.ics.uci.edu/ml/datasets/Soybean+%28Large%29

[13] http://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice

[14] http://archive.ics.uci.edu/ml/datasets/Covertype

[15] *https://www.kaggle.com/uciml/adult-census-income*