Machine Learning Algorithms for Binary Classification of Liver Disease

Anton Sokoliuk, Galyna Kondratenko,
Ievgen Sidenko, Yuriy Kondratenko,
Anatoly Khomchenko
Intelligent Information Systems Department
Petro Mohyla Black Sea National University
Mykolaiv, Ukraine
antonsokoliuk@gmail.com,
halyna.kondratenko@chmnu.edu.ua,
ievgen.sidenko@chmnu.edu.ua,
yuriy.kondratenko@chmnu.edu.ua, khan@chmnu.edu.ua

Igor Atamanyuk
Department of Higher and Applied Mathematics
Mykolayiv National Agrarian University
Mykolaiv, Ukraine
atamanyuk@mnau.edu.ua

***Abstract*—The number of patients with liver diseases has been continuously increasing because of excessive consumption of alcohol, inhale of harmful gases, intake of contaminated food, pickles, and drugs. Early diagnosis of liver problems will increase patients' survival rates. Liver disease can be diagnosed by analyzing the levels of enzymes in the blood. Creating automatic classification tools may reduce the burden on doctors. To achieve this numerous classification algorithm (Decision Tree, Random Forest, SVM, Neural Net, Naive Bayes, and others) from different machine learning libraries (Scikit-learn, ML.Net, Keras) are tested against existing liver patients' dataset, considering appropriate for each algorithm preliminary data processing. These algorithms evaluated based on three criteria: accuracy, sensitivity, specificity.**

Keywords—medicine; liver disease; machine learning; binary classification; data mining

I. INTRODUCTION

The liver is one of the most important organs of internal secretion. It is the largest gland involved in the digestion of food. Besides, the liver serves as a "filter", pumping through itself and purifying 1.5 liters of blood every minute.

Establishing an automated system for diagnosing liver disease can prevent sick or potentially ill people from having problems, and early visits to a doctor and medical attention can prevent the disease from spreading to a more severe stage. Such systems, if given the right accuracy, can help specialists with accurate diagnosis in the presence of leaky data [1].

Automated liver disease diagnostics systems can be implemented into mobile or web-based applications, in which users will enter their data and analysis results for further processing, or be part of a larger automated patient data collection system that, when detected deviations in biochemical parameters, automatically prescribes referrals appropriate doctor [2-3].

To create such systems, it is necessary to process large amounts of patient medical information in advance and to determine their correlation with specific diseases. The most effective way to do this now is through machine learning.

II. RELATED WORKS AND PROBLEM STATEMENT

Bendi Venkata Ramana et al. [4] compared popular classification algorithms for evaluating their classification performance in terms of accuracy, precision, sensitivity, and specificity in classifying liver patients dataset. These parameters are better for the research liver dataset compared to other liver datasets with all the selected algorithms. This can be attributed to more number of useful attributes like total bilirubin, direct bilirubin, and indirect bilirubin.

Bendi Venkata Ramana et al. [5] proposed Bayesian classification for the diagnosis of liver diseases. The Bayesian classification is combined with Bagging and Boosting for better accuracy. This accuracy can be further improved with a huge amount of data.

Manju Bhardwaj et al. [6] proposed and empirically evaluate a novel method for generating members of the ensemble based on 'learning from mistakes' paradigm. Support Vector Machines (SVM) is used as the base learner, and a series of dependent classifiers are obtained using model based instance selection method. Simple majority voting has been used to combine learners. It is found that the ensemble created shows better accuracy as compared to the ensembles created using AdaBoost, MAdaBoost, Bagging, and Arc-x4SVM.

Sumedh Sontakke, Jay Lohokare, and Reshul Dani in [7] investigates the effectiveness of SVMs and neural networks for the detection of patients with liver diseases based on biochemical parameters and the prospects of molecular biology and DNA analysis.

ML.NET is a free software machine learning library for the C# and F# programming languages. It also supports Python models when used together with NimbusML. The preview release of ML.NET included transforms for feature engineering like n-gram creation, and learners to handle binary classification, multi-class classification, and regression tasks. Additional ML tasks like anomaly detection and recommendation systems have since been added, and other approaches like deep learning will be included in future versions [1, 8].

Scikit-learn is a free software machine learning library for the Python programming language. It features various

2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T) | 978-1-7281-9177-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/PICST51311.2020.9468051

978-1-7281-9177-5/20/$31.00 ©2020 IEEE

classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy [9-12].

Keras [13-15] is an open-source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer are François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model [16-18].

The work aims to automate the process of diagnosing liver disease based on machine learning algorithms and tools to help doctors identify, classify, and warn potentially ill people about existing problems.

III. Analyzing and Using Different Machine Learning Classification Algorithms

In this paper Indian Liver Patient Dataset (ILPD), which is openly available at the University of California at Irvine (UCI) machine learning repository, was selected to analyze machine learning methods. The dataset contains data from 583 Indian patients from the Andhra Pradesh area on the 11 features given in Table 1.

TABLE I. Dataset features

Feature	Type
Age	Categorical (Male, Female)
Gender	Number
Total Bilirubin	Number
Direct Bilirubin	Number
Alkaline Phosphatase	Number
Alanine Aminotransferase	Number
Aspartate Aminotransferase	Number
Total Proteins	Number
Albumin	Number
Albumin and Globulin Ratio	Number
Sickness	Categorical (1 - sick, 2 - healthy)

It should be noted that the attribute "Sickness" is not an accurate diagnosis of the patient, but the expert opinion based on the analysis of these patients.

Confusion matrix in the field of machine learning and specifically the problem of statistical classification is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa). A confusion matrix for binary classification (Fig. 1) shows the four different outcomes: true positive, false positive, true negative, and false negative [2, 3].

To demonstrate the performance of the trained classifiers, it was decided to use indicators as accuracy, sensitivity, and specificity.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Fig. 1. The general look of the confusion matrix.

Accuracy is the proportion of true results among the total number of cases examined [2]:

$$Accuracy = \frac{True\ Positive\ +\ True\ Negative}{Positive\ +\ Negative}. \quad (1)$$

Sensitivity (also called the true positive rate, the recall, or probability of detection in some fields) measures the proportion of actual positives that are correctly identified as such [3]:

$$Sensitivity = \frac{True\ Positive}{True\ Positive\ +\ False\ Negative}. \quad (2)$$

Specificity (also called the true negative rate) measures the proportion of actual negatives that are correctly identified as such [2, 4]:

$$Specificity = \frac{True\ Negative}{False\ Positive\ +\ True\ Negative}. \quad (3)$$

Balanced accuracy was used to determine the most appropriate parameters for the classifiers, as the dataset is significantly unbalanced and most performance metrics show significant bias for a more represented class.

Balanced accuracy (also Informedness or Youden's J statistic) measures the proportion of correctly identified records of each class relative to class size. In case of binary classification can be described with the formula [1, 5]:

$$I = Sensitivity + Specificity - 1. \quad (4)$$

Many machine learning algorithms work better when features are on a relatively similar scale and close to normally distributed. Two ways of data preprocessing were used in this study [2, 19].

The scale is generally meant to change the range of values. The shape of the distribution doesn't change. Think about how a scale model of a building has the same proportions as the original, just smaller. That is why we say it is drawn to scale. The range is often set at 0 to 1 [2].

Standardize is generally means changing the values so that the distribution standard deviation from the mean equals one. It outputs something very close to a normal distribution. Scaling is often implied [19].

For Scikit-learn and Keras scaling with MinMaxScaler, MaxAbsScaler and standardizing with StandartScaler were tested.

For each ML.NET algorithm data was preprocessed using MinMax scaling. Class "healthy" selected as positive. The training used 5-fold cross-validation. Accuracy for different ML.NET classifiers is presented in Table 2.

TABLE II. ACCURACY FOR DIFFERENT ML.NET CLASSIFIERS

Algorithm	Accuracy	Sensitivity	Specificity
FastTree	0.69	0.44	0.8
FastForest	0.71	0.25	0.89
LightGBM	0.72	0.47	0.82
GAM	0.69	0.46	0.79
LinearSVM	0.64	0.51	0.70
SGD	0.64	0.71	0.62

The FastForest algorithm gave the best accuracy in identifying sick patients but showed a large margin of error in determining healthy patients. The LightGBM (Fig. 2) algorithm gave the highest overall accuracy but not the best for identified sick patients. In general, all the classifiers provided little accuracy for practical use.

```
Accuracy: 0,72
Sensitivity: 0,47
Specificity: 0,82
79,0  74,0
88,0  342,0
```

Fig. 2. Confusion matrix for LightGBM classifier.

GridSearchCV class was used to find the optimal parameters for each Scikit-learn algorithm. Data were preprocessed the most suitable way for each algorithm. Accuracy for different Scikit-learn classifiers is presented in Table 3.

TABLE III. ACCURACY FOR DIFFERENT SCIKIT-LEARN CLASSIFIERS

Algorithm	Accuracy	Sensitivity	Specificity
LogisticRegression	0.7	0.22	0.89
Perceptron	0.675	0.56	0.72
GaussianNB	0.55	0.95	0.39
KNeighbors	0.67	0.48	0.74
MLP	0.71	0.28	0.88
SGD	0.69	0.32	0.835
PassiveAggressive	0.69	0.22	0.88
DecisionTree	0.68	0.39	0.79
LinearDiscriminantAnalysis	0.675	0.44	0.77
NearestCentroid	0.6	0.67	0.58
SVC	0.7	0.38	0.83
NuSVC	0.635	0.64	0.635
AdaBoost+ SVC	0.7	0.06	0.96
AdaBoost+ Perceptron	0.685	0.29	0.84
GradientBoosting	0.72	0.4	0.845
XGBoost	0.7	0.4	0.81

Gradient Tree Boosting (Fig. 3) ensemble algorithm gave the best accuracy among all others; among the simpler algorithms, the support vector machine model and the multilayer perceptron performed well. In general, all the classifiers provided little accuracy for practical use [20-22].

Keras (Fig. 4) performed well compared to most models, but slightly worse than Gradient Tree Boosting. Analyzing the performance results of the classifiers, a tendency was found, according to which the accuracy of the determination of sick patients is much higher than the accuracy of the determination of healthy patients. This can be explained by the significant dominance of sick patient records in the dataset [23].

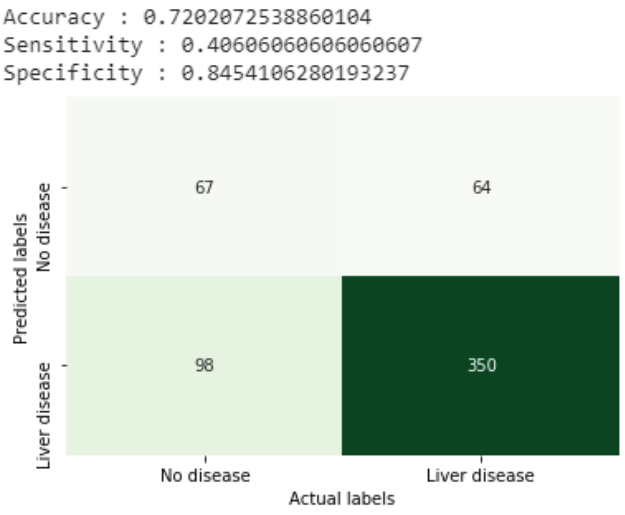

Fig. 3. Confusion matrix for Gradient Tree Boosting classifier.

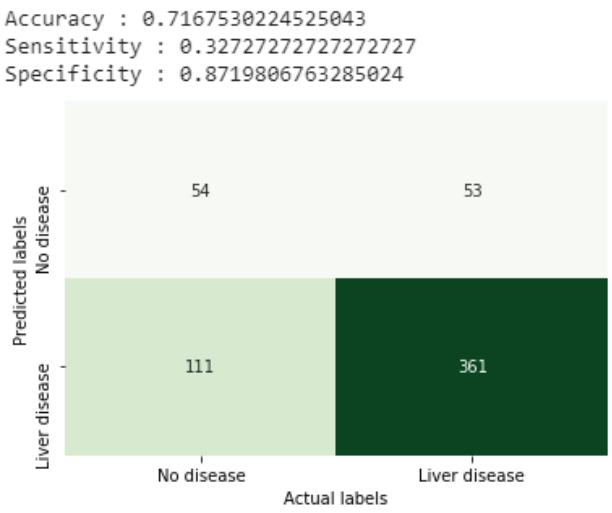

Fig. 4. Confusion matrix for the Keras sequential neural network.

A balanced set was created with the same amount of data of sick and healthy patients (165 each), on which the classifiers were trained, but the testing was performed on a full set. Data were preprocessed using Scikit-learn StandartScaler for each algorithm. Accuracy for selected Scikit-learn classifiers [23, 24] using a balanced dataset is presented in Table 4.

TABLE IV. ACCURACY FOR SELECTED SCIKIT-LEARN CLASSIFIERS USING BALANCED DATASET

Algorithm	Accuracy	Sensitivity	Specificity
KNeighbors	0.74	0.97	0.65
MLP	0.63	0.92	0.51
DecisionTree	0.66	0.8	0.6
AdaBoost+ SVC	0.58	0.91	0.45
AdaBoost+ DecisionTree	0.69	0.79	0.65
GradientBoosting	0.7	0.68	0.71
XGBoost	0.67	0.81	0.57

With the same number of records of both classes in the data set, there is a tendency of increased accuracy of the positive class "healthy". Ensemble methods [25] based on decision trees again show better results on average. The KNeighbors algorithm showed the best overall accuracy of

all tests but did poorly in identifying sick patients which is a priority. The Gradient Tree Boosting algorithm (Fig. 5) did not provide the accuracy needed for precise diagnostics but still acceptable for automatic monitoring, since it correctly identifies both sick and healthy patients with 70% accuracy, as this result can be considered the best of all tested.

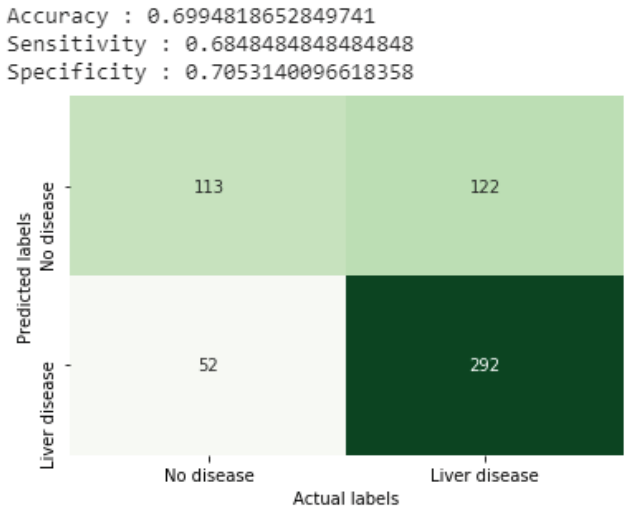

Fig. 5. Confusion matrix for Gradient Tree Boosting classifier for a balanced dataset.

Using the functionality of Colab Forms, authors create a simple application for entering values of medical analyzes of patients. To create interactivity, add a button using the Jupyter widget. The working results of the developed application are shown in Fig. 6.

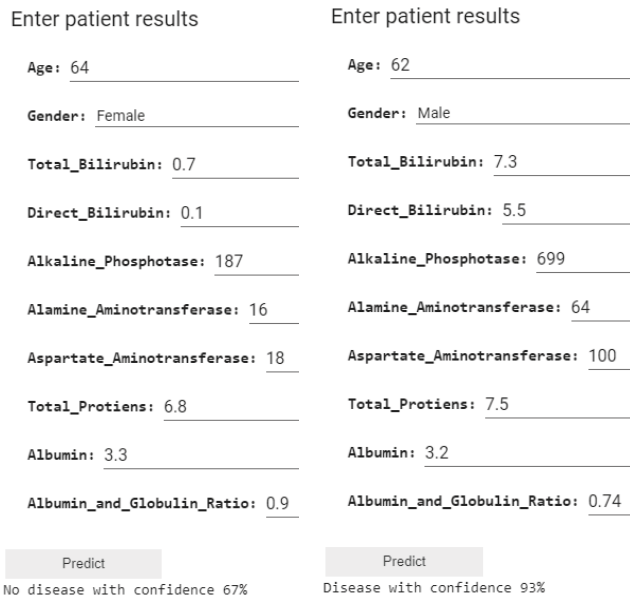

Fig. 6. Working results of the developed application for binary classification of liver disease.

Let us train the model for the Gradient Tree Boosting classifier for a balanced dataset that showed the best accuracy. The architecture of this algorithm also allows us to obtain not only a class but also its probability, which also makes sense to reflect. The developed application, based on the entered patient data, classifies the possible state of his health and the likelihood of this state using the best trained classifier.

IV. Conclusion

In this paper, multiple classification algorithms (from ML.Net, Scikit-learn, Keras) were tested in identifying patients with the liver disease using Indian Liver Patient Dataset. None of the algorithms showed accuracy needed for precise (doctor-free) diagnostic, but the Gradient Tree Boosting algorithm performed well enough for use in automatic monitoring systems. Other decision tree based ensemble methods as well as neural networks also performed relatively well. Most algorithms showed a tendency to overfitting when training on an unbalanced dataset.

Acquired accuracy can be improved using datasets with more features or containing information about patients with the same type of liver disease, which would decrease the dispersion of data of the same class.

References

[1] H. Subhani and S. Badugu, “A study of liver disease classification using data mining and machine learning algorithms,” in *Advances in Decision Sciences, Image Processing, Security and Computer Vision. Learning and Analytics in Intelligent Systems*, vol. 4, S. Satapathy, K. Raju, K. Shyamala, D. Krishna, and M. Favorskaya, Eds. Cham: Springer, 2020, pp. 630-640. DOI: 10.1007/978-3-030-24318-0_72.

[2] V. Gogi and M. Vijayalakshmi, “Analysis of liver disease and HCC inducing factors using machine learning algorithms,” in *Intelligent Data Communication Technologies and Internet of Things. ICICI 2019. Lecture Notes on Data Engineering and Communications Technologies*, vol. 38, D. Hemanth, S. Shakya, and Z. Baig, Eds. Cham: Springer, 2020, pp. 521-529. DOI: 10.1007/978-3-030-34080-3_59.

[3] R. Bhardwaj, R. Mehta, and P. Ramani, “A comparative study of classification algorithms for predicting liver disorders,” in *Intelligent Computing Techniques for Smart Energy Systems. Lecture Notes in Electrical Engineering*, vol. 607, A. Kalam, K. Niazi, A. Soni, S. Siddiqui, and A. Mundra, Eds. Singapore: Springer, pp. 753-760. DOI: 10.1007/978-981-15-0214-9_78.

[4] B. Ramana, M. Babu, and N. Venkateswarlu, “Critical study of selected classification algorithms for liver disease diagnosis,” *International Journal of Database Management Systems*, vol. 3, iss. 2, pp. 101-114, 2011.

[5] B. Ramana, M. Babu, and N. Venkateswarlu, “A critical evaluation of Bayesian classifier for liver diagnosis using Bagging and Boosting methods,” *International Journal of Engineering Science and Technology*, vol. 3, iss. 4, pp. 3422-3426, 2011.

[6] M. Bhardwaj, T. Gupta, T. Grover, and V. Bhatnagar, “An efficient classifier ensemble using SVM,” Proceedings of the IEEE International Conference on Methods and Models in Computer Science, pp. 240-246, 2009.

[7] S. Sontakke, J. Lohokare, and R. Dani, “Diagnosis of liver diseases using machine learning,” International Conference on Emerging Trends & Innovation in ICT (ICEI), pp. 129-133, 2017.

[8] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, “Multi-level net: a visual saliency prediction model,” in *Computer Vision – ECCV 2016 Workshops. ECCV 2016. Lecture Notes in Computer Science*, vol. 9914, G. Hua, and H. Jégou, Eds. Cham: Springer, 2016, pp. 302-315. DOI: 10.1007/978-3-319-48881-3_21.

[9] D. Paper, *Hands-on Scikit-learn for machine learning applications: data science fundamentals with Python*. Berkeley, CA: Apress, 2020. DOI: 10.1007/978-1-4842-5373-1.

[10] V. Porcu, “Scikit-learn,” in *Python for Data Mining Quick Syntax Reference*. Berkeley, CA: Apress, 2018, pp. 235-253. DOI: 10.1007/978-1-4842-4113-4_11.

[11] E. Bisong, “More supervised machine learning techniques with Scikit-learn,” in *Building Machine Learning and Deep Learning*

Models on Google Cloud Platform. Berkeley, CA: Apress, 2019, pp. 287-308. DOI: 10.1007/978-1-4842-4470-8_24.

[12] F. Nelli, "Machine learning with Scikit-learn," in *Python Data Analytics*. Berkeley, CA: Apress, 2015, pp. 237-264. DOI: 10.1007/978-1-4842-0958-5_8.

[13] J. Moolayil, *Learn Keras for deep neural networks: a fast-track approach to modern deep learning with Python*. Berkeley, CA: Apress, 2019. DOI: 10.1007/978-1-4842-4240-7.

[14] N. Ketkar, "Introduction to Keras," in *Deep Learning with Python*. Berkeley, CA: Apress, 2017, pp. 97-111. DOI: 10.1007/978-1-4842-2766-4_7.

[15] A. Nandy and M. Biswas, "Reinforcement learning with Keras, TensorFlow, and ChainerRL," in *Reinforcement Learning*. Berkeley, CA: Apress, 2018, pp. 129-153. DOI: 10.1007/978-1-4842-3285-9_5.

[16] E. Bisong, "TensorFlow 2.0 and Keras," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Berkeley, CA: Apress, 2019, pp. 347-399. DOI: 10.1007/978-1-4842-4470-8_30.

[17] P. Kushneryk, Y. Kondratenko, and I. Sidenko, "Intelligent dialogue system based on deep learning technology," 15th International Conference on ICT in Education, Research, and Industrial Applications: PhD Symposium (ICTERI 2019: PhD Symposium), vol. 2403, pp. 53-62, 2019.

[18] I. Sidenko, K. Filina, G. Kondratenko, D. Chabanovskyi, and Y. Kondratenko, "Eye-tracking technology for the analysis of dynamic data," IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), pp. 479-484, 2018. DOI: 10.1109/DESSERT.2018.8409181.

[19] C. Albon, *Machine learning with Python Cookbook: practical solutions from preprocessing to deep learning*. NY: O'Reilly Media, 2018.

[20] S. El-Bakry, E. El-Dahshan, and M. El-Bakry, "Total cross section prediction of the collisions of positrons and electrons with alkali atoms using Gradient Tree Boosting," *Indian J. Phys.*, vol. 85, iss. 9, pp. 1405-1415, 2011. DOI: 10.1007/s12648-011-0162-z.

[21] A. Khairuddin, R. Alwee, and H. Haron, "A proposed Gradient Tree Boosting with different loss function in crime forecasting and analysis," in *Emerging Trends in Intelligent Computing and Informatics. IRICT 2019. Advances in Intelligent Systems and Computing*, vol. 1073, F. Saeed, F. Mohammed, and N. Gazem, Eds. Cham: Springer, 2020, pp. 189-198. DOI: 10.1007/978-3-030-33582-3_18.

[22] Z. Gomolka, E. Dudek-Dyduch, and Y.P. Kondratenko, "From homogeneous network to neural nets with fractional derivative mechanism," in International Conference on Artificial Intelligence and Soft Computing (ICAISC-2017), Part I, L. Rutkowski et al., Eds., pp. 52-63, 2017.

[23] D. Paper, "Scikit-learn classifier tuning from complex training sets," in *Hands-on Scikit-Learn for Machine Learning Applications*. Berkeley, CA: Apress, 2020, pp. 165-188. DOI: 10.1007/978-1-4842-5373-1_6.

[24] V. Shebanin, I. Atamanyuk, Y. Kondratenko, and Y. Volosyuk, "Canonical mathematical model and information technology for cardio-vascular diseases diagnostics," 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings Open Access, pp. 438-440, 2017. DOI: 10.1109/CADSM.2017.7916170.

[25] E. Bisong, "Ensemble methods," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Berkeley, CA: Apress, 2019, pp. 269-286. DOI: 10.1007/978-1-4842-4470-8_23.