On Breast Cancer Detection: An Application of Machine Learning Algorithms on the Wisconsin Diagnostic Dataset

Abien Fred M. Agarap
Department of Computer Science
Adamson University
Manila, Philippines
abien.fred.agarap@adamson.edu.ph

ABSTRACT

This paper presents a comparison of six machine learning (ML) algorithms: GRU-SVM[4], Linear Regression, Multilayer Perceptron (MLP), Nearest Neighbor (NN) search, Softmax Regression, and Support Vector Machine (SVM) on the Wisconsin Diagnostic Breast Cancer (WDBC) dataset[22] by measuring their classification test accuracy and their sensitivity and specificity values. The said dataset consists of features which were computed from digitized images of FNA tests on a breast mass[22]. For the implementation of the ML algorithms, the dataset was partitioned in the following fashion: 70% for training phase, and 30% for the testing phase. The hyper-parameters used for all the classifiers were manually assigned. Results show that all the presented ML algorithms performed well (all exceeded 90% test accuracy) on the classification task. The MLP algorithm stands out among the implemented algorithms with a test accuracy of ≈99.04% Lastly, the results are comparable with the findings of the related studies[18, 23].

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Supervised learning by regression**; **Support vector machines**; **Neural networks**;

KEYWORDS

artificial intelligence; artificial neural networks; classification; linear regression; machine learning; multilayer perceptron; nearest neighbors; softmax regression; supervised learning; support vector machine; wisconsin diagnostic breast cancer dataset

1 INTRODUCTION

Breast cancer is one of the most common cancer along with lung and bronchus cancer, prostate cancer, colon cancer, and pancreatic cancer among others[2]. Representing 15% of all new cancer cases in the United States alone[1], it is a topic of research with great value.

The utilization of data science and machine learning approaches in medical fields proves to be prolific as such approaches may be considered of great assistance in the decision making process of medical practitioners. With an unfortunate increasing trend of breast cancer cases[1], comes also a big deal of data which is of significant use in furthering clinical and medical research, and much more to the application of data science and machine learning in the aforementioned domain.

Prior studies have seen the importance of the same research topic[18, 23], where they proposed the use of machine learning (ML) algorithms for the classification of breast cancer using the Wisconsin Diagnostic Breast Cancer (WDBC) dataset[22], and eventually had significant results.

This paper presents yet another study on the said topic, but with the introduction of our recently-proposed GRU-SVM model[4]. The said ML algorithm combines a type of recurrent neural network (RNN), the gated recurrent unit (GRU)[8] with the support vector machine (SVM)[9]. Along with the GRU-SVM model, a number of ML algorithms is presented in Section 2.4, which were all applied on breast cancer classification with the aid of WDBC[22].

2 METHODOLOGY

2.1 Machine Intelligence Library

Google TensorFlow[3] was used to implement the machine learning algorithms in this study, with the aid of other scientific computing libraries: matplotlib[12], numpy[20], and scikit-learn[15].

2.2 The Dataset

The Wisconsin Diagnostic Breast Cancer (WDBC) dataset[22] was used to implement the machine learning algorithms (in Section 2.4) for breast cancer diagnosis. According to [22], the dataset consists of features which were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The said features describe the characteristics of the cell nuclei found in the image[22].

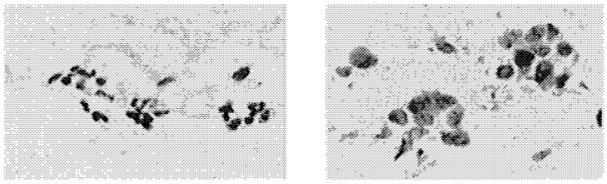

Figure 1: Image from [22] as cited by [23]. Digitized images of FNA: (a) Benign, (b) Malignant.

There are 569 data points in the dataset: 212 – Malignant, 357 – Benign. Accordingly, the dataset features are as follows: (1) radius, (2) texture, (3) perimeter, (4) area, (5) smoothness, (6) compactness, (7) concavity, (8) concave points, (9) symmetry, and (10) fractal dimension.

With each feature having three information[22]: (1) mean, (2) standard error, and (3) "worst" or largest (mean of the three largest values) computed. Thus, having a total of 30 dataset features.

2.3 Dataset Preprocessing

To avoid inappropriate assignment of relevance, the dataset was standardized using Eq. 1.

$$z = \frac{X - \mu}{\sigma} \tag{1}$$

where X is the feature to be standardized, μ is the mean value of the feature, and σ is the standard deviation of the feature. The standardization was implemented using `StandardScaler().fit_transform()` of `scikit-learn`[15].

2.4 Machine Learning (ML) Algorithms

This section presents the machine learning (ML) algorithms used in the study. The Stochastic Gradient Descent (SGD) learning algorithm was used for all the ML algorithms presented in this section except for GRU-SVM, Nearest Neighbor search, and Support Vector Machine. The code implementations may be found online at https://github.com/AFAgarap/wisconsin-breast-cancer.

2.4.1 GRU-SVM. We proposed a neural network architecture[4] combining the gated recurrent unit (GRU) recurrent neural network (RNN) and the support vector machine (SVM), for the purpose of binary classification.

$$z = \sigma(\mathbf{W}_z \cdot [h_{t-1}, x_t]) \tag{2}$$

$$r = \sigma(\mathbf{W}_r \cdot [h_{t-1}, x_t]) \tag{3}$$

$$\tilde{h}_t = tanh(\mathbf{W} \cdot [r_t * h_{t-1}, x_t]) \tag{4}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{5}$$

where z and r are the *update gate* and *reset gate* of a GRU-RNN respectively, $\tilde{h}_t$ is the candidate value, and h_t is the new RNN cell state value[8]. In turn, the h_t is used as the input in the L2-SVM classifier (see Eq. 19 in Section 2.4.6) of the network instead of the conventional Softmax classifier. Then, the Adam[13] algorithm was used to learn the parameters $\mathbf{W}$ to minimize the loss L of the model. The same learning algorithm was used for the implementation of SVM (Section 2.4.6) in this study.

2.4.2 Linear Regression. Despite an algorithm for regression problem, linear regression (see Eq. 6) was used as a classifier for this study. This was done by applying a threshold for the output of Eq. 6, i.e. subjecting the value of the regressand to Eq. 7.

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i \cdot \mathbf{x}_i \tag{6}$$

$$f\big(h_\theta(x)\big) = \begin{cases} 1 & h_\theta(x) \geq 0.5 \\ 0 & h_\theta(x) < 0.5 \end{cases} \tag{7}$$

To measure the loss of the model, the mean squared error (MSE) was used (see Eq. 8).

$$L(y, \theta, x) = \frac{1}{N} \sum_{i=0}^{N} \big((\theta_i \cdot \mathbf{x}_i) - y_i\big)^2 \tag{8}$$

where y represents the actual class, and $(\theta \cdot \mathbf{x})$ represents the predicted class. This loss is minimized through the use of SGD algorithm, which learns the parameters θ of Eq. 6. The same method of loss minimization could be said for MLP and Softmax Regression.

2.4.3 Multilayer Perceptron. The perceptron model was developed by Rosenblatt (1958)[17] based on the neuron model by McCulloch & Pitts (1943)[14]. The multilayer perceptron (MLP)[7] consists of hidden layers (composed by a number of perceptrons) that enables the approximation of any functions, that is, through activation functions such as *tanh* or *sigmoid* σ.

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i \mathbf{x}_i \tag{9}$$

$$f\big(h_\theta(x)\big) = \mathbf{h}_\theta(\mathbf{x})^+ = max(0, \mathbf{h}_\theta(\mathbf{x})) \tag{10}$$

For this study, the activation function used for MLP was ReLu[11] (see Eq. 10), while there were three hidden layers that each consists of 500 nodes (500-500-500 architecture). As for the loss, it was computed using the cross entropy function (see Eq. 14).

2.4.4 Nearest Neighbor. This is a form of an optimization problem that seeks to find the closest point $p_i \in \mathbf{p}$ to a query point $q_i \in \mathbf{q}$. In this study, both the L1 (Manhattan, Eq. 11) and L2 (Euclidean, Eq. 12) norm were used to measure the distance between $\mathbf{p}$ and $\mathbf{q}$.

$$\|\mathbf{p} - \mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i| \tag{11}$$

$$\|\mathbf{p} - \mathbf{q}\|_2 = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{12}$$

The code implementation was based on the work of Damien (2017)[10] in GitHub. A learning algorithm such as SGD and Adam[13] is not applicable to Nearest Neighbor search, as it is practically a geometric approach for classification.

2.4.5 Softmax Regression. This is a classification model generalizing logistic regression to multinomial problems. But unlike linear regression (Section 2.4.2) that produces raw scores for the classes, softmax regression produces a probability distribution for the classes. This is accomplished using the Softmax function (see Eq. 13).

$$P(\mathbf{y}_i \mid \mathbf{x}) = \frac{e^{h_\theta(x)_i}}{\sum_{i=0}^{n} e^{h_\theta(x)_i}} \tag{13}$$

$$L(y, \theta, x) = -\sum_{i=0}^{n} y_i \cdot log\big(h_\theta(x)_i\big) \tag{14}$$

The loss is measured by using the cross entropy function (see Eq. 14), where y represents the actual class, and $h_\theta(x)$ represents the predicted class.

2.4.6 Support Vector Machine. Developed by Vapnik[9], the support vector machine (SVM) was primarily intended for binary classification. Its main objective is to determine the optimal hyperplane $f(w, x) = \mathbf{w} \cdot \mathbf{x} + b$ separating two classes in a given dataset having input features $\mathbf{x} \in \mathbb{R}^P$, and labels $\mathbf{y} \in \{-1, +1\}$.

SVM learns by solving the following constrained optimization problem:

$$min \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{p} \xi_i \tag{15}$$

$$s.t\ y_i'(\mathbf{w}\cdot\mathbf{x}+b) \geq 1-\xi_i \tag{16}$$

$$\xi_i \geq 0, i = 1, ..., p \tag{17}$$

where $\mathbf{w}^T\mathbf{w}$ is the Manhattan norm, ξ is a cost function, and C is the penalty parameter (may be an arbitrary value or a selected value using hyper-parameter tuning). The corresponding unconstrained optimization problem (Eq. 15) is the following:

$$min\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{p} max\big(0, 1-y_i'(\mathbf{w}^T\mathbf{x}_i+b)\big) \tag{18}$$

where $\mathbf{w}^T\mathbf{x}_i + b_i$ is the predictor function. The objective of Eq. 18 is known as the primal form problem of L1-SVM, with the standard hinge loss. The problem with L1-SVM is the fact that it is not differentiable[19], as opposed to its variation, the L2-SVM:

$$min\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{p} max\big(0, 1-y_i'(\mathbf{w}^T\mathbf{x}_i+b)\big)^2 \tag{19}$$

The L2-SVM is differentiable and provides more stable results than its L1 counterpart[19]. However, for this study, the regularizer used was Eq. 20. Along with the said change, the hinge loss used was Eq. 21.

$$R(w) = \frac{1}{N}\sum_{i}^{N} w_i^2 \tag{20}$$

$$L = \frac{1}{p}\sum_{i=1}^{p} max\big(0, 1-y_i'(\mathbf{w}^T\mathbf{x}_i+b)\big)^2 \tag{21}$$

Eq. 20 is the L2-Norm, but with an amendment with its λ parameter of $\frac{1}{N}$. As for Eq. 21, its mean value is computed with the introduction of $\frac{1}{p}$, then it is multiplied by the penalty parameter C. Essentially still Eq. 19, but with different parameters: $R(w) + C \cdot L$.

2.5 Data Analysis

There were two phases of experiment for this study: (1) training phase, and (2) test phase. All the ML algorithms described in Section 2.4 were trained and tested on WDBC. The dataset was partitioned by 70% (training phase) / 30% (testing phase).

The parameters considered in the experiments were the following: (1) Test Accuracy, (2) Epochs, (3) Number of data points, (4) False Positive Rate (FPR, Eq. 24), (5) False Negative Rate (FNR, Eq. 25), (6) True Positive Rate (TPR, Eq. 22), and (7) True Negative Rate (TNR, Eq. 23).

$$TPR = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{22}$$

$$TNR = \frac{True\ Negative}{True\ Negative + False\ Positive} \tag{23}$$

$$FPR = 1 - TNR \tag{24}$$

$$FNR = 1 - TPR \tag{25}$$

3 RESULTS AND DISCUSSION

All experiments in this study were conducted on a laptop computer with Intel Core(TM) i5-6300HQ CPU @ 2.30GHz x 4, 16GB of DDR3 RAM, and NVIDIA GeForce GTX 960M 4GB DDR5 GPU.

Table 1 shows the manually-assigned hyper-parameters used for the ML algorithms. The hyper-parameters of a ML algorithm affects its performance in training, speed, and generalization capability. In this study, the hyper-parameters set were not obtained through hyper-parameter optimization / tuning, but they were set by hand. To determine the most optimal hyper-parameters, cross validation (CV) techniques such as k-fold cross validation must be used[6]. Unfortunately, this study was limited to the conventional dataset partitioning of 70/30, training dataset and testing dataset respectively.

Table 2 summarizes the experiment results. In addition to the reported results, the results from the studies by [18] and [23] are put into comparison. First, [18] implemented five ML algorithms for classification on WDBC. Namely, (1) Naive Bayes (NB), (2) MLP, (3) SVM, (4) K-Nearest Neighbor (KNN), and (5) Decision Tree J48[16]. The single classifier algorithms they used were implemented using WEKA[21], and their hyper-parameters and architecture (for MLP) were not specified. Their results have shown the following test accuracies: (1) NB had 92.9701%, (2) MLP had 96.6608%, (3) SVM had 97.71533%, (4) KNN had 95.9575%, and (5) Decision Tree J48 had 93.1459%. Then, [23] implemented a SVM with Gaussian Radial Basis Function (RBF) as its kernel for classification on WDBC. Their experiment revealed that the said SVM had its highest test accuracy of 89.28% with its free parameter $\sigma = 0.6$ in Eq. 26.

$$K(x, x') = exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) \tag{26}$$

Looking at Table 2, the ML algorithms used in this study may be considered on par with the findings of [18] and [23]. With MLP of $500-500-500$ architecture, and its hyper-parameters specified in Table 1, it reached a test accuracy of ≈99.04%, comparable to [18]. With a test accuracy of ≈96.09%, the L2-SVM lies in the between the findings of [18](WEKA[21] SVM, 97.71533%) and [23](Gaussian RBF, 89.28%). Meanwhile, the L2-NN is not far from the KNN in [18], with a test accuracy of ≈94.74%. Keeping in mind that the KNN is a generalization of NN that seeks to find K nearest points, while NN seeks to find the closest point $p_i \in \mathbf{p}$ to $q_i \in \mathbf{q}$.

Figure 2 shows the training accuracy of the ML algorithms: (1) GRU-SVM finished its training in 2 minutes and 54 seconds with an average training accuracy of 90.6857639%, (2) Linear Regression finished its training in 35 seconds with an average training accuracy of 92.8906257%, (3) MLP finished its training in 28 seconds with an average training accuraacy of 96.9286785%, (4) Softmax Regression finished its training in 25 seconds with an average training accuracy of 97.366573%, and (5) L2-SVM finished its training in 14 seconds with an average training accuracy of 97.734375%. There was no recorded training accuracy for Nearest Neighbor search since it does not require any training, as the norm equations (Eq. 11 and Eq. 12) are directly applied on the dataset to determine the "nearest neighbor" of a given data point $p_i \in \mathbf{p}$.

The empirical evidence presented in this section corroborates the findings of [18] and [23], and attests to the effectiveness of ML algorithms on the diagnosis of breast cancer. While the experiment

Table 1: Hyper-parameters used for the ML algorithms.

Hyper-parameters	GRU-SVM	Linear Regression	MLP	Nearest Neighbor	Softmax Regression	SVM
Batch Size	128	128	128	N/A	128	128
Cell Size	128	128	128	N/A	128	128
Dropout Rate	0.5	N/A	None	N/A	N/A	N/A
Epochs	3000	3000	3000	1	3000	3000
Learning Rate	1e-3	1e-3	1e-2	N/A	1e-3	1e-3
Norm	L2	N/A	N/A	L1, L2	N/A	L2
SVM C	5	N/A	N/A	N/A	N/A	5

Table 2: Summary of experiment results on the ML algorithms.

Parameter	GRU-SVM	Linear Regression	MLP	L1-NN	L2-NN	Softmax Regression	SVM
Accuracy	93.75%	96.09375%	99.038449585420729%	93.567252%	94.736844%	97.65625%	96.09375%
Data points	384000	384000	512896	171	171	384000	384000
Epochs	3000	3000	3000	1	1	3000	3000
FPR	16.666667%	10.204082%	1.267042%	6.25%	9.375%	5.769231%	6.382979%
FNR	0	0	0.786157%	6.542056%	2.803738%	0	2.469136%
TPR	100%	100%	99.213843%	93.457944%	97.196262%	100%	97.530864%
TNR	83.333333%	89.795918%	98.732958%	93.75%	90.625%	94.230769%	93.617021%

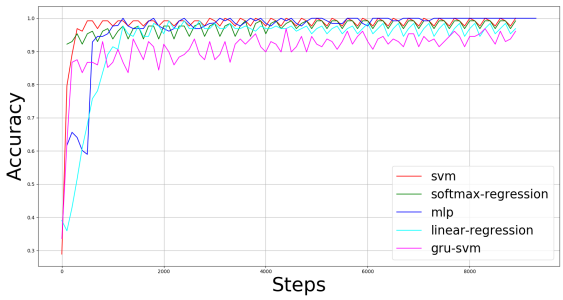

Figure 2: Plotted using `matplotlib`[12]. Training accuracy of the ML algorithms on breast cancer detection using WDBC.

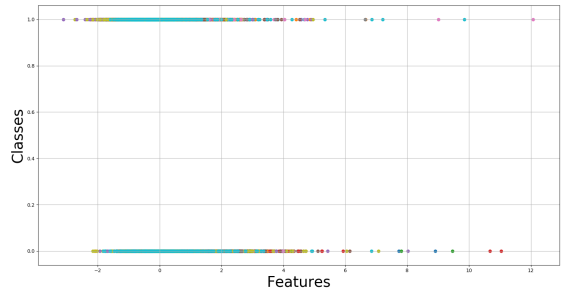

Figure 3: Plotted using `matplotlib`[12]. Scatter plot of the WDBC

results are all commendable, the performance of the GRU-SVM model proposed in [4] warrants a discussion. The mid-level performance of GRU-SVM with a test accuracy of 93.75% is hypothetically attributed to the following information: (1) the non-linearities introduced by the GRU model[8] through its gating mechanism (see Eq. 2 and Eq. 3) to its output may be the cause of a difficulty in generalizing on a linearly-separable data such as the WDBC dataset, and (2) the sensitivity of RNNs to weight initialization[5]. Since the weights of the GRU-SVM model are assigned with arbitrary values, it will also prove limited capability of result reproducibility, even when using an identical configuration[5].

Despite the given arguments, it does not necessarily revoke the fact that GRU-SVM is comparable with the presented ML algorithms, as what the results have shown. In addition, it was an expectation that the upper hand goes to the linear classifiers (Linear Regression and SVM) as the utilized dataset was linearly separable (see Figure 3).

4 CONCLUSION AND RECOMMENDATION

This paper presents an application of different machine learning algorithms, including the proposed GRU-SVM model in [4], for the diagnosis of breast cancer. All ML algorithms presented in Section 2.4 exhibited high performance (all exceeded 90% test accuracy) on the binary classification of breast cancer, i.e. determining whether benign tumor or malignant tumor. Consequently, the statistical measures on the classification problem were also satisfactory.

To further substantiate the results of this study, a CV technique such as k-fold cross validation should be implemented in the application of the specified ML algorithms. The application of such a technique would prove to be prolific as not only will it provide a more accurate measure of model prediction performance, but it will also assist in determining the most optimal hyper-parameters

for the ML algorithms[6]. Finally, dimensionality reduction techniques such as principal component analysis (PCA) may prove to be beneficial on the preprocessing of dataset, as evident from the findings of [18].

5 ACKNOWLEDGMENT

An expression of gratitude to Dr. William H. Wolberg of the University of Wisconsin for the WDBC dataset used in this study. Also, an appreciation of the open source community, especially Cross Validated, GitHub, Google, Python, and Stack Overflow.

REFERENCES

[1] [n. d.]. ([n. d.]). https://seer.cancer.gov/statfacts/html/breast.html
[2] 2017. Cancer Statistics. (Mar 2017). https://www.cancer.gov/about-cancer/understanding/statistics
[3] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). http://tensorflow.org/ Software available from tensorflow.org.
[4] Abien Fred Agarap. 2017. A Neural Network Architecture Combining Gated Recurrent Unit (GRU) and Support Vector Machine (SVM) for Intrusion Detection in Network Traffic Data. *arXiv preprint arXiv:1709.03082* (2017).
[5] Abdulrahman Alalshekmubarak and Leslie S Smith. 2013. A novel approach combining recurrent neural network and support vector machines for time series classification. In *Innovations in Information Technology (IIT), 2013 9th International Conference on*. IEEE, 42–47.
[6] Yoshua Bengio, Ian J Goodfellow, and Aaron Courville. 2015. Deep learning. *Nature* 521 (2015), 436–444.
[7] Christopher M Bishop. 1995. *Neural networks for pattern recognition*. Oxford university press.
[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[9] C. Cortes and V. Vapnik. 1995. Support-vector Networks. *Machine Learning* 20.3 (1995), 273–297. https://doi.org/10.1007/BF00994018
[10] Aymeric Damien. 2017, August 29. (2017, August 29). https://github.com/aymericdamien/TensorFlow-Examples/blob/master/examples/2_BasicModels/nearest_neighbor.py Accessed: November 17, 2017.
[11] Richard HR Hahnloser, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas, and H Sebastian Seung. 2000. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature* 405, 6789 (2000), 947–951.
[12] J. D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering* 9, 3 (2007), 90–95. https://doi.org/10.1109/MCSE.2007.55
[13] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[14] Warren S McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* 5, 4 (1943), 115–133.
[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[16] J Ross Quinlan. 1993. C4. 5: Programming for machine learning. *Morgan Kauffmann* 38 (1993).
[17] Frank Rosenblatt. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review* 65, 6 (1958), 386.
[18] Gouda I Salama, M Abdelhalim, and Magdy Abd-elghany Zeid. 2012. Breast cancer diagnosis on three different datasets using multi-classifiers. *Breast Cancer (WDBC)* 32, 569 (2012), 2.
[19] Yichuan Tang. 2013. Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239* (2013).
[20] Stéfan van der Walt, S Chris Colbert, and Gael Varoquaux. 2011. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering* 13, 2 (2011), 22–30.
[21] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. 2016. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.
[22] William H Wolberg, W Nick Street, and Olvi L Mangasarian. 1992. Breast cancer Wisconsin (diagnostic) data set. *UCI Machine Learning Repository [http://archive.ics. uci. edu/ml/]* (1992).
[23] Elias Zafiropoulos, Ilias Maglogiannis, and Ioannis Anagnostopoulos. 2006. A support vector machine approach to breast cancer diagnosis and prognosis. *Artificial Intelligence Applications and Innovations* (2006), 500–507.