International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 01 | Jan 2021 www.irjet.net p-ISSN: 2395-0072

Machine Learning Algorithms for the Detection of Diabetes

Ranjith M S[1], Santhosh H S[2], Swamy M S[3]

[1,2,3]*Student, Dept. of ECE, The National Institute of Engineering, Mysuru, India*

Abstract - *Diabetes is one of the fastest-growing chronic life-threatening diseases that is the seventh leading cause of death according to the estimation of WHO in 2016. Due to the presence of a relatively long asymptomatic phase, early detection of diabetes is always desired for a clinically meaningful outcome. Diabetes is a major cause of blindness, kidney failure, heart attacks, stroke, and lower limb amputation. These Could be avoided if it is detected in early stages. A large amount of clinical data available due to the digital era has made it possible for Deep learning techniques to give good results on medical diagnosis and prognosis. We use the diabetes dataset to train the model in predicting it. We have analyzed the dataset with Logistic Regression Algorithm, Random Forest Algorithm, Deep Neural Network, and a DNN having the embeddings for the categorical features. The DNN with embeddings makes the correct predictions on most of the test set examples i.e., nearly 100% of the examples, and has an F1 score of 1.0 on test data.*

Key Words: Deep learning, Regression, Diabetes, Health Care, Neural Networks, TensorFlow

1. INTRODUCTION

Diabetes Mellitus, a chronic metabolic disorder, is one of the fastest-growing health crises of this era regardless of geographic, racial, or ethnic context. Commonly, we know about two types of diabetes called type 1 and type 2 diabetes. Type 1 diabetes occurs when the immune system mistakenly attacks the pancreatic beta cells and very little insulin is released to the body or sometimes even no insulin is released to the body. On the other hand, type 2 diabetes occurs when our body doesn't produce proper insulin or the body becomes insulin resistant [1].

Some researchers divided diabetes into Type 1, Type 2, and gestational diabetes [2]. Agrawal, P et.al [3] showed 88.10% accuracy using the SVM and LDA algorithm together on a dataset having 738 patient data. They have also tried other methods like CNN, KNN, SVM, NB.

Joshi et.al [4] used logistic regression, SVM, ANN on a dataset having 7 attributes. After comparing their features, the researcher opinionated that the Support Vector Machine (SVM) as the best classification method.

Sapon et.al [5] found 88.8% accuracy with the Bayesian Regulation algorithm on a dataset from 250 diabetes patients' data from Pusat Perubatan University Kebangsaan Malaysia, Kuala Lumpur.

Rabina et.al [6] predicted using supervised and unsupervised algorithms. They used the software tool WEKA to find a better prediction algorithm in machine learning. Finally, they concluded that ANN or Decision tree is the best way for diabetes prediction.

The main contributions of this paper are:

a) We used the dataset from Islam, MM Faniqul, et al [7] and used different algorithms like Decision tree, Random forest, logistic regression, Naïve Bayes, KNN, Neural network, and Neural Networks with embeddings in predicting diabetes.
b) Our neural network using the embeddings layer on the categorical features learns a distributed representation of the categories. This method outperforms every other method leading to an accuracy of 100%, and an F1 score of 100% with sensitivity near to 100%.

Since we are using only dense networks we use *He initialization* [8] as compared with *Xavier Initialization* [9] for initializing the weights while training the network. The optimizers such as Adam [10], RMS [11], LARS [12] lead to faster convergence. "*One of the key elements of super-convergence is training with one learning rate cycle and a large maximum learning rate*" [13]. To speed up training we use *Batch Normalization* [14].

2. EXPERIMENTAL SETUP

2.1 Data pre-processing

The dataset contains both categorical and continuous features. Categorical features include Gender, Polyuria, Polydipsia, sudden weight loss, weakness, Polyphagia, Genital thrush, visual blurring, itching, Irritability, delayed healing, partial paresis,

muscle stiffness, Alopecia, Obesity. All these categorical features have only two classes yes or no except sex which has male or female as two classes. The only continuous feature present in the dataset is Age varying between 16 and 90. The output is having two classes yes or no indicating whether a person has diabetes or not.

The categorical features are encoded into integers. The continuous variable Age is mean normalized in order to have mean 0 and standard deviation of 1 which helps in faster convergence and also counters the problems like vanishing gradient when parameterized models like neural network are used.

The mean normalization is given by,

$$x_i = \frac{x_i - \mu_i}{s_i}$$

Where μ(i) is the **average** of all the values for feature (i) and *s(i)* is the range of values (max-min)

2.2 Reducing the size of dataset

When the CSV file is read as a data frame in the python environment the default size of the values will be Int64 for the integers and float32 for floating-point numbers. These default allocations lead to the consumption of a large amount of memory in runtime. The memory size is reduced by considering the maximum and minimum values of the column. For e.g., the categorical columns have two values 0 and 1 so the default allocation of int64 is changed to uint8. After applying these reductions on all columns, the size of the dataset is reduced to 12.65% of the initial size.

3 TRAINING

The preprocessed dataset is trained using different machine learning algorithms, decision tree, random forest, Naïve Bayes, regression, K nearest neighbors, Artificial neural networks, and ANN's with embedding layers for the categorical features.

3.1 ANN's with embedding layers

TensorFlow Keras is used for the implementation and training of the neural network**.** The embedding layer will capture more details on the relationship between the categorical variables. The embedding layer is introduced only to the categorical features and the dimension is chosen according to the number of unique values present in a particular attribute. Since in our case all the categorical attributes have two unique values we chose the output dimension to be 2 for each attribute. All these embeddings are flattened and given as input to the dense layer. Two dense layers are introduced exclusively for the categorical features with each layer having 16 hidden units. The output of the dense layer is concatenated with the continuous feature age as shown in fig.1. These are further connected to the other 2 dense layers having 16 and 8 hidden units respectively. The batch normalization and dropout are used after each dense layer to train the model effectively. The output layer has one hidden unit and has a sigmoid activation function whereas other intermediate dense layers have ReLU as the activation function.

Binary cross-entropy is used as the loss function and is given by,

$$BCE = -\sum_{i=1}^{c'=2} t_i log\left(f(s_i)\right) = -t_1 log\left(f(s_1)\right) - (1-t_1) log\left(1-f(s_1)\right)$$

Where $f(s)_i$ represents sigmoid function given by,

$$f(s_i) = \frac{1}{1+e^{-s_i}}$$

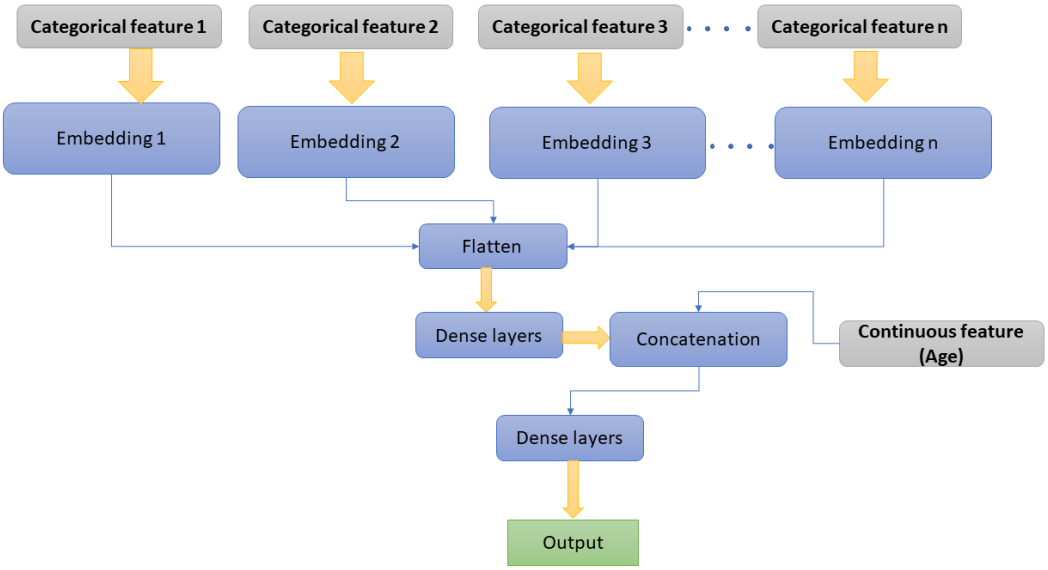

FIG.-1 Architecture

The model is trained for 150 epochs with Adam optimizer. The model was trained on Tesla K80 GPU which can do up to 8.73 Teraflops single-precision and up to 2.91 Teraflops double-precision performance with NVIDIA GPU Boost. This network gives a validation accuracy of 100 %. Fig.2 shows the variation of training and validation loss during each epoch of training.

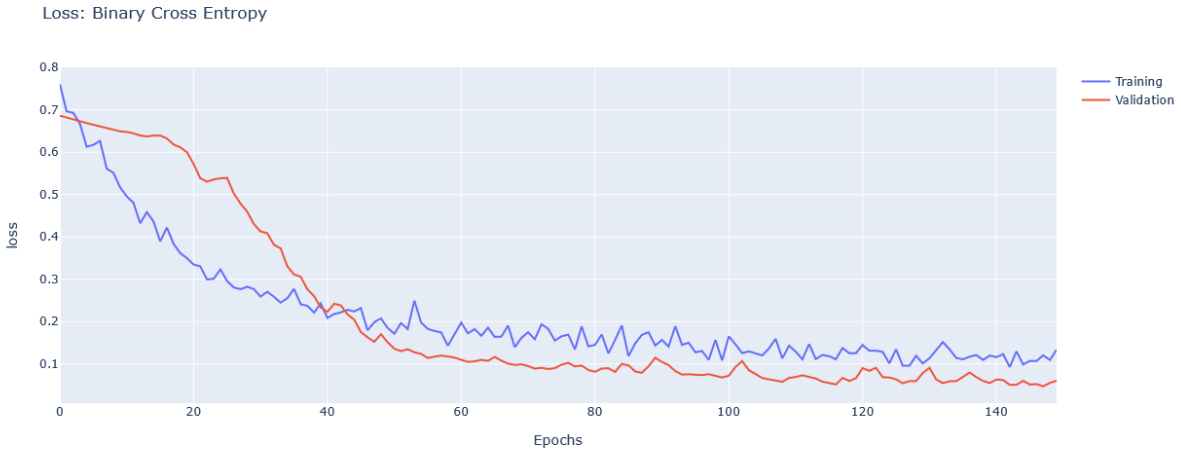

Fig-2: Variation of loss with epochs

3.2 Artificial Neural Network

TensorFlow Keras is used for the implementation and training of the neural network**.** The created network has four layers having 128, 64, 16, 4 hidden units respectively. The output layer has 2 nodes since categorical cross-entropy is used as a loss function. ReLU is used as an activation function in intermediate layers and the output layer uses SoftMax as the activation function.

Categorical cross-entropy is given by,

$$CE = -\sum_{i}^{C} t_i \log(f(s)i)$$

Where $f(s)_i$ represents SoftMax function and is given by,

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}}$$

Where C represents the total number of class and s_j are the score inferred by the neural network for each class in C.

Table:1 shows the network structure implemented.

Table-1: Network structure

MODEL STRUCTURE AND PARAMETERS		
Layer	**OUTPUT SHAPE**	**NO. of Parameters**
Dense_1	(None,128)	2176
Dropout_1	(None,128)	0
Dense_2	(None,64)	8256
Dropout_2	(None,64)	0
Dense_3	(None,16)	1040
Dropout_3	(None,16)	0
Dense_4	(None,4)	68
Droout_4	(None,4)	0
Dense_5	(None,2)	10
Total Parameters: 11,550 **Trainable Parameters: 11,550** **Non -Trainable Parameters: 0**		

3.3 Random Forest

Random Forest is an ensemble algorithm. Decision trees with a maximum depth of 100, 5 minimum leaf nodes were used in the structure. 500 Decision trees were ensembled to get a training accuracy of 0. 9547 and test accuracy of 0. 9487 which performs better than the individual decision tree.

3.4 K-Nearest Neighbors

K-Nearest Neighbors take the Kth number of nearest neighbors and calculate the distance between query-instance and all the samples in the training set. Based on that it predicts the output that has minimum distance among all neighbors. In this experiment, we took K as 15 to achieve 0.9208 train accuracy and 0.9358 test accuracy**.**

3.5 Naïve Bayes

Naïve Bayes is a probabilistic algorithm that works based on the Bayes theorem. The equation of Bayes theorem is stated below

P(A\B) = p(B\A) *p(A) / p(B

It predicts the output based on the likelihood probability of each class. We achieved 0.9095 train accuracy and 0.8974 test accuracy using this.

3.6 Decision Tree

A tree structure with a depth of 50 and a minimum of 15 leaf nodes was created. Using the decision tree, a data set is broken down into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed.

3.7 Logistic regression

Logistic regression is a parameterized machine learning algorithm. Where the parameters along with the activation function define the decision boundary. Sigmoid is used as activation and the parameters are learned by looking at the dataset.

4. RESULTS

The entire dataset is split into 85% for training and 15% for validation. All the algorithms were given the same set of training and validation data.

Matrices		Logistic Regression	Naïve Bayes	K nearest neighbors	Decision Tree	Random Forest	Neural network	Neural network with embeddings
accuracy	Train	0.9276	0.9095	0.9208	0.9208	0.9547	**0.9915**	0.9699
	Test	0.9358	0.8974	0.9358	0.9102	0.9487	0.9872	**1.0**
f1 score	Train	0.9411	0.9270	0.9323	0.9353	0.9629	**0.9963**	0.9749
	Test	0.9462	0.9148	0.9450	0.9247	0.9565	0.9894	**1.0**
Precision score	Train	0.9446	0.9236	0.9877	0.9440	0.9737	**0.9927**	0.9628
	Test	0.9565	0.9148	0.9772	0.9347	0.9777	0.9791	**1.0**
Recall score	Train	0.9377	0.9304	0.8827	0.9267	0.9523	**1.0000**	0.9882
	Test	0.9361	0.9148	0.9148	0.9148	0.9361	**1.0000**	**1.0**
specificity	Train	0.9112	0.8757	0.9822	0.9112	0.9585	**0.9881**	0.9455
	Test	0.9354	0.8709	0.9677	0.9032	0.9677	0.9677	**1.0**

Table-2: Results

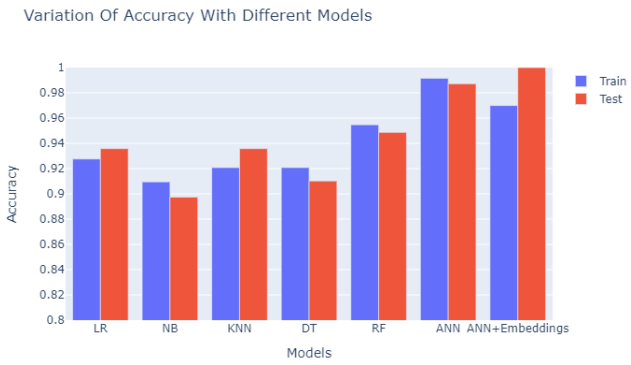

Fig-3. Performance of different models

Table-2 shows the performance of all the models and it is observed that all the best score belongs to ANN or the ANN's with Embeddings. Pure ANN's are slightly biased towards predicting the diseased condition as its evident from a recall score of 1 and a precision of 0.97. ANN+embeddings has a test accuracy of 1 indicating that it predicts all the test examples correctly.

Since logistic regression involves a smaller number of parameters it is observed that there is less difference in the training and the test scores indicating its robustness towards overfitting. It obtains a .94+ F1 score both on the training and validation set. Naïve Bayes is probabilistic based so is expected to perform poorly on the test set as it does not learn about the features and is evident from 0.89 test set accuracy. KNN has a test accuracy of 0.93.

5. DISCUSSION

As the data mining methods, techniques and tools are becoming more promising to predict diabetes and eventually several patients reduce the treatment cost, its role in this medical health care is undeniable. We found that the ANN+ebmeddings performs very well on the test dataset and is also highly robust to variance. They have correctly predicted all the test set examples indicating its credibility in the real-world scenario. The reason behind its performance is embedding layers' ability to capture distributed representation meaning, they help in capturing more insights about each category in each of the categorical attributes. Pure ANN's are also very near to the performance of ANN+embeddings but the latter gives the prediction based on the more insights gained through embedding layers which makes it more reliable.

REFERENCES

[1] The 6 Different Types of Diabetes: (5 Mar 2018). The diabetic journey. https:// thediabeticjourney.com/the-6-different-types-of-diabetes
[2] Statistics About Diabetes: American Diabetes Association, 22 Mar 2018. https://www.diabetes. org
[3] Agrawal, P., Dewangan, A.: A brief survey on the techniques used for the diagnosis of diabetesmellitus. Int. Res. J. Eng. Technol. (IRJET). 02(03) (2015). e-ISSN: 2395-0056; p-ISSN: 2395- 0072
[4] Joshi, T.N. Chawan, P.M.: Diabetes prediction using machine learning techniques. Dewangan, S. et.al. Int. J. Eng. Res. Appl. (Part -II) 8(1), 09–13 (2018). ISSN: 2248-9622

[5] Sapon, M.A., Ismail, K., Zainudin, S.: Prediction of diabetes by using artificial neural network. In: 2011 International Conference on Circuits, System and Simulation IPCSIT, vol. 7. IACSIT Press, Singapore (2011)
[6] Rabina1, Er. Anshu Chopra2: Diabetes prediction by supervised and unsupervised learning with feature selection, 2(5). ISSN: 2454-132
[7] Islam, MM Faniqul, et al. 'Likelihood prediction of diabetes at early stage using data mining techniques.' Computer Vision and Machine Intelligence in Medical Image Analysis. Springer, Singapore, 2020. 113-125.
[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification
[9] Bengio, Yoshua and Glorot, Xavier. Understanding thedifficulty of training deep feedforward neural networks.InProceedings of AISTATS 2010, volume 9, pp. 249–256, May 2010.
[10] Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization. arXiv:1412.6980v9 [cs.LG]
[11] Geoffrey Hinton. 2012. Neural Networks for MachineLearning - Lecture 6a - Overview of mini-batch gradi-ent descent.
[12] Yang You, Igor Gitman, Boris Ginsburg. Large Batch Training of Convolutional Networks. arXiv:1708.03888v3 [cs.CV]
[13] Leslie N. Smith, Nicholay Topin. Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates. arXiv:1708.07120v3 [cs.LG]
[14] Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167v3 [cs.LG]
[15] Alexandre de Brébisson, Étienne Simon, Alex Auvolat, Pascal Vincent, Yoshua Bengio. Artificial Neural Networks Applied to Taxi Destination Prediction. arXiv:1508.00021v2 [cs.LG]
[16] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S.,Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G.,Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv:1603.04467 [cs.DC]
[17] Ranjith M S, S Parameshwara. Optimizing Neural Networks for Embedded Systems. IRJET Volume 7, Issue 4, April 2020 S.NO: 211
[18] S. Coric, I. Latinovic and A. Pavasovic, "A neural network FPGA implementation," Proceedings of the 5th Seminar on Neural Network Applications in Electrical Engineering. NEUREL 2000 (IEEE Cat. No.00EX287), Belgrade, Yugoslavia, 2000, pp. 117-120.
[19] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.
[20] Plotly Technologies Inc. (2015). Collaborative data science. Montreal, QC: Plotly Technolo-gies Inc.
[21] S. Sahin, Y. Becerik1i, S. Yazici, "Neural network implementation in hardware using FPGAs", NIP Neural Information Processing, vol. 4234, no. 3, pp. 1105-1112, 2006.
[22] Yufeng Hao. A General Neural Network Hardware Architecture on FPGA arXiv:1711.05860 [cs.CV]