Machine Learning Nanodegree

Capstone project: XGBoost models for cancer diagnoses in Wisconsin breast cancer data set

Christina Kallendorf

August 19th, 2020

DEFINITION

Project overview

Machine Learning opens up a vast spectrum of medical applications. Some examples are medical image processing for diagnosis, classification of medical findings as well as disease prognosis and risk assessment [1]. Sidey-Gibbons claim that machine learning can be highly beneficial to the areas of diagnosis and outcome prediction in medical applications [2]. In particular, Machine Learning can be supportive to diagnose and classify specific sub-types of cancer from biopsy probes. In this context, Machine Learning can lead to an "effective and accurate decision making", as the authors of [3] point out. This may help to diagnose malignant tumors early and better define a suitable treatment. As pointed out by [13], this may help to "increase survival rates from 56% to more than 86%".

The project proposed here takes up one classification example of the area of cancer research and focuses on the identification of malignant breast tumors based on cytology features from biopsies. It employs a data set of breast tissue biopsies that was collected by University of Wisconsin Hospitals, Madison, by Dr. William H. Wolberg [4, 5, 6, 7] from tumor patients during the years of 1989 till 1991. This data set comprises eleven cytology characteristics of breast fine-needle aspirates. Specifically, the cytology features presented in this data where computed from images of biopsies, using methods developed in [4-6]. It comprises 699 data samples with nine cytology features, Ids and the classification of benign tissue, by value 0, and malignant, i. e. cancer, tissue by value 1. Note that feature extraction is not subject of this project and features are provided with the data.

Problem statement

Based on a medical patient's biopsy, it needs to be determined whether the investigated tissue contains malignant, i. e. cancer cells. In order to react to the growth of cancer cells quickly and appropriately, it is necessary to define such cells at an early state. Usually, biopsy results are classified by laboratory staff, based on their investigation of cell properties under a microscope. Therefore, a diagnosis is subject to individual judgement and human error [2].

A Machine Learning model can be able to identify breast cancer reliably for a given sample. It is a supervised binary classification that is itself based on a defined set of cytology features from biopsies, such as shape and size. As a result, cytology features obtained from biopsies enable a Machine Learning model to provide an effective, accurate and fast diagnosis. This has advantages over a subjective diagnosis generated by laboratory staff, see also [2], and may help physicians in decision-making, refer to [13].

In recent years, extreme gradient boosting (XGBoost) algorithms have become popular, giving "state-of-the-art results on many standard classification benchmarks" [14], while succeeding most popular algorithms in speed and scaling well to high data volumes, as the authors of [14] point out. Motivated by the fact, that XGBoost models have been used by many price-winners in machine learning competitions, such as Kaggle or KDDCup in 2015 [14], I intend to create an XGBoost model for classifying the Wisconsin Breast Cancer Dataset introduced above. This project is guided by one central question: How does an XGBoost model compare to models presented in literature previously for supervised binary classification of the Wisconsin Breast Cancer data set (WBCD)?

Definition of evaluation metrics

Using machine learning for diagnostic classification, major objectives should be minimization of false positives and false negatives, while optimizing accuracy of the model. As suggested by [2], results presented here are measured and optimized for the following metrics,

- Sensitivity=true positives/(true positives + false negatives), i. e. the clinical term for recall.
- Specificity=true negatives/(true negatives + false positives),
- Accuracy= (true positives + true negatives) / total predictions.

ANALYSIS

Data exploration

The original data is published at https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29 , please refer to [8]. The data set employed here was obtained from https://www.kaggle.com/johnyquest/wisconsin-breast-cancer-cytology-features as a formatted .csv file .

	id	thickness	size	shape	adhesion	single	nuclei	chromatin	nucleoli	mitosis	class
0	1000025	5	1	1	1	2	1.0	3	1	1	0
1	1002945	5	4	4	5	7	10.0	3	2	1	0
2	1015425	3	1	1	1	2	2.0	3	1	1	0
3	1016277	6	8	8	1	3	4.0	3	7	1	0
4	1017023	4	1	1	3	2	1.0	3	1	1	0

Fig. 1: Sample of original data records

In preparation for training and testing, the Wisconsin Breast Cancer data set (WBDC) is imported as `DataFrame`. Some sample records are displayed by image Fig. 1 . Duplicates as well as rows with null values are removed from the data records, resulting in a set of 675 samples, out of which 236, i. e. 35%, are labeled as malignant and 439, i. e. 65% are labeled as benign. The column with IDs of original aspirates is removed, as they do not contain relevant information to classifying any data.

The samples comprise the following cytology characteristics of the fine needle aspirates with values ranging from 1 to 10:

- Clump thickness,

- Uniformity of cell size,
- Uniformity of cell shape,
- Marginal adhesion,
- Single epithelial cell size,
- Bare nuclei,
- Bland chromatin,
- Normal nucleoli,
- Mitoses.

Exploratory visualization

In order to give a first overview of the distribution of the respective data samples, the method `plot_feature_distribution` systematically creates scatter plots of the given data with respect to two indicated features. It is applied to all features pairwise. Some of the results are presented in Fig. 3; purple spots indicate samples of benign probes, yellow marks malignant cases. These plots indicate that in benign cases, most of the feature values concentrate in the left lower quadrant, while malignant data points scatter throughout the outer area. However, many outliers illustrate that neither of the features is a clear marker for malignant or benign cases. Note that the presented plots do not allow for conclusions regarding the frequency of these combinations. Furthermore, some data sets may occupy the same pair of values, which cannot be sufficiently drawn in this plot.

Furthermore, the feature's correlation matrix (i. e. two components' variances normalized by the product of standard variances) is determined and illustrated by Fig. 3 below.

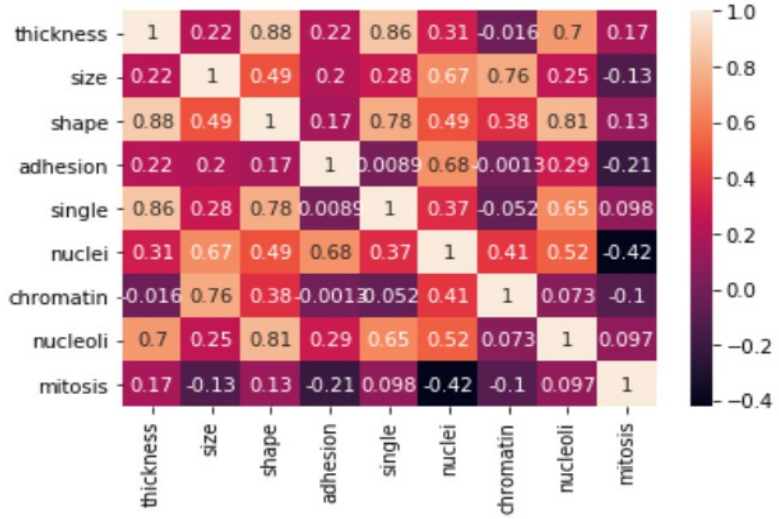

Fig. 2: Heatmap of correlation values of all features

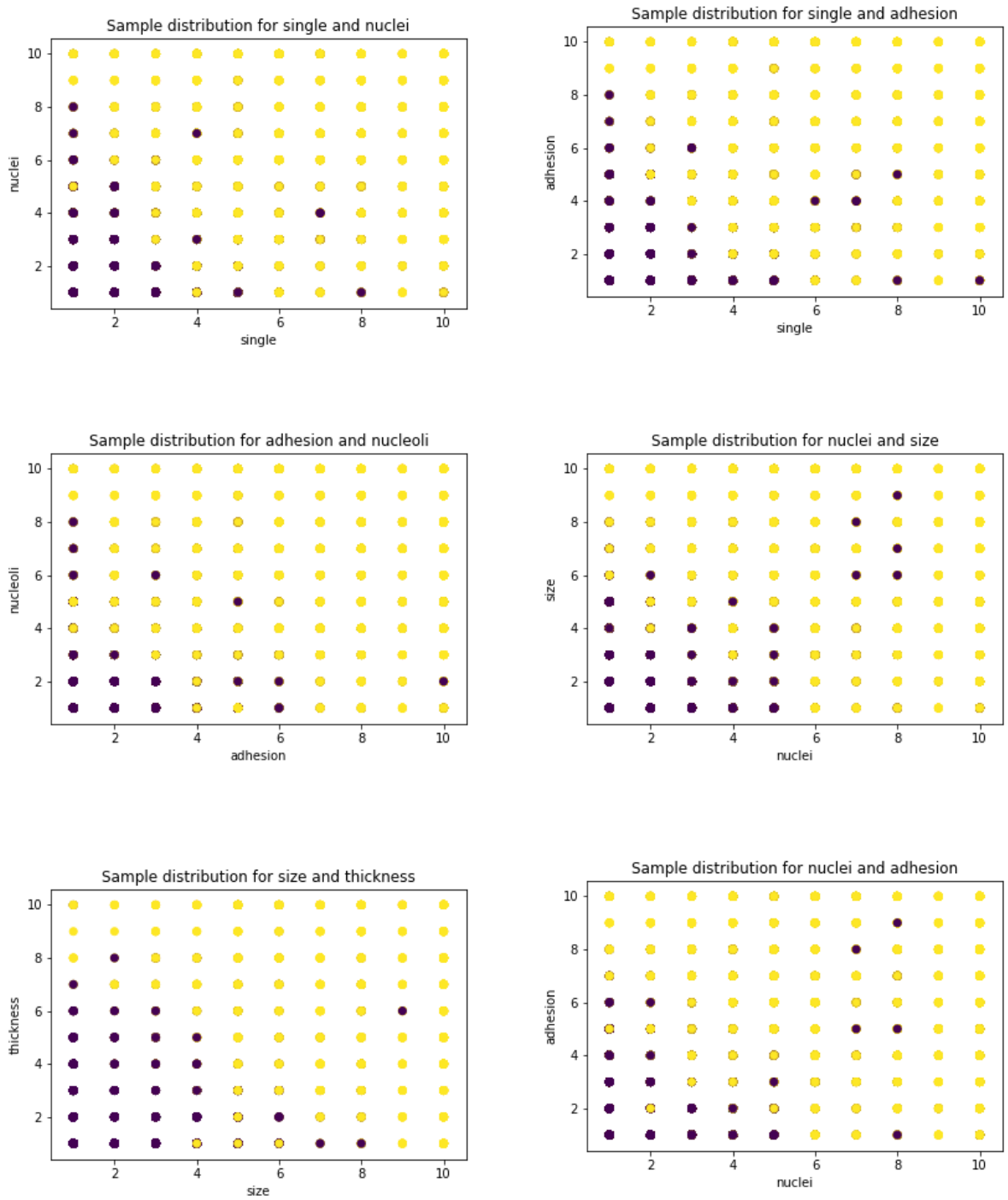

Fig. 3: Scatter plot of some of the feature distributions. Purple: benign samples, yellow: malignant samples.

The matrix indicates that neither of these features is sufficiently described by any other and therefore, could be simply omitted. However, few features prove to be at least highly correlated with a correlation greater than 80%, which are, in detail:

- shape and thickness,
- shape and nucleoli,
- single and thickness.

Algorithms and Techniques

XGBoost will be employed for building a supervised binary classifier of cytologoy features.

What again is XGBoost? Let us shortly recapitulate:

XGBoost is a learning algorithm that applies boosting to both linear and tree learners. Specifically, boosting implies that an optimal predictor is constructed incrementally from a sum of weaker predictors, known as weak learners, using a greedy algorithm. Each new predictor is obtained by adding the previously predictors to the current predictor and optimizing the weights of this linear combination, which are basically obtained from partial derivatives, i. e. the gradient of the loss function [16]. Extreme gradient boosters can be seen as a new generation focusing on high-performance computing, by providing scalability and speed increase and supporting distributed computing, also refer to [14].

XGBoost is available as open source algorithm, for instance, in Python via the XGBoost package, as well as an built-in managed algorithm on commercial machine learning platforms such as SageMaker.

Benchmarks

The data set used here has previously been employed for machine learning classification studies, in particular using SVM as well as Neural Networks. For further details, refer, for instance, to references [8-13]. In particular, results presented by [2] will be used as a benchmark. Specifically, I want to investigate whether an XGBoost model really outperforms an SVG model, which achieves sensitivity of 97%, specificity of 94% and an accuracy of 96%?

METHOLOGY

Data pre-processing

After removal of duplicates, rows containing null values and ID column, data records are randomly shuffled, as the provided data set is ordered by creation time of the fine needle aspirates. It is divided into 50% training , 25% test and 25% validation data. Labels, i. e. the classes of the sample, are separated. Next, the function data_as_txt is used to prepare data as csv files with labels as first column, followed by all features of interest, upload these files to S3 (default) bucket and return the s3-path for further use.

Implementation & Refinement

Setting up, training, optimizing and deployment of the XGBoost classifier will be done in two steps .

In a first step, the different types of boosters are evaluated using the xgboost open library. For this purpose, either an `XGBRegressor or an XGBClassifier` from `xgboost` library can be used. Note that the objective should be set to binary hinge, which is suitable for 1-0 classifications, or in other

words, true-false classifications. As a remark, this library is not available per default to Python kernels, but can still be installed locally.

Boosters and their variants are not part of a standardized hyper-parameter tuning. Specifically, this includes

- type of booster,
- tree-method, if a tree-booster is applied,
- as well as the evaluation metric

used for building the model. Due to the data set's small size, computation can be completed fast; prediction does not require a separate deployment to a computational machine. For this part of optimization, standard values are kept for all other hyper-parameters.

Performance of the model is evaluated with respect to recall, specificity and accuracy. The method `eval_metrics` implements the computation of each of these evaluation metrics, based on the test labels and the predicted labels of the test data. Furthermore, it documents results as well as the hyper-parameters of the estimators in a *metrics.csv* file. In this way, results are easily accessible later on.

With respect to recall and specificity, results in the below table illustrate that the tree-boosters(gbtree and dart), applied with an exact greedy algorithm succeed the approximate as well as the faster histogram optimized approximate greedy algorithms, as well as the linear booster, independent of the applied evaluation metrics. With respect to accuracy, a linear booster performs slightly better.

Booster	**gbtree**		**dart**		**gblinear**
tree_method	Auto/exact	Approx/hist	auto/exact	approx/hist	N/a
eval_metric	mae, rmse, error, mlogloss	mae, rmse, error, mlogloss	mae, rmse, error, mlogloss	mae, rmse, error, mlogloss'	mae, rmse, error, mlogloss
Recall	0.97222222	0.94444444	0.97222222	0.94444444	0.95833333
Specificity	0.9787234	0.95918367	0.978723	0.95918367	0.96938776
Accuracy	0.95857988	0.95857988	0.95857988	0.95857988	0.9704142

2. In a second step, the tuning of further relevant hyper-parameters is performed. For this purpose, Amazon SageMaker 's HyperparameterTuning library is employed. The tuning is based on an `XGBoost` estimator from Amazon SageMaker library*;* it is used as framework with a customized training script as entry point. The customized training script, available at `source/train.py`, provided as entry point is a simple preparation for additional tasks which could arise in future, for example, regarding data processing. This estimator is set up with optimal parameters regarding recall and specificity. The original

estimator is instantiated with booster `gbtree`, using the `exact` tree method and `rmse` as evaluation metrics.

One disadvantage of using a gradient boosting algorithm is the variety of hyperparameters which impact the quality of the model. For a systematic checkup of combinations of different hyperparameters, sagemaker hyperparameter tuning is employed for 20 combinations on 4 machines in parallel. The following parameters are selected by Bayesian optimization, which selects values from hyper-parameter space based on optimization:

- numerical rounds, out of an integer range from 40 to 200,
- maximal depth, out of an integer range from six to twelve,
- eta and gamma, out of continuous parameter ranges in an interval from 0 to 0.4, and 0 to 1, respectively.

The model with optimal hyper-parameters resulting from hyper-parameter training job is finally attached and deployed. This is then used for evaluating our test data.
All combinations of hyper-parameters that were used by the Bayes algorithms are available in SageMaker in the category 'Hyperparameter tuning job'. In detail, the optimal combination of this job was given by:

- eta: 0.2572110893586683,
- gamma: 0.5662623155210383,
- max_depth: 10
- num_round: 135.

Implementing XGBoost with a customized training script as entry point was particularly challenging with respect to data input. Input as .csv file was uncritical in when running the hyper-parameter tuning, as the fit function allows to define file format as well as line split. However, after deployment of the optimal model, the predict function does not provide for the further parameters. Matrix as input format was rejected for security reasons.

For this reason, I implemented an additional function prediction_labels. The function expects a path to a data file, from which it reads records per line. It calls the predict function and appends the results to one array.

Another aspect covered by implementation is the idea of component reduction, as three pairs of four features are correlated by more than 80%.This may imply that two features are redundant. This is interesting to investigate, as a high correlation of features can have a negative impact on performance of the algorithms employed.

Against this background, I utilize a PCA Analysis for reduction of component space. PCA is applied with seven components and the data transformed After that, an XGBoost model with the previously determined hyper-parameters is trained, deployed and tested, still leading to good results. Note, in order to account for the change in data, another round of hyper-parameter tuning makes sense here.

RESULTS

Model evaluation

The developed XGBoost model achieves an the following results with respect to the initially defined classification metrics:

- Accuracy: 0.9763313609467456,
- Specificity: 0.9894736842105263
- Recall: 0.9861111111111112.

Results were obtained based on 25% randomly selected records of the original data that was saved as testing material, where class labels have been removed.

Hence, the presented XGBoost model exceeds all models given by [2], in detail GLM, SVM and a neural network (ANN) model with respect to each of the evaluation metrics. Details of comparison are displayed in the results table presented below.

	GLM (ref. [2])	SVM (ref. [2])	ANN (ref. [2])	XGBoost (gbtree)	XGBoost (gbtree) with PCA reduction to 7 components
Recall	0.99	0.97	0.99	0.99	0.96
Specificity	0.87	0.94	0.86	0.99	0.97
Accuracy	0.95	0.96	0.94	0.98	0.97

Finally, I would like to discuss on correlation of features. The initial data exploration indicated that there is a correlation of more than 80% between the features shape with both thickness and nucleoli, as well as single and thickness. Does this imply that data records can be reliably classified using less features?

This still leads to surprisingly high results, with an accuracy of 97%, specificity of 97% and recall of 96%. With respected to specificity and accuracy, this is still an improvement as compared to the reference models. For additional improvement, a separate hyper-parameter tuning for the reduced input data should be conducted as well.

Justification

In this context, I would like to highlight , that first of all, my original hypotheses has been positively confirmed: XGBoost does not only outperform the SVM model presented by [2], but also each of the other models presented by [2]. As the reference models have been built and evaluated with respect to the same input data, this comparison is well justified.

Second, as per the results enlisted earlier, initial model optimizations have helped to achieve almost ideal results with respect to specificity, recall and accuracy. This was accomplished by comparison of different boosters and their relevant settings, such as the employed tree method, and of different hyper-

parameters. This constitutes a particular advantage of XGBoost as compared, for example to neural networks. A given set of parameters can be optimized by means of fairly automated processing; while the setup of a neural network rather represents a “black box”, with a wide variety of parameters, such as the number of layers, needs to be determined rather experimentally.

CONCLUSION

Free-form visualization, Reflection

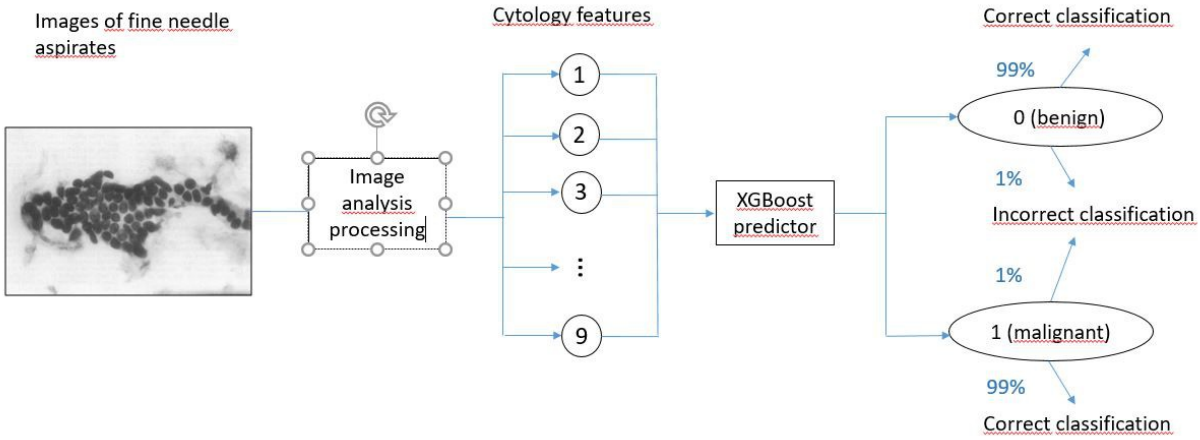

Fig. 4: Process of classifying the fine needle aspirates. Image is retrieved from [2].

In the project presented here, a supervised binary classifier for breast cytology features was developed and optimized. Features used for computation were originally extracted from images of fine needle aspirates of breast tissues using image analysis software [4, 5, 6, 7] . The classifier provides the second part of processing of the biopsies. As Fig. 4 illustrates, the XGBoost predictor presented is able to classify benign and malignant tumors excellently with an approximate 1% incorrect classification in either of the cases. Taking a look at medical testing, 99% of sensitivity is often the start of the range for clinical scenarios.

Achieving high sensitivity and specificity is a desirable target for medical applications , as one always strives to minimize false positives or false negatives in medical diagnosis.

Finally, two critical aspects need to be considered:

First of all, train and validation data set comprise a rather small number of samples, considering in particular the number of variants, which are possible for all 9 features and the high number of outliers. This may imply that results are 'incidentally' good just for a small range of test data used here, but would not hold for a larger set of data. However, at least when comparing to previous investigations of the same data set, one may argue that these results had the same limitation on data.

Second, it is well-known that tree-learning methods are sensitive to over-fitting. A higher volume of test data records would be required in order to investigate on this aspect.

Improvement

Further improving results with respect to evaluation metrics will be rather challenging to achieve. As mentioned before, neural networks offer more varieties in structure and parameters, so that it may be

even possible to optimize this model further. This , however, requires much patience with an experimental investigation of performance.

From image Fig. 4 is becomes clear that this binary classification is enabled by previous (automated) analysis of the original images. In fact, we do not have any information about the loss of precision which this processing step causes. This might be a better option for obtaining better results.

Assuming that the original image data files are accessible, a learning model that process these files directly should replace this step. As one possibility, a convolutional neural network is suitable for explicit image classification. Alternatively, a neural network can also "preprocess" image data, learning how to rate the different cytology features and either perform a final classification, based on this knowledge, or hand these results over to another algorithm, such as the XGBoost learner developed here, and obtain the final binary classification of the problem.

REFERENCES:

[1] de Bruijne, Marleen (2016). Machine learning approaches in medical image analysis: From detection to diagnosis. *Medical Image Analysis*, 33. doi: 10.1016/j.media.2016.06.032

[2] Sidey-Gibbons, J., Sidey-Gibbons, C. (2019). Machine learning in medicine: a practical introduction. *BMC Med Res Methodol*, 19(64). doi: 10.1186/s12874-019-0681-4

[3] Konstantina Kourou, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, Dimitrios I. Fotiadis (2015). Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal*, Volume 13, 2015, pp. 8-17. doi: 10.1016/j.csbj.2014.11.005.

[4] William H. Wolberg and O.L. Mangasarian (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Sciences, U.S.A*., 87, 9193-9196.

[5] O. L. Mangasarian and W. H. Wolberg (1990). Cancer diagnosis via linear programming. *SIAM News*, 23 (5), 1-18.

[6] O. L. Mangasarian, R. Setiono, and W.H. Wolberg: "Pattern recognition via linear programming: Theory and application to medical diagnosis", in *Large-scale numerical optimization*, Thomas F. Coleman and Yuying Li, editors, SIAM Publications, Philadelphia 1990, pp 22-30.

[7] K. P. Bennett & O. L. Mangasarian (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software,* 1, 23-34.

[8] 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.

[9] Gavin Brown. Diversity in Neural Network Ensembles. The University of Birmingham. 2004.

[11] Hussein A. Abbass. An evolutionary artificial neural networks approach for breast cancer diagnosis. Artificial Intelligence in Medicine, 25. 2002.

[12] Agarap, Abien Fred M. (2019). On Breast Cancer Detection: An Application of Machine Learning Algorithms on the Wisconsin Diagnostic Dataset. arXiv:1711.07831v4 [cs.LG] 7 Feb 2019

[13]Montazeri M, Montazeri M, Montazeri M, Beigzadeh A. (2016). Machine learning models in breast cancer survival prediction. *Technol Health Care,* 24(1), 31-42. doi:10.3233/THC-151071

[14] T. Chen , C. Guestrin (2016) . XGBoost: A Scalable Tree Boosting System .*arXiv*. 1603.02754v3

[16] David Forsyth (2019). Applied Machine Learning. *Springer Int. Publishing*,Ed. 1. Doi: 10.1007/978-3-030-18114-7. Isbn 978-3-030-18114-7 .