TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

REPORT PROJECT

MACHINE LEARNING

Project title: Breast Cancer Detection

Professor : Thân Quang Khoát
Subject : IT3190E
Class : 141183

Group: 07

Students: Hoàng Duy Anh - 20214944
Lê Đức Dũng - 20214952
Cao Gia Khánh - 20214962
Nguyễn Phương Thảo - 20214973

Hà Nội, 07-2023

1 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[20] 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[20]. For the implementation of the ML algorithms, the dataset was partitioned in the following fashion: 70 percents for training phase, and 30 percents 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 percents test accuracy) on the classification task. The MLP algorithm stands out among the implemented algorithms with a test accuracy of 99.04 percents.

1.1 CCS CONCEPTS

Computing methodologies -> Supervised learning by classification; Supervised learning by regression; Support vector machines; Neural networks;

2 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 percents 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[17, 21], where they proposed the use of machine learning (ML) algorithms for the classification of breast cancer using the Wisconsin Diagnostic Breast Cancer (WDBC) dataset[20], 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[20]

3 Methodology

3.1 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[19], and scikit-learn[15].

3.2 The Dataset

The machine learning algorithms were trained to detect breast cancer using the Wisconsin Diagnostic Breast Cancer (WDBC) dataset[20]. According to [20], 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[20].

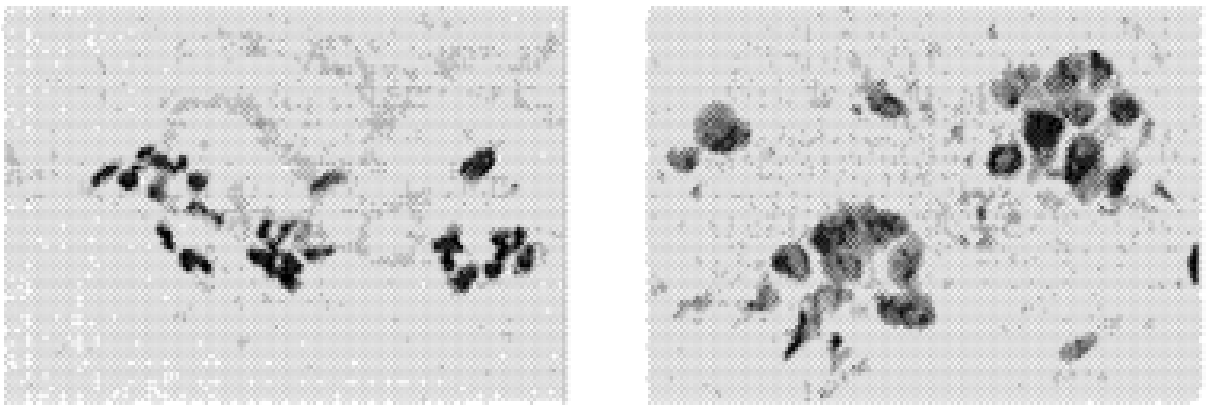

Figure 1: Image from [20] as cited by [21].
Digitized images of FNA: (a) Benign, (b) Malignant

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[20]: (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

3.3 Dataset Preprocessing

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

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

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

3.4 Machine Leaning (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

3.4.1 Logistic Regression

It works by modeling the probability of a binary outcome (e.g. whether a patient has a particular disease or not) based on one or more predictor variables (e.g. age, gender, blood pressure, etc.). The output of logistic regression is a probability value between 0 and 1, which can then be thresholded to make a binary prediction. The logistic regression function is defined as follows:

$$p(y = 1|x) = \frac{1}{1 + e^{-z}} \tag{2}$$

where: p(y=1|x) is the probability of the positive class given the input variables x

$$z = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

is the linear combination of the input variables and their coefficients, where b0 is the intercept and bi (i=1,2,...,n) are the coefficients of the input variables The logistic function is also known as the sigmoid function, which maps any real-valued number to a probability value between 0 and 1. The sigmoid function is defined as:

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

where z is the linear combination of the input variables and their coefficients as described above. The logistic regression algorithm involves finding the optimal values of the coefficients $b_0, b_1, b_2, ..., b_n$ that minimize the difference between the predicted probabilities and the actual binary labels in the training data. This is typically done using an optimization algorithm such as gradient descent. In addition to the standard logistic regression algorithm, there are several variations that can handle more complex classification problems, such as multiclass logistic regression and regularized logistic regression. Multiclass logistic regression extends the binary logistic regression to handle more than two classes, while regularized logistic regression adds a penalty term to the cost function to prevent overfitting on the training data.

3.4.2 K-Nearest Neighbor

K-nearest neighbor (KNN) is a popular algorithm in machine learning for solving classification and regression problems. It is a non-parametric algorithm, meaning that it does not make any assumptions about the underlying distribution of the data. Instead, it works by finding the K nearest examples in the training data to a given test example and using these neighbors to make a prediction. The K-nearest neighbor function for classification is as follows: Given a set of training examples $X = x_1, x_2, ..., x_n$ with corresponding class labels $Y = y_1, y_2, ..., y_n$, and a new test example x_q, the K-nearest neighbor algorithm works as follows: Calculate the distance between x_q and all training examples using a distance metric such as Euclidean distance:

$$d(x_q, x_i) = \sqrt{\sum_{j=1}^{m} (x_{qj} - x_{ij})^2}$$

where m is the number of features in the data. Select the K nearest neighbors of x_q based on their distances. Assign the class label of x_q as the majority class label among its K nearest neighbors.

The K-nearest neighbor function for regression is similar, where instead of assigning the class label of the majority of the K nearest neighbors, the algorithm assigns the average value of the target variable among the K nearest neighbors as the predicted value for the test example.

The choice of the value of K is an important hyperparameter in KNN. A small value of K can lead to overfitting, while a large value of K can lead to underfitting. In practice, the value of K is chosen using cross-validation or grid search.

K-nearest neighbor is a simple and intuitive algorithm that is easy to implement and can work well for low-dimensional data with clear decision boundaries. However, it can be sensitive to the choice of distance metric and the curse of dimensionality, where accuracy can decrease as the number of features increases.

3.4.3 Random Forest

Random forest is a popular algorithm in machine learning for solving classification and regression problems. It is an ensemble learning method that combines multiple decision trees to make predictions. The random forest function for classification is as follows: Given a set of training examples $X = x_1, x_2, ..., x_n$ with corresponding class labels $Y = y_1, y_2, ..., y_n$, the random forest algorithm works as follows: Randomly select a subset of the training examples and features. Build a decision tree using the selected subset of data. Repeat steps 1-2 multiple times to build a forest of decision trees. To make a prediction for a new test example, pass it through all the decision trees in the forest and assign the class label with the highest frequency among the predictions. The random forest function for regression is similar, where instead of assigning the class label with the highest frequency, the algorithm assigns the average value of the target variable among the predictions. The random forest algorithm has several advantages over a single decision tree. First, it can handle high-dimensional data with complex decision boundaries. Second, it is less prone to overfitting compared to a single decision tree, as it combines multiple trees which reduces the variance of the model. Third, it can provide feature importance scores which can be useful for feature selection. The hyperparameters of the random forest algorithm include the number of trees in the forest, the depth of the trees, and the number of features to consider at each split. These hyperparameters can be tuned using cross-validation or grid search.

3.4.4 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) = w · x +b separating two classes in a given dataset having input features $x \in \mathbb{R}^p$, and labels $y \in \{-1, +1\}$. SVM learns by solving the following constrained optimization problem:

$$min \frac{1}{p} W^T W + C \sum_{i=1}^{P} \xi \tag{3}$$

$$s.t y_i'(w.x + b) \geq 1 - \xi_i \tag{4}$$

$$\xi_i \geq 0, i = 1, ..., p \tag{5}$$

where $w^T w$ 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). Its

corresponding unconstrained optimization problem is the following:

$$min\frac{1}{p}W^TW + C\sum_{i=1}^{P} max(0, 1 - y_i'(w_i x_i + b)) \tag{6}$$

where wx + b is the predictor function. The objective of Eq. 19 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[18], as opposed to its variation, the L2-SVM:

$$min\frac{1}{p}||w||_2^2 + C\sum_{i=1}^{P} max(0, 1 - y_i'(w_i x_i + b))^2 \tag{7}$$

The L2-SVM is differentiable and provides more stable results than its L1 counterpart[18].

3.5 Data Analysis

There were two phases of experiment for this study: (1) training phase, and (2) test phase. The dataset was partitioned by 70 percents (training phase) / 30 percents (testing phase). The parameters considered in the experiments were as follows: (1) Test Accuracy, (2) Epochs, (3) Number of data points, (4) False Positive Rate (FPR), (5) False Negative Rate (FNR), (6) True Positive Rate (TPR), and (7) True Negative Rate (TNR).

4 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. Table 2 summarizes the experiment results. In addition to the reported results, the result from [21] was put into comparison. [21] implemented the SVM with Gaussian Radial Basis Function (RBF) as its kernel for classification on WDBC. Their experiment revealed that their SVM had its highest test accuracy of 89.28 percents with its free parameter $\sigma = 0.6$. However, their experiment was based on a 60/40 partition (training/testing respectively). Hence, we would not be able to draw a fair comparison between the current study and [21]. Comparing the results of this study on an intuitive sense may perhaps be close to a fair comparison, recalling that the partition done in this study was 70/30. With a test accuracy of ≈ 96.09 percents , the L2-SVM in this study bares superiority against the findings of [21] (SVM with Gaussian RBF, having a test accuracy of 89.28 percents). But then again, it was based on a higher training data of 10 percents (70 percents vs 60 percents). Figure 2 shows the training accuracy of the ML algorithms: (1)

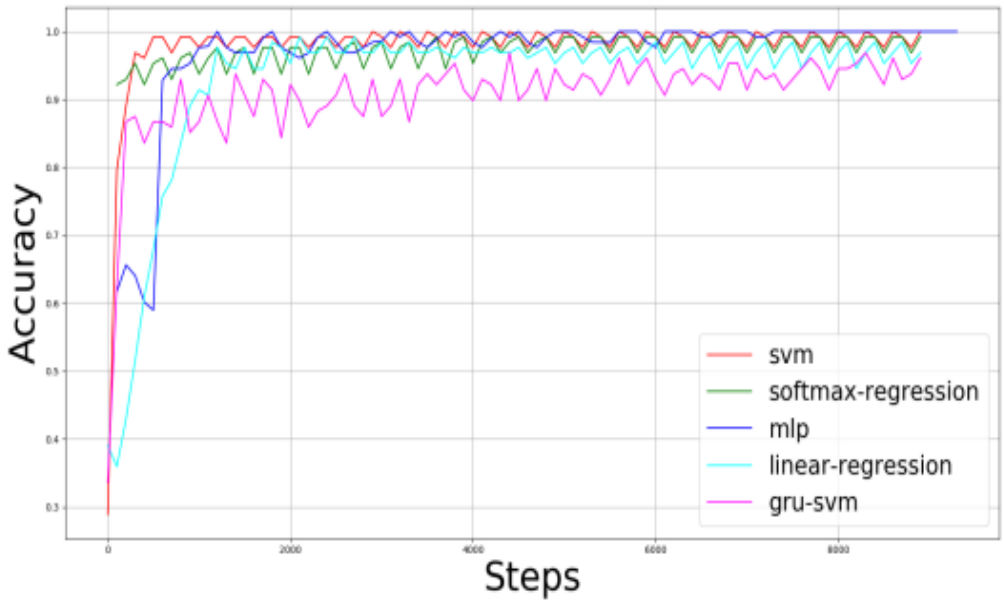

Figure 2: Plotted using matplotlib[12]. Training accuracy of the ML algorithms on breast cancer detection using WDBC.

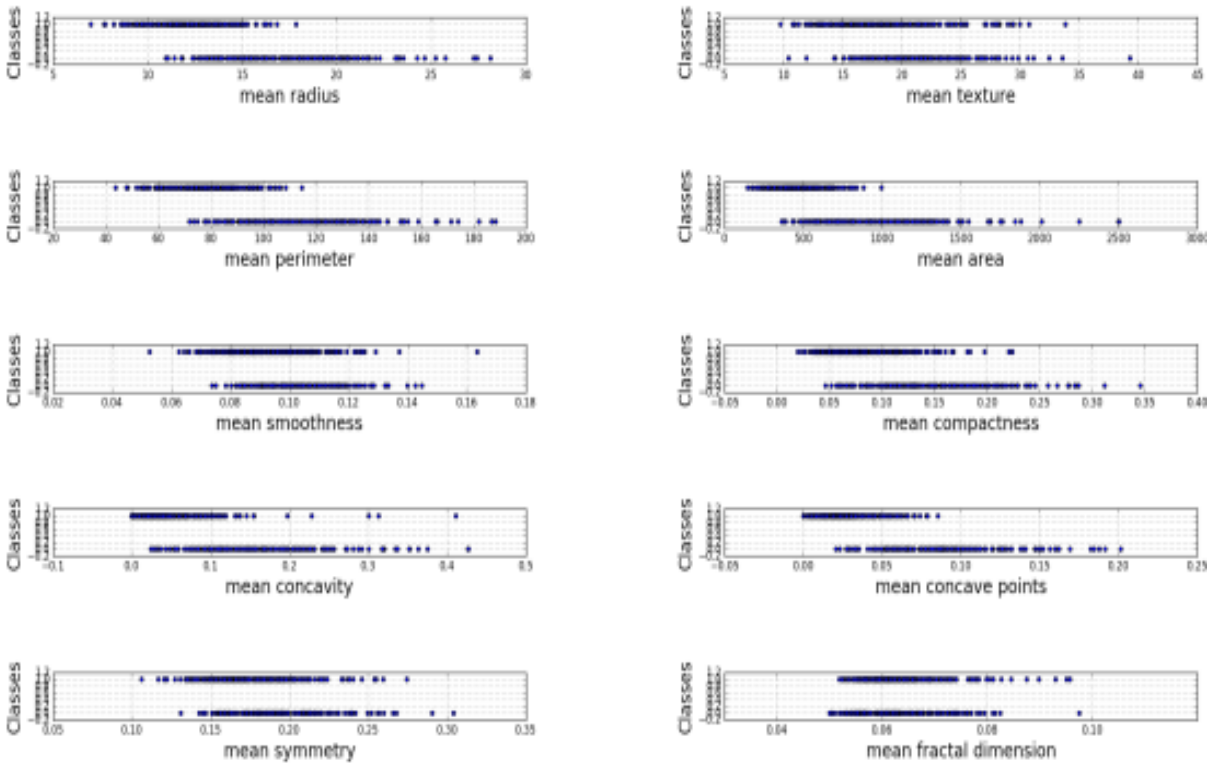

Figure 3:plotted using matplotlib[12]. Scatter plot of mean features (x_0 - x_9) in the WDBC.

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

Hyper Parameters	GRU-SVM	Linear Regression	MLP	Nearest Neighbor	Softmax Regression	SVM
Batch Size						
Cell Size						
Dropout Rate						
Epochs						
Learning Rate						
Norm						
SVM C						

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

Parameter	GRU-SVM	Linear Regression	MLP	L1-NN	L2-NN	Software Regression	SVM
Accuracy							
Data points							
Epochs							
FPR							
FNR							
TPR							
TNR							

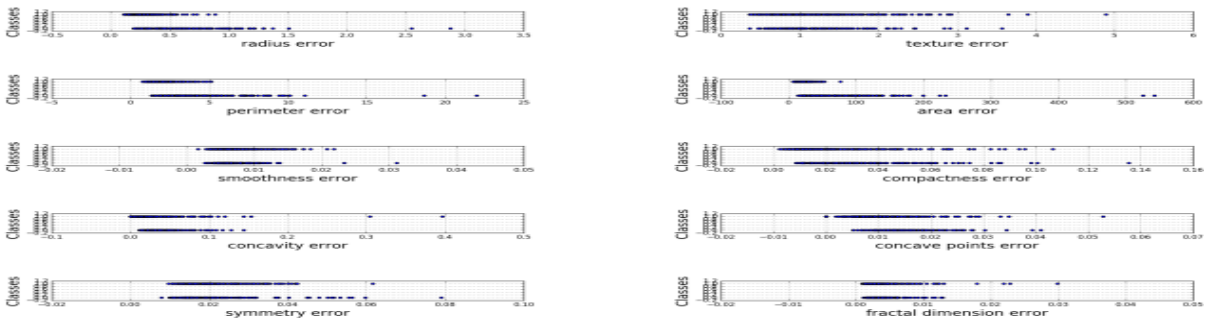

Figure 4: Plotted using matplotlib[12]. Scatter plot of error features (x_10 - x_19) in the WDBC

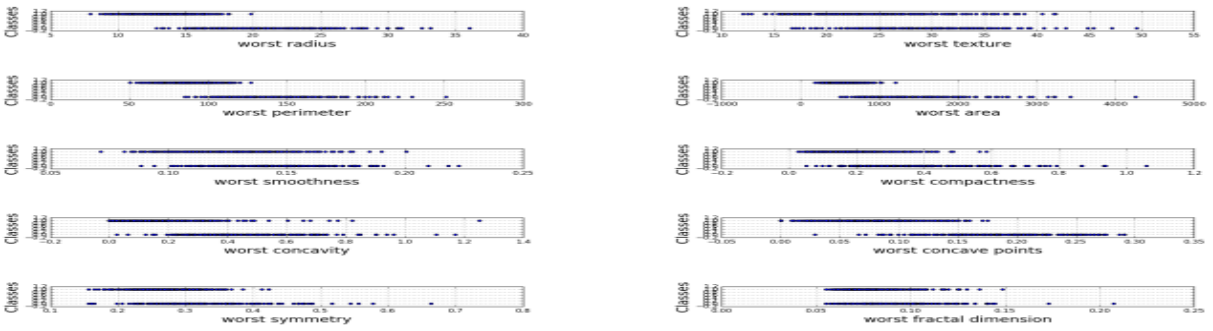

Figure 5: Plotted using matplotlib[12]. Scatter plot of worst features (x_20 - x_29) in the WDBC.

5 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 presented ML algorithms exhibited high performance 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 employed. The application of such a technique will not only 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].