2023 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT) | 979-8-3503-2405-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/JEEIT58638.2023.10185873

Diagnosis of Polycystic Ovary Syndrome Using Random Forest with Bagging Technique

Amjed Al-Mousa
Computer Eng. Dept.
Princess Sumaya University for Technology
Amman, Jordan
a.almousa@psut.edu.jo

Badr Mansour
Computer Eng. Dept.
Princess Sumaya University for Technology
Amman, Jordan
bad20180009@std.psut.edu.jo

Hamsa Al-Dabbagh
Computer Eng. Dept.
Princess Sumaya University for Technology
Amman, Jordan
ham20180206@std.psut.edu.jo

Mohammad Radi
Computer Eng. Dept.
Princess Sumaya University for Technology
Amman, Jordan
moh20180187@std.psut.edu.jo

Abstract— **The goal of this research is to aid doctors in the diagnosis of PCOS in female patients. Diagnosing the condition in question depends on many factors making it complex to diagnose. The model developed would help confirm a doctor's diagnosis to further its reliability. The model tested several classifiers, including Extreme Gradient Boosting (XGBoost), Linear Discriminant Analysis (LDA), and Adaptive Boosting (Ada-Boost). The highest accuracy was 94.4% using the Random Forest classifier with the Bagging method. This accuracy surpasses any previously achieved results using the same dataset, which were 91% and 92%. The results achieved were using a 10-Fold cross-validation.**

Keywords— Machine Learning, Classification, Polycystic Ovary Syndrome, PCOS Diagnosis, Random Forest, Bagging, Kerala India Dataset

I. Introduction

Polycystic Ovarian Syndrome (PCOS) is a hormonal imbalance caused by the ovaries creating a massive amount of male hormones (androgens), which will cause the female hormones to become imbalanced. As a result, people with PCOS often have erratic menstrual cycles, missed periods, and unpredictable ovulation. Small cysts may develop on the ovaries (fluid-filled sacks) due to a lack of ovulation (anovulation) [1].

PCOS is one of the most common causes of female infertility. It can also increase the risk of other health conditions, such as diabetes, high blood pressure, depression, and anxiety [2]. According to World Health Organization (WHO) data statistics, approximately 116 million women (3.4% of the population) are affected by PCOS globally [3].

Diagnosing PCOS has several challenges due to its vague symptoms and lack of research. The condition can affect people differently based on age, race, and other factors, which makes it difficult for physicians to recognize. Additionally, a lack of coordination between doctors can lead to a lack of comprehensive assessment, making it more difficult to diagnose PCOS. Furthermore, PCOS is often underdiagnosed, with up to 70% of women with the condition remaining undiagnosed, particularly adolescents [4].

The dataset was collected from 10 hospitals across Kerala, India, to train and test the machine learning model. Moreover, the dataset contains 43 features based on the required medical tests and the symptoms for diagnosis; one feature represents the result. The dataset contains 541 instances; 177 have been diagnosed with PCOS, and the rest were not [5].

The rest of the paper is structured as follows: Section II will explore literature related to our research, discussing the disease further and how a machine learning model can help diagnose it. Section III is the experimental setup, such as data pre-processing and preparing the dataset for training and testing. Section IV will detail the classifiers used for our model before the testing and voting process. Section V examines the results, including confusion matrices and the accuracies observed for the classifiers. Finally, the last section details the conclusions made from the research.

II. Related Work

Machine learning algorithms help identify complex patterns using large amounts of data. They commonly diagnose illnesses like heart disease [6], stroke [7], and diabetes [8] using multiple classifiers. Such algorithms will produce better treatment plans for patients in medical applications by making recommendations for building useful healthcare systems [9].

In [4], a model was built to automate PCOS diagnosis using machine learning. For training purposes, a dataset consisted of 39 features ranging from metabolic imaging to hormonal and biochemical parameters for 541 patients. They used several classification algorithms to train and test, such as Logistic Regression (LR), Decision Tree (DT), Naïve Bayes (NB), Linear Support Vector Machine (LSVM), Polynomial Support Vector Machine (PSVM), Radial Basis Function Support Vector Machine (RBF SVM), K-Nearest Neighbors (KNN), AdaBoost (AD), Linear Discriminant classifier (LD), Quadratic Discriminant classifier (QD), and Random Forest (RF). In this paper, the LSVM classifier (using a 10-fold CV) was chosen, as it performed best among the others in terms of precision at 93%, recall at 80%, and overall accuracy at **91%**.

Another machine learning model was constructed in [10] to diagnose PCOS using MATLAB. In that paper, seven classifiers were used, scoring an accuracy of **92%** using the Linear Discriminant classifier [10].

Papers [4] and [10] used the same dataset for this research; therefore, those are the accuracy results to beat. The dataset was compiled by authors from India who trained their initial model with five classifying algorithms: K-nearest Neighbor, Naive Bayes Classifier, Support Vector Machine (SVM), Decision Tree Classifier, and Logistic Regression. The best accuracy (81%) was scored using the Decision Tree classifier. After training and testing the model, an application was developed which aims to help with the early classification of PCOS by simply asking questions to patients and making the classification based on the answers provided. However, it still suggests having a doctor's consultation before getting treatment [11].

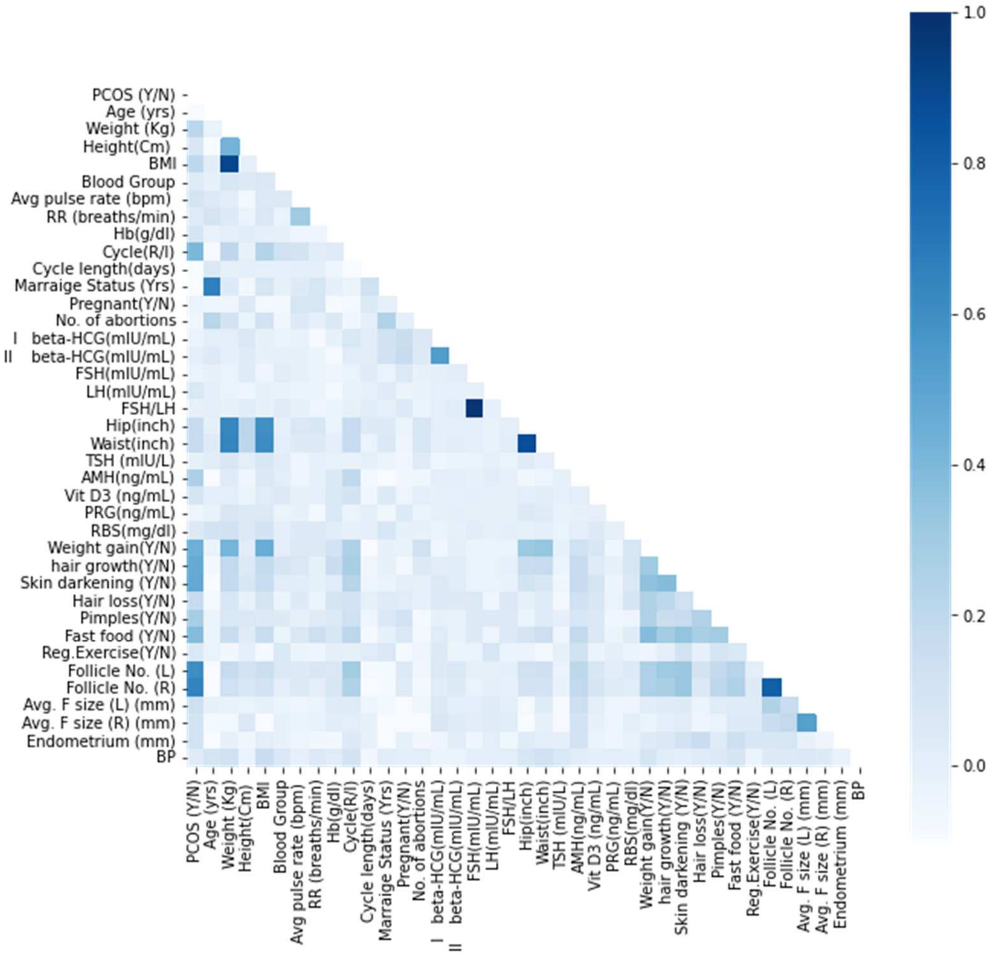

Figure 1: Heatmap showing the correlation of features

III. Experimental Setup

The research created a classification model to diagnose PCOS in female patients. The dataset was collected from Indian patients in Kerala. The data was split for training and testing; part was used to train the model and part to test it (diagnosis classification) [8].

A. The Dataset's Features

The dataset used initially had 44 features, 1 of which is the diagnosis (yes/no value). The features mainly result from tests taken on various patients by hospitals in the previously mentioned area. Some relate to the patient's activity and overall wellness; others are specific measurements from such tests. Due to a large number of features, there needed to be further examination done to determine what's relevant to training the classification model.

First, the most basic data-relevance metric was examined, the correlation, which would give a decent representation of how much certain features would impact the overall outcome of the diagnosis -before training the model. Also, it's important to note that scaling, feature engineering, and dropping are needed before the training process to make the training data set more reasonable.

After testing for correlations, a heat map can be drawn to illustrate how features correlate, as shown in Figure 1. It includes a list of all the features kept, which are now 38, and the diagnosis.

The figure shows that the highest correlations achieved were approximately 0.6. These metrics belonged to the "Follicle No. (R)" and "Follicle No. (L)" features, which are the number of follicles on the left and right side. This was unsurprising as these measures are what most doctors typically use to determine the likelihood of a patient having PCOS.

Histograms were plotted for each feature to individually examine the features with the highest correlation values, as shown in Figure 2. The histograms for Follicle No. (L) and (R) corroborate the heatmap and correlation measurements, showing the highest correlation out of all the features. Some histograms visually represent potential outliers in the data, which helps with the next stage of the experimental setup, preparing the data for training.

B. Data Preparation & Feature Engineering

Pre-processing data is crucial to getting logical and accurate results when testing. Therefore, the data should undergo several stages before training the model.

First, any null values in the data set should be filled by data cleaning. The opted-for method here uses the mean values of each feature to fill those nulls with the average measurement of that feature. Two features had null value entries, those being "Marriage Status (Yrs.)" and "Fast Food (Y/N)". The mean values of those features were found and placed inside two new variables, then using the "fillna()" method, the nulls were filled with these two variables.

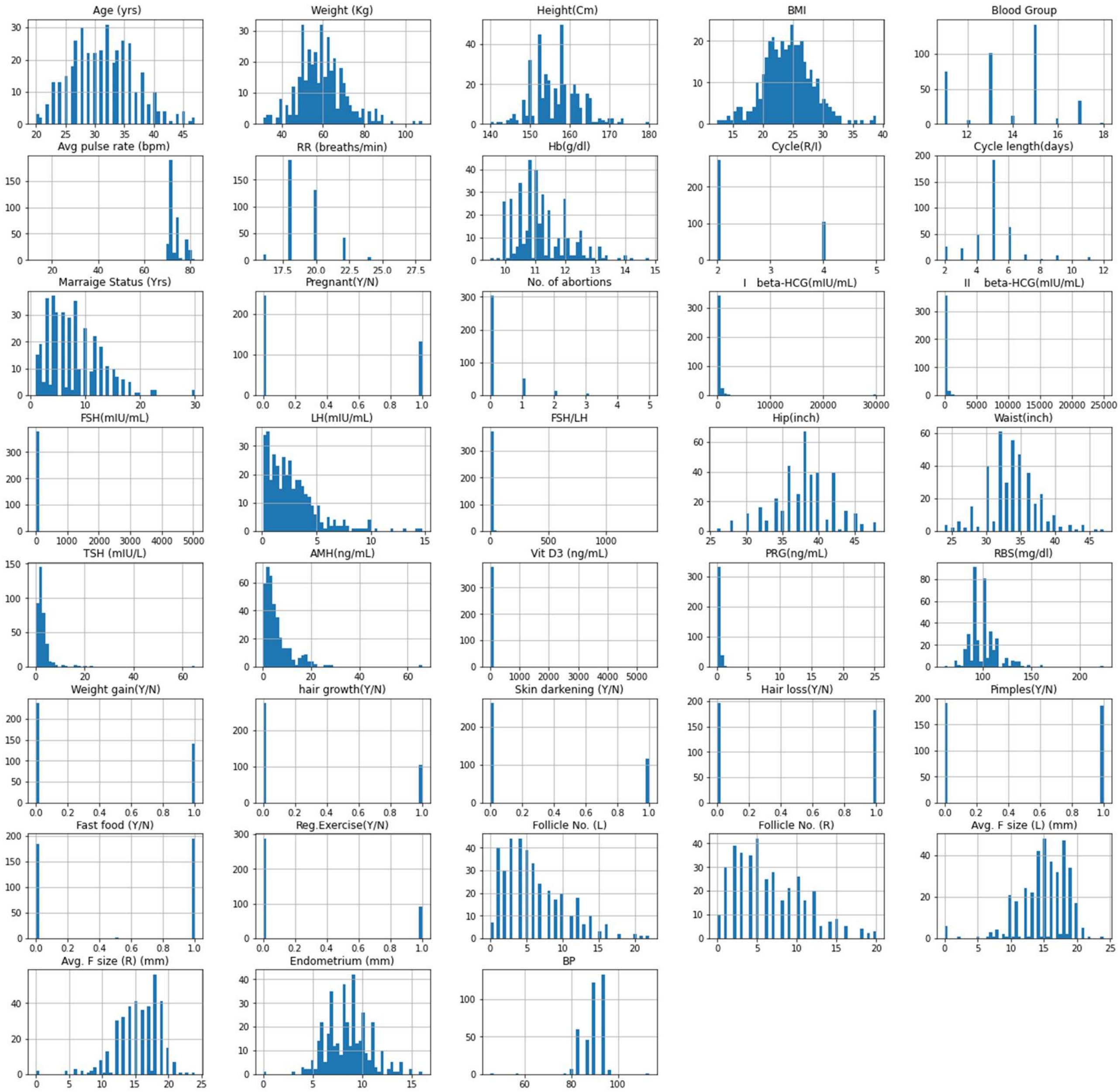

Figure 2: Histograms of each feature

Next, the features with the lowest correlation values were dropped entirely from the dataset. These include features that are either utterly irrelevant to training (the patient's file number and the field entry number) or features that have other features representing them (either already or will be added with feature engineering later). These included the Systolic and Diastolic blood pressure readings combined into one Blood Pressure "BP" feature. The remaining dropped features had the worst correlation score among all the features. These were the waist-to-hip ratio and PRL hormone levels in the patient's blood.

As mentioned prior, feature engineering was helpful for the blood pressure features. These were previously "BP_Systolic" and "BP_Diastolic"; combined into one feature to become "BP". This was done using the Mean Arterial Pressure (MAP) formula. Equation (1), the formula for MAP, was used to create this feature combination.

$$BP = Dp + \frac{Sp - Dp}{3} \tag{1}$$

Dp represents the value of the "BP_Diastolic" feature, *Sp* represents that of the "BP_Systolic" feature, and BP is the new feature created as a combination of the two.

Initially, the dataset had varying scales, which made observing correlation from the histograms difficult. Standardization (the "StandardScaler()" method) was used to solve this inconsistency. This way, the range used by the features wasn't as heavily impacted by outliers.

C. Testing Parameters

The last step before splitting the data into a train and test set was to set specific parameters instrumental to the success and consistency of the training process.

- **Stratify:** the dataset itself was initially imbalanced since only around 32% of the entries had the value 1 for the “PCOS (Y/N)” feature, the label the model will classify when testing. The solution to this was to use stratified sampling when training the data (by setting the “stratify” parameter to “Y”).
- **CV:** this hyperparameter is used to apply cross-validation to select classifiers. It was set to “kf” which is the variable used for Stratified K-Fold; that way, the classifier it was used on will use K-Fold Cross Validation to optimize its training.
- **Random state:** this model hyperparameter is set in multiple places to ensure the position of test values stays consistent [12] when splitting the data into the training and test set. It is also set when using specific classifiers such as Random Forest (discussed in the next section). The default value of “42” was used for it in all cases.
- **N estimators:** this parameter is used for ensemble learning methods to set the number of estimators to use in the ensemble [13], which was needed for the Decision Tree classifier since ensemble learning was used.

Finally, the training and testing data are split into two separate data sets with a ratio of 70:30 (i.e., 70% training and 30% testing).

IV. Machine Learning Algorithms

After preparing the data and setting the necessary parameters, the dataset was split into a training set and a testing set (with stratified sampling), which will be used for training and testing. The classifier models that were tested are as follows.

A. Random Forest Classifier with Bagging Method

The first classifier used was the Random Forest classifier which fuses multiple decision trees to produce a finer predictive performance than a single decision tree. The main benefit of ensemble models is that combining multiple weak models will have a much better outcome. Using the Bagging method combines predictions on different subsets of the dataset, which in the case of imbalanced datasets helps balance class distribution. When paired with stratified sampling, the dataset’s imbalance does not hinder the model's performance. Each subset is used for training their independent decision tree, and the average of their predictions is taken together [14].

B. Extreme Gradient Boosting (XGBoost) Classifier

Gradient Boosting is yet another technique used in ensemble learning. As ensemble learning is based on decision tree models, each constructed tree is error corrected by the tree before that model. Models are trained using any arbitrary differentiable loss function and gradient descent optimization algorithm. As the negative gradient is minimized each time a model is trained hence the name “Gradient Boosting” [15].

C. Random Forest Classifier

This classifier is one of the least complex algorithms. It uses several independent decision trees to work efficiently as one. Each model classifies the label to the class an entity is a member of. The class that receives the most votes is the one that is predicted [8].

D. Linear Discriminant Analysis (LDA) Classifier

The Linear Discriminant Analysis utilizes dimensionality reduction by projecting higher-dimension features onto a lower-dimension. The predictors were combined linearly to minimize the within-group variance, and the between-group variance was maximized. This way, the classifier tries to distinguish (or discriminate) the training dataset's samples by their class value as well as possible [16].

E. Adaptive Boosting (Ada-Boost) Classifier

Ada-boost is another method of ensemble learning that is iterative. It merges multiple classifiers to increase accuracy. It combines several weak classifiers to create a robust classifier that has a higher degree of accuracy. The fundamental idea is to train the data sample and adjust the classifier weights in each iteration to provide accurate predictions of uncommon observations [17].

F. Voting Classifier

A voting classifier uses many classifiers to produce predictions; the output changes based on the parameter (soft or hard voting) contained in the model. Random Forest classifier utilizing the bagging technique, Extreme Gradient Boosting classifier, Random Forest classifier, Linear Discriminant classifier, and Adaptive Boosting classifier are the five models selected as the classifiers for both soft and hard voting.

V. Results

First, the Random Forest classifier with Bagging is considered. This trained model had the highest accuracy of all the non-voting classifiers on the test set (**94.4%**) when using stratified K-Fold (10 folds). It achieved a precision of 100%, recall of 83%, and F1 score of 91%. Figure 3 shows the confusion matrix acquired from this model.

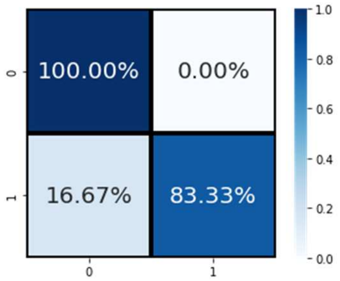

Figure 3: Random Forest classifier with Bagging confusion matrix

The second model was trained using the Extreme Gradient Boosting classifier, which scored an accuracy of **89.57%**. It achieved a precision of 86%, recall of 81%, and F1 score of 83%. Figure 4 shows the confusion matrix acquired from this model.

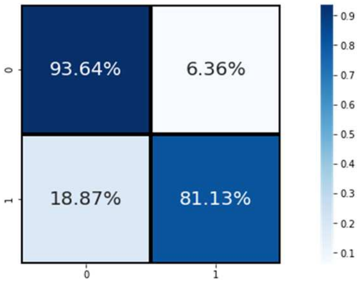

Figure 4: XGBoost classifier confusion matrix

The third model used was the Random Forest classifier which reached an accuracy of 88.96%. It achieved a precision of 91%, a recall of 74%, and an F1-score of 81%. Figure 5 shows the confusion matrix acquired from this model.

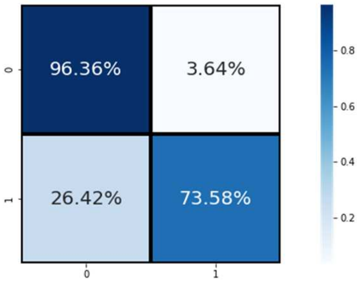

Figure 5: Random Forest classifier confusion matrix

Next, the model was trained using Linear Discriminant Analysis. The accuracy when running the model on the test data was **87.11%**. It achieved a precision of 82%, recall of 77%, and F1 score of 80%. Figure 6 shows the confusion matrix acquired from this model.

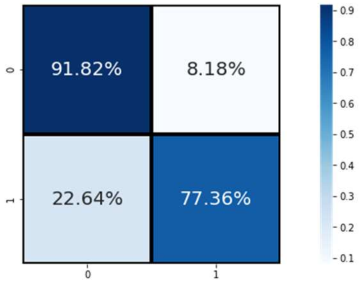

Figure 6: LDA classifier confusion matrix

The next model was trained using the Adaptive Boosting classifier, which reached an accuracy of **86.5%**. It achieved a precision of 82%, recall of 75%, and F1 score of 78%. Figure 7 shows the confusion matrix acquired from this model.

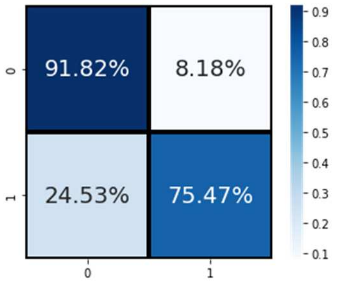

Figure 7: Ada-Boost classifier confusion matrix

As for the Soft Voting classifier; it achieved an accuracy of **88.35%,** a precision of 84%, a recall of 79%, and an F1 score of 82%. Figure 8 shows the confusion matrix acquired from this model.

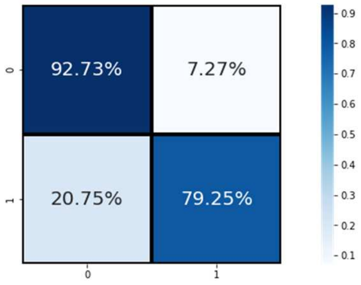

Figure 8: Soft Voting confusion matrix

Lastly, the model was trained using a Hard-Voting classifier; all models were included in this classifier to make the decision. The accuracy of the hard-voting model was **89.57%**. It achieved a precision of 89%, recall of 77%, and F1 score of 83%. Figure 9 shows the confusion matrix acquired from this model.

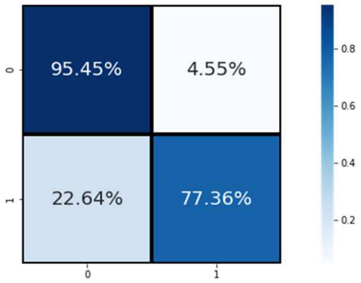

Figure 9: Hard Voting confusion matrix

Table 1 shows a summary of the classifiers' performance metrics, which shows RF with Bagging achieving the highest accuracy of 94.4% and the highest precision, recall, and F1 score of all the classifiers. The results presented are for testing the dataset, which indicates that there was no over-fitting.

Table 1: Performance of classifiers

Classifier Name	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**
RF with Bagging	**94.4%**	**100%**	**83%**	**91%**
XGBoost	89.6%	86%	81%	83%
RF	89.0%	91%	74%	81%
LDA	87.1%	82%	77%	80%
Ada-Boost	86.5%	82%	75%	78%
Soft Voting Classifier	88.4%	84%	79%	82%
Hard Voting Classifier	89.6%	89%	77%	83%

VI. Conclusion

Finally, the goal of the research was to develop a machine-learning classification model that would assist in diagnosing PCOS by confirming a doctor's suspicion of a patient's condition.

The research explored related work to help develop the best possible implementation of this model. After training, the model was tested using many methods to determine the most accurate classifier. The result was that the Random Forest classifier using Bagging achieved the highest accuracy of 94.4% when using stratified k-fold cross-validation. This result beats the previous accuracy scores that used the same dataset, as shown in Table 2.

Table 2: Comparison with other research papers

Best Classifier	**Highest Accuracy**	**Precision**	**Recall**	**F1 Score**
RF with Bagging (This work)	**94.4%**	**100%**	83%	91%
LSVM - [4]	91%	93%	80%	86%
LDA - [10]	92%	97%	**92%**	**94%**

It's important to note that the precision achieved in this paper was 100% which is ideal for this medical use case since no patient will be wrongfully diagnosed if they don't have PCOS. This precision is higher than that of the other papers, shown in Figure 10. There are a few other key factors to note when comparing further between the papers:

- [4] used z-score for normalization, whereas this paper's data was normalized using min-max scaling
- [10] had a ratio of 80:20 (i.e., 80% training and 20% testing) when splitting the data
- Neither of the other papers mentioned data stratification

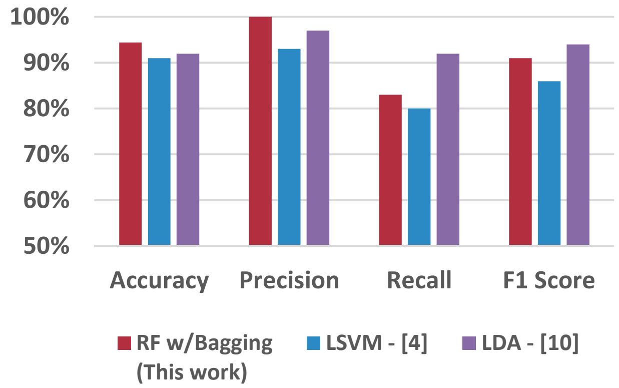

Figure 10: Comparison with other research

References

[1] "Cleveland Clinic," [Online]. Available: https://my.clevelandclinic.org/health/diseases/8316-polycystic-ovary-syndrome-pcos.

[2] "Office on Women's Health," [Online]. Available: https://www.womenshealth.gov/a-z-topics/polycystic-ovary-syndrome#:~:text=Who%20gets%20PCOS%3F,at%20any%20age%20after%20puberty..

[3] "CureUS," [Online]. Available: https://www.cureus.com/articles/108330-polycystic-ovarian-syndrome-prevalence-predisposing-factors-and-awareness-among-adolescent-and-young-girls-of-south-india.

[4] Y. A. A. Adla, D. G. Raydan, M. -Z. J. Charaf, R. A. Saad, J. Nasreddine and M. O. Diab, "Automated Detection of Polycystic Ovary Syndrome Using Machine Learning Techniques," in *2021 Sixth International Conference on Advances in Biomedical Engineering (ICABME)*, 2021, pp. 208-212.

[5] "Kaggle Dataset," [Online]. Available: https://www.kaggle.com/code/karnikakapoor/pcos-diagnosis.

[6] R. Atallah and A. Al-Mousa, "Heart Disease Detection Using Machine Learning Majority Voting Ensemble Method," in *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)*, 2019, pp. 1-6.

[7] H. Al-Zubaidi, M. Dweik and A. Al-Mousa, "Stroke Prediction Using Machine Learning Classification Methods," in *2022 International Arab Conference on Information Technology (ACIT)*, Abu Dhabi, United Arab Emirates, 2022, pp. 1-8.

[8] N. Abdulhadi and A. Al-Mousa, "Diabetes Detection Using Machine Learning Classification Methods," in *2021 International Conference on Information Technology (ICIT), 2021*, 2021, pp. 350-354.

[9] K. Shailaja, B. Seetharamulu and M. A. Jabbar, "Machine Learning in Healthcare: A Review," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2018*, 2018, pp. 910-914.

[10] D. Hdaib, N. Almajali, H. Alquran, W. A. Mustafa, W. Al-Azzawi and A. Alkhayyat, "Detection of Polycystic Ovary Syndrome (PCOS) Using Machine Learning Algorithms," in *2022 5th International Conference on Engineering Technology and its Applications (IICETA), 2022*, 2022, pp. 532-536.

[11] P. Chauhan, P. Patil, N. Rane, P. Raundale and H. Kanakia, "Comparative Analysis of Machine Learning Algorithms for Prediction of PCOS," in *2021 International Conference on Communication information and Computing Technology (ICCICT), 2021*, 2021, pp. 1-7.

[12] "Medium TowardsDS," [Online]. Available: https://towardsdatascience.com/why-do-we-set-a-random-state-in-machine-learning-models-bb2dc68d8431.

[13] "Medium Vignesh," [Online]. Available: https://vigneshmadanan.medium.com/ensemble-learning-explained-part-2-498ab87788b5.

[14] "DT Ensembles," [Online]. Available: https://towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9#:~:text=Ensemble%20methods%2C%20which%20combines%20several,to%20form%20a%20strong%20learner..

[15] "XGBoost Algo," [Online]. Available: https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/.

[16] "LDA for DimRed," [Online]. Available: https://machinelearningmastery.com/linear-discriminant-analysis-for-dimensionality-reduction-in-python/.

[17] "AdaBoost classifier," [Online]. Available: https://www.datacamp.com/tutorial/adaboost-classifier-python#adaboost-classifier.