Comparative Analysis of Machine Learning and Deep Learning Models for Polycystic Ovary Syndrome (PCOS) Detection

Course NO. **: EEE 4709**
Course Name **: Artificial Intelligence and Machine Learning**

Instructor Name **: Md. Arefin Rabbi Emon**

Group NO. **: A1**

Group Members:

STUDENT ID	NAME
200021102	Khandaker Adeba Tabassum
200021106	Sadat Al Rashad
200021132	MD. Nazmul Aman

TABLE OF CONTENTS

ABSTRACT

Polycystic Ovary Syndrome (PCOS) stands as one of the most harmful endocrine disorders among women in their reproductive years with complications ranging from infertility to metabolic irregularities and cardiovascular risks. Timely and accurate diagnosis of PCOS is very important for effective intervention, yet conventional diagnostic approaches often stumble due to their dependence on subjective assessments and inconsistent accuracy. This study delves into the potential of different machine learning (ML) and deep learning (DL) methodologies to refine PCOS detection, by using a dataset consisting of 541 patients and their 41 clinical and physical attributes. A bunch of ML models was defined and evaluated, including Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), and K-Nearest Neighbors (KNN), alongside DL frameworks such as Feedforward Neural Networks (FNN), Convolutional Neural Networks (CNN), Long Short-Term Memory - Convolutional Neural Networks (LSTM-CNN), Gated Recurrent Units (GRU), and Multilayer Perceptrons (MLP). The dataset was preprocessed employing median imputation to handle the missing values, MinMax normalization for scaling, and Synthetic Minority Oversampling Technique (SMOTE) to rectify class imbalances. Feature selection was implemented using Recursive Feature Elimination (RFE) and SelectKBest, while hyperparameter optimization was done through GridSearchCV and Keras Tuner. The results show that RF outperformed it's ML counterparts, attaining an accuracy of 97.26% via split validation. In the DL domain, FNN soared to an accuracy of 98.48%, achieving the total high score. However, the study also exposed the limitations of certain models, such as DT and KNN, which underperformed despite different optimizations. Moreover, DL architectures like LSTM-CNN and GRU exhibited erratic validation accuracy, hinting at inherent challenges in achieving consistent generalization. This research illuminates the promise of ML and DL models, particularly RF and FNN, in revolutionizing PCOS diagnosis, presenting a formidable alternative to traditional diagnostic paradigms. Future endeavors will focus on broadening the dataset, refining feature engineering strategies, and incorporating ensemble techniques to amplify model efficacy and adaptability.

1. INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

Polycystic Ovary Syndrome (PCOS) is the most common endocrine disorder, primarily affecting women during their reproductive years[1]. This hormonal condition not only disrupts ovarian function but also leads to a range of health issues, including metabolic disorders, cardiovascular diseases[2], infertility, mental health challenges, and type 2 diabetes[3]. Additionally, PCOS is linked to impaired insulin secretion from the pancreas. While the exact cause of PCOS remains unclear, it is thought to arise from a combination of genetic factors[4], hormonal imbalances, and environmental influences. Alarmingly, PCOS affects approximately 5–18% of women worldwide, making it a significant health concern that demands attention[5].

1.2 PROBLEM STATEMENT

Despite its prevalence, diagnosing PCOS is a complex and often challenging task, especially in its early stages. Over the years, several biomarkers have been proposed for diagnosing PCOS, with the Rotterdam criteria being the most widely used[6]. These criteria focus on three key features: androgen excess, ovulatory dysfunction, and polycystic ovarian morphology (PCOM). However, relying solely on these criteria can lead to diagnostic inaccuracies. For instance, the presence of polycystic ovaries, as detected through 3D ultrasonography, does not always indicate PCOS[7]. Furthermore, early treatment can reduce the progression of PCOS. This limitation highlights the need for more reliable and comprehensive diagnostic methods.

1.3 OBJECTIVES

In today's world, Artificial Intelligence (AI) and machine learning (ML) are transforming how diseases are detected and diagnosed. This study aims to develop a robust and accurate model for diagnosing PCOS using advanced AI techniques. We explored various ML algorithms, including Random Forest Classifier, Decision Tree, Support Vector Machine (SVM), Logistic Regression and K-Nearest Neighbors (KNN) as well as deep learning algorithms such as Feedforward Neural Networks (FNN), 1D Convolutional Neural Networks (1D-CNN), Long Short-Term Memory Convolutional Neural Networks (LSTM-CNN), Gated Recurrent Units (GRU), and Multilayer Perceptrons (MLP). To prepare the data, we normalized it and applied the Synthetic Minority Over-sampling Technique (SMOTE) to address class imbalance. For feature selection, we used Recursive Feature Elimination (RFE) and SelectKBest, and optimized the models using k-fold cross-validation, hyperparameter tuning with GridSearchCV, and split validation. The final model was chosen based on its ability to achieve high accuracy and precision in predicting PCOS.

1.4 SCOPE AND LIMITATIONS

The main contributions of this work are as follows:

1. Application of SMOTE to address the issue of imbalanced datasets and improve classification performance.
2. Implementation of Recursive Feature Elimination (RFE) to identify and select the most relevant features for better model efficiency.

3. Hyperparameter tuning using GridSearchCV to optimize model performance and achieve the best possible configuration.
4. Comparison of two different validation techniques K-fold cross-validation with a fold of 10 and split validation to determine the most effective validation approach for higher accuracy and generalization.

2. LITERATURE REVIEW / RELATED WORK

2.1 EXISTING STUDIES

Recent studies have employed a variety of machine learning classifiers for PCOS detection. Thakre et al. proposed a method utilizing five different ML classifiers, namely Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Gaussian Naïve Bayes (NB), and K-Nearest Neighbor (KNN). Among these, RF demonstrated the highest accuracy of 90.9% [8]. Purnama et al. explored PCOS detection using ultrasound images and applied SVM and KNN classifiers. Their study found that SVM with an RBF kernel achieved the best accuracy at approximately 82.55%[9]. Similarly, Dutta et al. incorporated Synthetic Minority Oversampling Technique (SMOTE) with five ML techniques—LR, RF, Decision Tree (DT), SVM, and KNN—to improve PCOS classification. Their results indicated that LR outperformed other models in classifying PCOS. Deep learning models have also been applied in PCOS detection[10]. Sumathi et al. implemented a Convolutional Neural Network (CNN)-based image processing method, achieving an accuracy of 85% in detecting cysts from ultrasound images[11]. Cahyono et al. proposed an automatic feature extraction technique using CNN, attaining an F1-score of 100% and an average accuracy of 76.36% with five-fold cross-validation[12]. Bharati et al. applied number of classifiers gradient boosting, random forest, logistic regression, and hybrid random forest and logistic regression (RFLR) and demonstrated that RFLR performed best among these classifiers exhibiting accuracy of 91.01% and recall value of 90% using cross-fold validation[13]. Hdaib et al. worked with K Nearest Neighbor (KNN), Neural Network, Naïve Bayes, Support Vector Machine (SVM), Classification Tree, Logistic Regression, and Linear Discriminant models and found highest accuracy of 92.60% with 97.6% precision in Linear Discriminant model [14]. Ahmad et al. proposed a SMOTE based lightweight DL models-based approach and achieved highest accuracy of 96.59% with 96.6% ROC-AUC using CNN-based model[15].

2.2 COMPARISON WITH EXISTING WORK

In our study, we explored various machine learning (ML) and deep learning (DL) techniques to detect Polycystic Ovary Syndrome (PCOS) and evaluated their performance using different validation methods. After extensive experimentation with multiple algorithms, the highest accuracy of 97.26% was achieved using the Random Forest (RF) classifier, with an F1-score of 97.26%. This result was obtained using the split validation technique, which demonstrated superior performance compared to other methods. The RF classifier outperformed other ML models, including Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), and K-Nearest Neighbors (KNN), which achieved accuracies of 94.52%, 90.41%, 88.36%, and 88.36%, respectively.

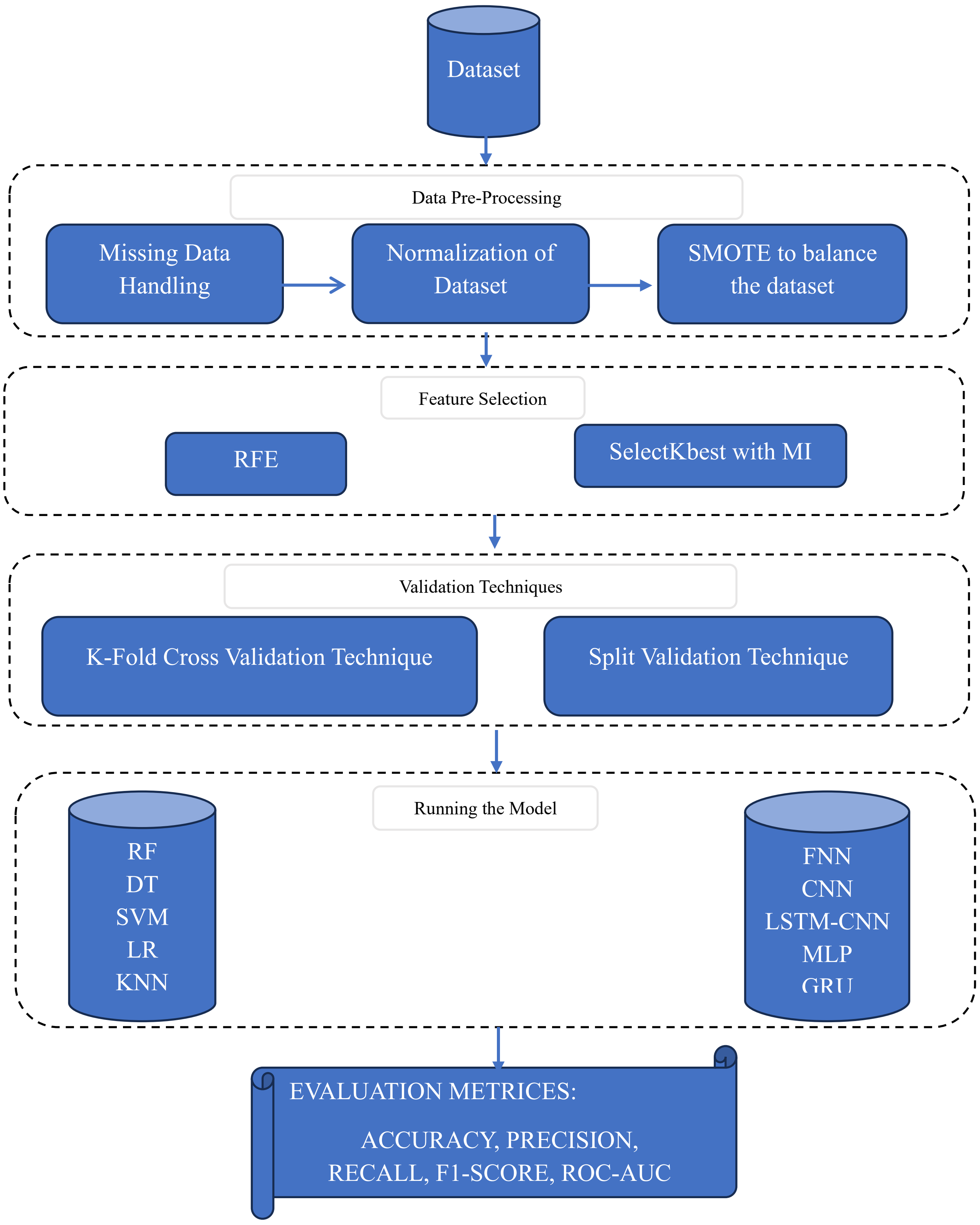
Initially, we employed K-fold cross-validation with a fold of 10 to evaluate the models. However, the accuracies were comparatively lower in this setup. Specifically, the RF classifier achieved an accuracy of 93.68%, while DT, SVM, LR, and KNN yielded accuracies of 86.4%, 92.17%, 90.79%, and 86.81%, respectively. These results indicate that the split validation technique provided more robust performance metrics for our dataset. To enhance the performance of the models, we preprocessed the data using MinMax Scaler, Synthetic Minority Oversampling Technique (SMOTE) for balancing the dataset, and Recursive Feature Elimination (RFE) for feature selection. Additionally, hyperparameters were fine-tuned using the GridSearchCV algorithm to optimize the models.

In the case of deep learning (DL) algorithms, we followed similar preprocessing and optimization techniques. The Feedforward Neural Network (FNN) achieved the highest accuracy of 98.48%, demonstrating its effectiveness in handling the complexity of the dataset. For Convolutional Neural Networks (CNN), the best parameters yielded an accuracy of 93.84%. When Long Short-Term Memory (LSTM) layers were integrated with CNN, the model achieved an average accuracy of 95%, with a ROC-AUC score of 98%, indicating strong classification performance. The Gated Recurrent Unit (GRU) model also performed well, achieving an accuracy of 920% and an F1-score of 90%. Finally, the Multilayer Perceptron (MLP) model achieved an accuracy of 92.47%, accompanied by the highest ROC-AUC score of 98.16%, further validating its robustness in PCOS detection.

These results highlight the effectiveness of both ML and DL techniques in detecting PCOS, with RF and FNN emerging as the top-performing models. The preprocessing steps, including SMOTE and RFE, along with hyperparameter tuning, played a critical role in enhancing model performance. However, the split validation technique consistently provided better results compared to K-fold cross-validation, suggesting its suitability for this specific dataset. Future work could explore the integration of ensemble methods and advanced DL architectures to further improve accuracy and generalization capabilities.

3. SYSTEM ARCHITECTURE

3.1 OVERALL SYSTEM DESIGN



3.2 HARDWARE AND SOFTWARE REQUIREMENTS

We developed our model using the Python programming language in Jupyter Notebook. Python, being a versatile and widely-used language, offers a rich ecosystem of built-in libraries for machine learning (ML) and deep learning (DL). For this project, we utilized several key libraries, including pandas for data manipulation, numpy for numerical computations, scikit-learn (sklearn) for implementing ML algorithms, TensorFlow for building DL models, and matplotlib for data visualization. These libraries provided the necessary tools and functionalities to efficiently preprocess data, train models, and evaluate their performance.

3.3 DATA SOURCES AND PREPROCESSING

The dataset used in this study was collected from Kaggle, comprising physical and clinical parameters essential for diagnosing Polycystic Ovary Syndrome (PCOS) and infertility-related issues. The data was gathered from 10 different hospitals across Kerala, India, and includes records from 541 patients with 41 parameters[16]. Among these patients, 341 were healthy, while 177 were diagnosed with PCOS, representing approximately 32.7% of the dataset. This imbalance in the target variable necessitated careful preprocessing to ensure robust model performance.

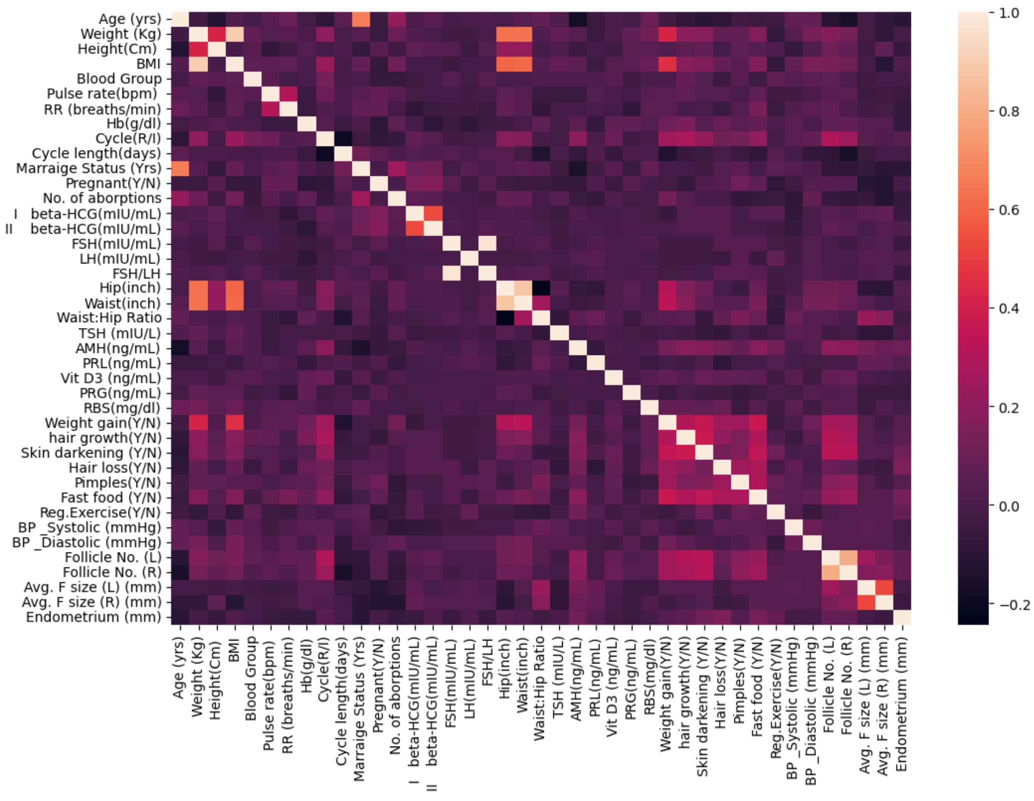


Fig. Heatmap for Correlation between each features

The dataset contained missing values in some columns, which were addressed using median imputation. Median imputation was chosen due to its robustness to outliers, ensuring that the central tendency of the data was preserved without being skewed by extreme values. This approach is particularly suitable for clinical datasets, where outliers may represent rare but valid cases.

The parameters in the dataset were measured on different scales, which could lead to biased model performance. To address this, we normalized the dataset using the MinMax scaling algorithm. MinMax scaling transforms the data into a fixed range, typically [0, 1], by subtracting the minimum value and dividing by the range (maximum minus minimum). This normalization technique is crucial for ensuring that all features contribute equally to the model's learning process, preventing features with larger magnitudes from dominating those with smaller magnitudes.

The dataset exhibited a significant class imbalance, with only 32.7% of the samples belonging to the PCOS-positive class. To mitigate this imbalance and improve the model's ability to generalize, we applied the Synthetic Minority Oversampling Technique (SMOTE). SMOTE generates synthetic samples for the minority class by interpolating between existing instances, thereby balancing the class distribution. This technique is particularly effective in medical datasets, where minority classes often represent critical cases that require accurate detection. By oversampling the minority class, SMOTE ensures that the model is not biased toward the majority class, leading to more reliable and accurate predictions.

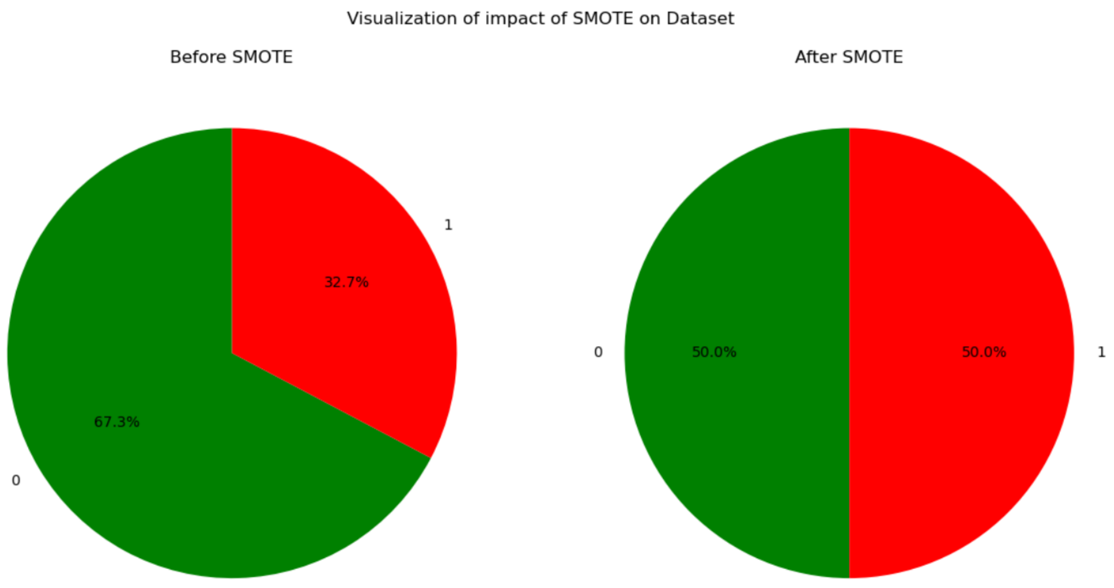


Fig. Visualization of impact of SMOTE on DATASET

After applying SMOTE to address class imbalance, we proceeded to eliminate unnecessary parameters to enhance the model's efficiency and accuracy. The dataset initially contained 41 parameters, not all of which were relevant for the model. To identify and retain the most significant features, we employed the Recursive Feature Elimination (RFE) technique. RFE is a feature selection method that recursively removes the least important features based on their contribution to the model's performance, ultimately retaining the most informative ones. Through this process, 6 parameters were eliminated, leaving us with 35 key parameters for model training. This reduction in dimensionality not only improved computational efficiency but also ensured that the model focused on the most relevant features, thereby enhancing its predictive accuracy.

Target class and feature names from dataset:

No.	Name of the Parameter	No.	Name of the Parameter
1	PCOS(Y/N) → Target Class	22	Waist:Hip Ratio
2	Age (yrs)	23	TSH (mIU/L)
3	Weight (Kg)	24	AMH(ng/mL)
4	Height(Cm)	25	PRL(ng/mL)
5	BMI	26	Vit D3 (ng/mL)
6	Blood Group	27	PRG(ng/mL)
7	Pulse rate(bpm)	28	RBS(mg/dl)
8	RR (breaths/min)	29	Weight gain(Y/N)
9	Hb(g/dl)	30	hair growth(Y/N)
10	Cycle(R/I)	31	Skin darkening (Y/N)
11	Cycle length(days)	32	Hair loss(Y/N)
12	Marriage Status (Yrs)	33	Pimples(Y/N)
13	Pregnant(Y/N)	34	Fast food (Y/N)
14	No. of abortions	35	Reg.Exercise(Y/N)
15	I beta-HCG (mIU/mL)	36	BP _Systolic (mmHg)
16	II beta-HCG (mIU/mL)	37	BP _Diastolic (mmHg)
17	FSH(mIU/mL)	38	Follicle No. (L)
18	LH(mIU/mL)	39	Follicle No. (R)
19	FSH/LH	40	Avg. F size (L) (mm)
20	Hip(inch)	41	Avg. F size (R) (mm)
21	Waist(inch)	42	Endometrium (mm)

4. METHODOLOGY

4.1 THEORETICAL FOUNDATIONS

4.1.1 Median Imputation

When a dataset contains missing values, represented in Python as NaN (Not a Number), further processing or analysis cannot proceed without addressing these gaps. Handling missing values is a critical step in data preprocessing. Median imputation is one such technique where missing values are replaced with the median of the corresponding feature across the dataset. This method is particularly advantageous when the data contains outliers or extreme values, as the median is less sensitive to such anomalies compared to the mean. By using the median, the central tendency and overall distribution of the data are preserved, ensuring that the imputed values do not distort the dataset's statistical properties.

4.1.2 MinMax normalization

In many datasets, features may not lie within a fixed range, with some features spanning significantly larger or smaller values than others. This disparity can pose challenges for machine learning algorithms, particularly those relying on gradient descent for optimization. When features have vastly different scales, the cost function's landscape becomes uneven, making it difficult for gradient descent to converge efficiently. MinMax scaling addresses this issue by normalizing features to a fixed range, typically [0, 1]. This is achieved by subtracting the minimum value of the feature and dividing by its range (the difference between the maximum and minimum values). By scaling features to a common range, MinMax normalization prevents features with larger magnitudes from dominating those with smaller magnitudes, ensuring that all features contribute equally to the model's learning process. This preprocessing step makes the dataset more suitable for training machine learning models. Mathematically, for a feature X, the scaled value X_{scaled} is calculated as:

$$X_{scaled} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

4.1.3 SMOTE (Synthetic Minority Oversampling Technique)

In classification tasks, datasets often suffer from class imbalance, where one class (the minority class) is significantly underrepresented compared to the other (the majority class). For instance, a dataset might contain only 5-10% samples of the minority class and 90-95% of the majority class. Training a model on such imbalanced data can lead to high accuracy, but the model may fail to generalize well to the minority class, as it becomes biased toward the majority class due to its overrepresentation. To address this issue, SMOTE is employed. SMOTE generates synthetic samples for the minority class by interpolating between existing minority class samples and their nearest neighbors. This oversampling technique balances the dataset, ensuring that the model is exposed to a more equitable distribution of both classes during training. As a result, the model becomes better at detecting minority class instances, improving its overall performance and robustness. Mathematically, for a minority class sample x_i and its neighbor x_{zi}, the synthetic sample x_{new} is generated as:

$$x_{new} = x_i + \lambda(x_{zi} - x_i)$$

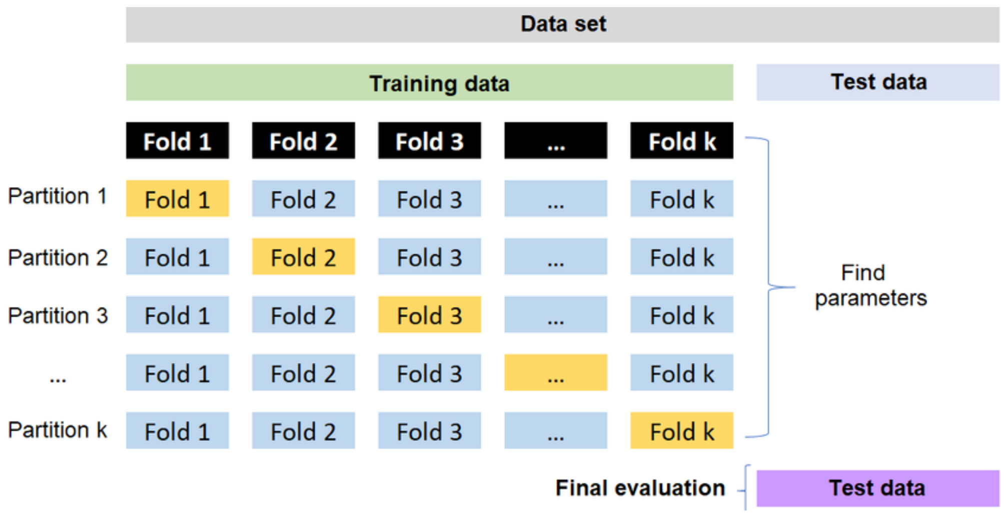
where λ is a random number between 0 and 1.

4.1.4 RFE (Recursive Feature Elimination)

In datasets with a large number of features, not all features contribute equally to the model's predictive power. Some features may be redundant, irrelevant, or even noisy, which can degrade model performance and increase computational complexity. Manually selecting important features can be time-consuming and subjective. Recursive Feature Elimination (RFE) is a powerful feature selection technique that automates this process. RFE works by recursively removing the least important features and rebuilding the model until the desired number of features is reached. The importance of features is typically determined by their coefficients in linear models or their contribution to the model's performance in tree-based algorithms. By eliminating redundant or irrelevant features, RFE enhances the model's computational efficiency and improves its accuracy by focusing on the most informative attributes. This process also aids in interpreting the model by highlighting the key features that drive predictions.

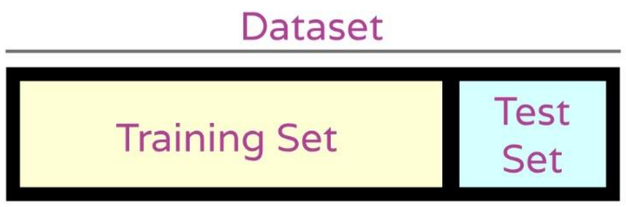
4.1.5 K-Fold Cross Validation

K-Fold Cross Validation is essential for reliably evaluating a model's performance on unseen data. Without it, models may overfit to the training data, leading to overly optimistic performance estimates. This technique ensures the model's performance is consistent across different subsets of the data, providing a more accurate assessment of its generalization ability. K-Fold Cross Validation splits the dataset into k equal-sized folds. The model is trained k times, each time using k−1 folds for training and the remaining fold for validation. The final performance metric is the average of the metrics from all k iterations, ensuring every data point is used for both training and validation.



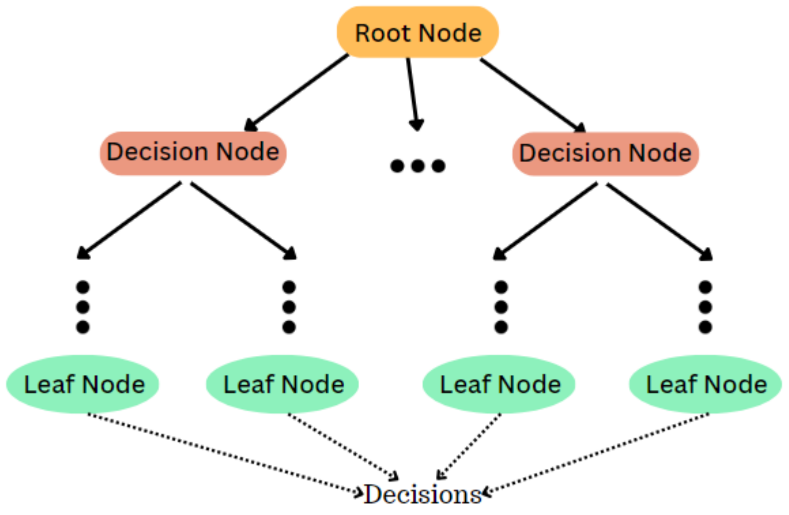
4.1.6 Split Validation

Split validation is crucial for evaluating a model's performance on unseen data. Without it, a model may perform well on training data but fail to generalize to new data. This technique helps detect overfitting and provides a realistic assessment of the model's performance. Split validation divides the dataset into a training set and a testing set. Typically, 70-80% of the data is used for training, and the remaining 20-30% is reserved for testing. The model is trained on the training set and evaluated on the testing set to assess its generalization ability.



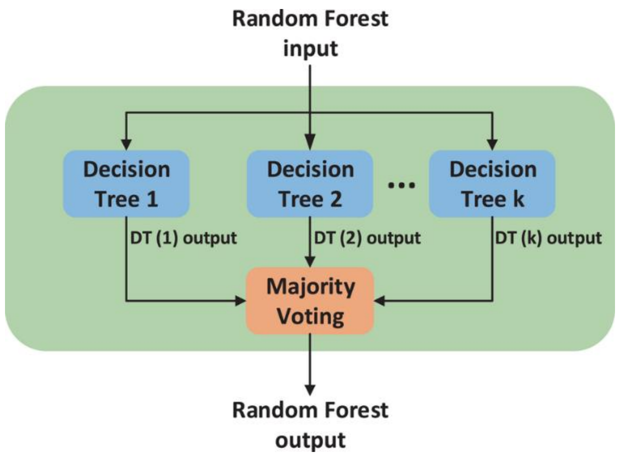
4.1.7 Decision Tree (DT)

Decision Trees are simple yet effective models for both classification and regression. They are interpretable and can handle non-linear relationships, making them useful for understanding data patterns and feature importance. A Decision Tree splits the data into subsets based on feature values, creating a tree-like structure. Each internal node represents a decision based on a feature, and each leaf node represents a prediction. The tree is built by recursively splitting the data to maximize information gain or minimize impurity.



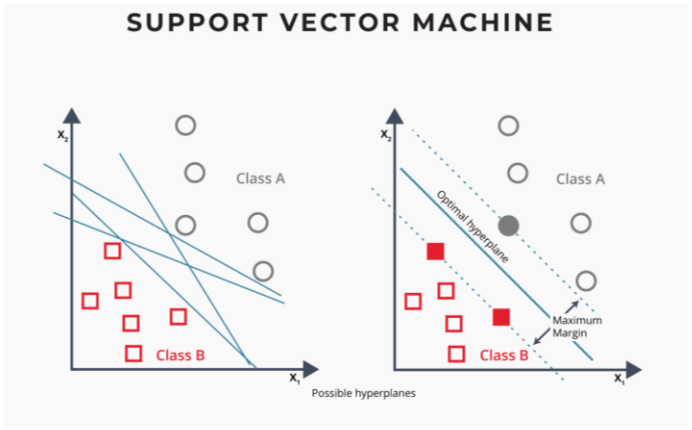
4.1.8 Random Forest (RF)

Random Forest is a powerful ensemble learning method that addresses the limitations of single decision trees, such as overfitting and instability. By combining multiple decision trees, RF improves prediction accuracy and robustness, making it suitable for both classification and regression tasks. Random Forest builds multiple decision trees during training, each trained on a random subset of the data and features. The final prediction is obtained by averaging the predictions using majority voting for classification. This approach reduces overfitting and enhances generalization.



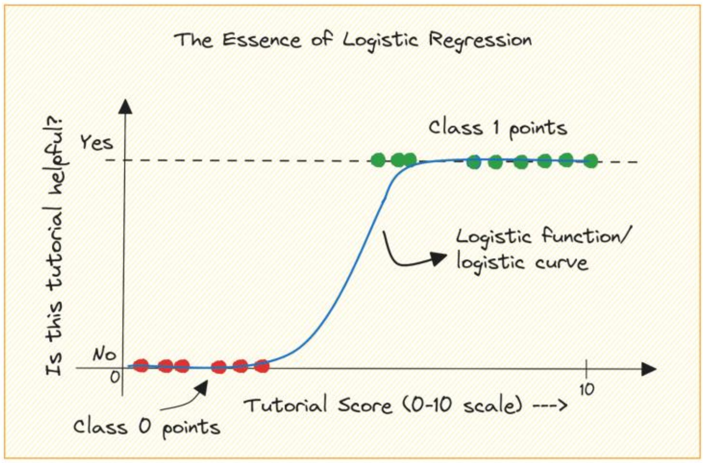
4.1.9 Support Vector Machine (SVM)

SVM is a robust algorithm for classification and regression, particularly effective in high-dimensional spaces. It finds the optimal hyperplane that maximizes the margin between classes, making it suitable for complex datasets. SVM constructs a hyperplane or set of hyperplanes in a high-dimensional space to separate classes. For non-linear data, it uses kernel functions to transform the data into a higher-dimensional space where separation is possible.



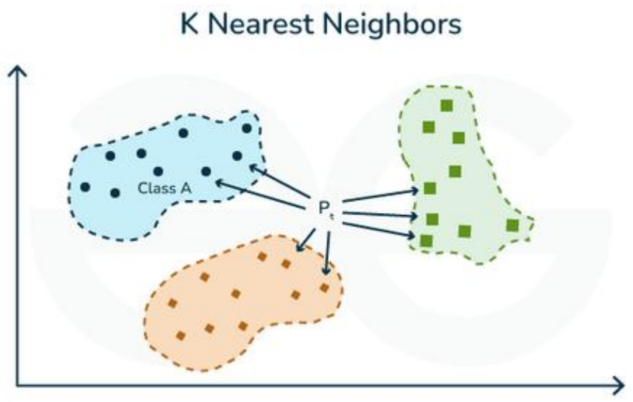
4.1.10 Logistic Regression (LR)

Logistic Regression is a fundamental algorithm for binary classification. It models the probability of a binary outcome, making it useful for problems like disease prediction or spam detection. Logistic Regression uses the logistic function to map input features to a probability between 0 and 1. The model is trained by maximizing the likelihood of the observed data.



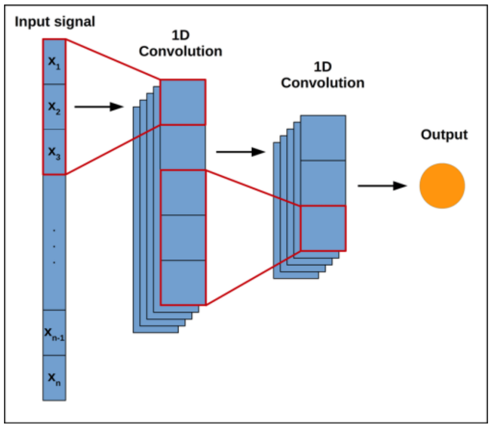
4.1.11 K-Nearest Neighbors (KNN)

KNN is a simple, non-parametric algorithm used for classification and regression. It is useful when the data distribution is unknown or when interpretability is important. KNN predicts the output for a data point based on the majority class (for classification) or average value (for regression) of its k*k* nearest neighbors in the feature space.



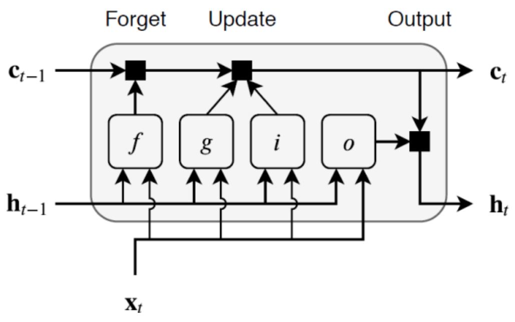
4.1.12 1D Convolutional Neural Network (1D CNN)

A 1D CNN is a type of convolutional neural network designed to process one-dimensional data, such as time series, signals, or sequences. Convolutional Neural Network consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers. The Convolutional layer applies filters to the input image to extract features, the Pooling layer downsamples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.



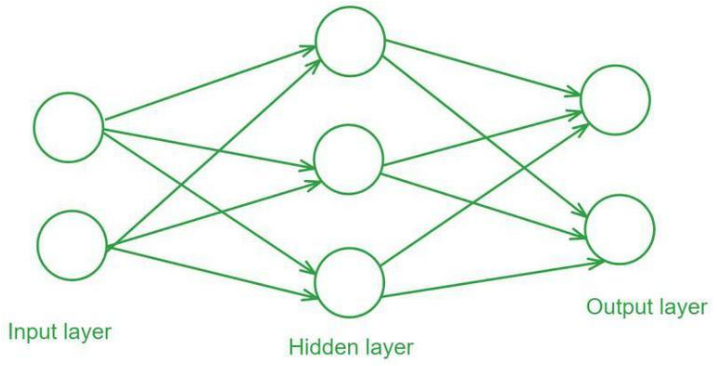
4.1.16 LSTM (Long Short-Term Memory)

Long Short-Term Memory (LSTM) is an advanced type of Recurrent Neural Network (RNN) designed to capture long-term dependencies in sequential data. The architecture of an LSTM includes a memory cell that is regulated by three gates: the input gate, the forget gate, and the output gate. These gates determine what information should be added to the memory cell, removed or outputted. This mechanism allows LSTM networks to selectively learn or discard information as it moves through the network, enabling them to effectively learn and remember patterns over longer periods.



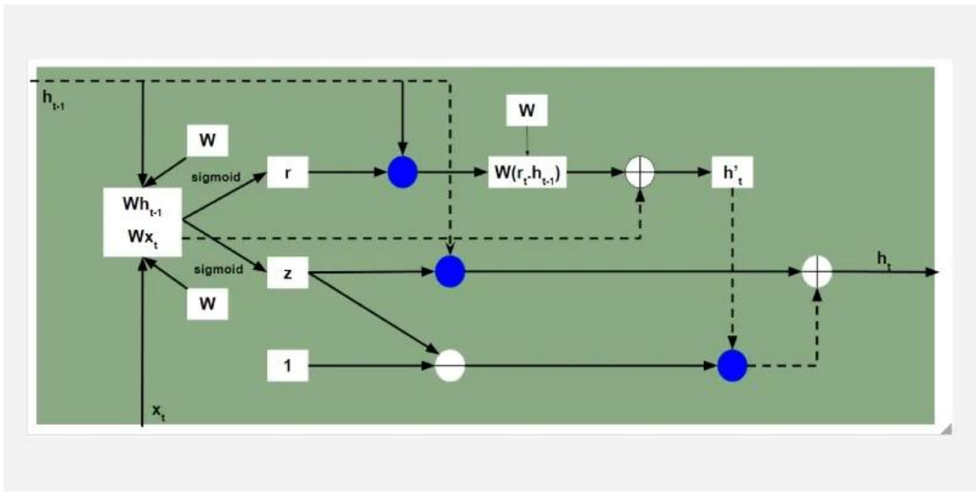
4.1.17 FNN (Feedforward Neural Network)

A Feedforward Neural Network (FNN) is the simplest type of artificial neural network where information flows in one direction—from input to output—without loops or feedback connections. It takes the input data, processes it through layers of neurons, applies weights and activation functions, and gives a result. There's no memory or feedback, so each input is treated independently.



4.1.18 Gated Recurrent Unit (GRU)

A GRU is a simplified version of an LSTM. Instead of three gates, a GRU uses two which are Reset Gate and Update Gate. It combines the cell state and hidden state into a single "memory" unit, making it faster and easier to train while still being effective at remembering long-term patterns.



4.1.19 Multilayer Perceptron (MLP)

A Multilayer Perceptron (MLP) is a type of feedforward neural network with multiple layers of neurons. It's one of the simplest forms of deep learning models, designed for tasks like classification and regression. The architecture consists of an input layer, one or more hidden layers, and an output layer. Each neuron in the hidden layers applies weights to the input, adds a bias, and passes the result through a nonlinear activation function, such as the Rectified Linear Unit (ReLU).

4.2 EXPERIMENTAL SETUP

Our experiment focuses on comparing different algorithms for detecting Polycystic Ovary Syndrome (PCOS) and evaluating their performance. At first the dataset was preprocessed following the steps mentioned in Section 3.3. After preparing the data, we move on to feature selection.

For this, we use Recursive Feature Elimination (RFE) with a RandomForestClassifier to repeatedly remove the least important features. RFE was applied to get the top 35 features. SelectKBest was also applied to get the 35 most important features for comparison purposes.

Once feature selection is complete, we train and evaluate several machine learning models namely: Random Forest, Decision Tree, Support Vector Machine (SVM), Linear Discriminant Analysis, and Logistic Regression. Stratified K-Fold Cross-Validation was used to keep the class distribution balanced across different splits of the data. This validation gives us key metrics like accuracy, precision, recall, and F1-score.

Hyperparameters were fined tuned using GridSearchCV to get the best combinations across different combinations of settings for each model. The tuned hyperparameters are namely: For Random Forest: n_estimators (number of trees), max_depth (tree depth) ,min_samples_split (minimum samples needed to split a node) and min_samples_leaf . For Decision Tree: max_depth, min_samples_split and min_samples_leaf. For Support Vector Machine (SVM): C (regularization strength) and kernel. For Logistic Regression: C (regularization strength) and solver (optimization algorithm). For K-Nearest Neighbors (KNN): n_neighbors (number of neighbors), weights (how neighbors contribute to predictions) and metric (distance measurement method). Using those hyperparameters models were retrained and evaluated on the test set. Split Validation was also done which includes a train-test split validation step, where the dataset is split into training and testing sets using train_test_split. The models are trained on the training set and evaluated on the test set.

After completing the initial classification phase, we extend our analysis by incorporating clustering techniques. We apply three clustering algorithms: K-Means, DBSCAN, and Gaussian Mixture Model (GMM). The cluster labels generated from these algorithms are added as a new feature to the dataset. After generating the labels, we retrain the Random Forest Classifier including the labels as a feature and optimize its hyperparameters using GridSearchCV. The best-performing model is selected based on accuracy, and its performance is also evaluated using 5-fold cross-validation. Additionally, the final model is tested on a hold-out test set, where key performance metrics such as accuracy, F1-score, precision, and recall are calculated to measure its effectiveness. Hyperparameter tuning was also performed for the clustering algorithms. For K-Means, we experiment with different numbers of clusters (n_clusters), for GMM, we adjust the number of components (n_components). The best-performing clustering configurations are selected based on accuracy. The performance metrics for each clustering algorithm—K-Means, DBSCAN, and GMM—are determined.

Afterwards we incorporate the use of CNN for our classification use. After preprocessing, feature selection was performed using Recursive Feature Elimination (RFE) with a Random Forest base estimator to identify the most relevant features for classification. Next, a 1D Convolutional Neural Network (1D CNN) was designed using Keras. The model architecture consisted of a total of 17 layers. The model begins with three sets of convolutional layers (Conv1D) for feature extraction, each followed by a BatchNormalization layer for stability, a MaxPooling1D layer for downsampling, and a Dropout layer for regularization. After these convolutional blocks, a Flatten layer converts the output into a 1D vector. Finally, two Dense layers are used for classification. To evaluate model performance, two validation strategies were employed: Stratified K-Fold Cross-Validation and a Train-Test Split, where 80% of the data was used for training and 20% for testing. Hyperparameter tuning was conducted using Randomized Search with a KerasClassifier. Key parameters such as the number of filters, neurons, and dropout rates were adjusted to enhance performance. The best model was selected based on evaluation metrics like accuracy, ROC-AUC score, confusion matrices, and classification reports.

Next, Multi-Layer Perceptron (MLP) classifier was used for prediction. After preprocessing, two validation strategies were employed: Stratified K-Fold Cross-Validation and a Train-Test Split, where 80% of the data was used for training and 20% for testing. Initially the MLP model was trained with two hidden layers with 64 and 32 neurons, tanh was used as the activation function for 500 iterations. Afterwards, GridSearchCV was used to optimize the MLP classifier's hyperparameters like hidden layer sizes, activation function, and solver. The results were evaluated using metrics like accuracy, precision, recall, F1-score, and ROC-AUC score to assess the model's performance.

For LSTM classifier, we selected 30 features using RFE after the initial preprocessing. Afterwards our dataset was reshaped to a 3D array which was suitable to be used as an input for our LSTM model. Then like before we employed both Stratified 5-Fold Cross-Validation and Split Validation. The LSTM model begins with an initial LSTM layer containing 32 units and configured to output a sequence for subsequent layers. This layer employs the Glorot uniform initializer for weight initialization and incorporates L2 regularization to prevent overfitting. Following this layer, a Batch Normalization layer is applied to stabilize and accelerate training by normalizing activations. A Dropout layer with a rate of 0.2 is added to minimize overfitting by randomly deactivating some input units. The model then progresses through three more LSTM layers, each with a decreasing number of units (32, 16, and 16 respectively). Between each LSTM layer, there are Batch Normalization and Dropout layers for regularization and training stability. The final LSTM layer contains 8 units and is configured to produce a single output vector instead of a sequence. It is followed by Batch Normalization and Dropout layers. Ultimately, the model culminates in a Dense layer with a single unit and a sigmoid activation function. This layer serves as the output layer, generating a probability score that represents the model's prediction for PCOS. This model was evaluated using both the mentioned validation methods. Afterwards, Keras Tuner was used to optimize the model's hyperparameters, including the number of units in LSTM layers, dropout rates, and learning rate. Hyperband tuning algorithm was utilized for efficient search of the hyperparameter space. The best model was determined and trained using the preprocessed and reshaped data. The model's performance was evaluated using metrics such as accuracy, ROC-AUC score, confusion matrix, and classification report.

For GRU, we selected 30 features using RFE after the initial preprocessing. The preprocessed data was reshaped to fit the input requirements of the GRU model, specifically into a 3D format. A sequential GRU-based neural network designed for binary classification was defined. It begins with an input GRU layer containing 64 units, followed by Batch Normalization and a Dropout layer with a rate of 0.1. This is followed by three more GRU layers with 80, 24, and 36 units, respectively. Each of these GRU layers is also followed by Batch Normalization and Dropout layers with varying dropout rates (0.4, 0.3, and 0.4). All GRU layers except the last one use the 'relu' activation function and have 'sigmoid' as the recurrent activation function. They also use the Glorot Uniform initializer and L2 regularization for the kernel weights. The final layer is a Dense layer with a single unit and a 'sigmoid' activation, producing the binary classification output. This model was evaluated using both Stratified 5-Fold Cross-Validation and Split Validation. Afterwards, Keras Tuner was used to optimize the model's hyperparameters, including the number of units in LSTM layers, dropout rates, and learning rate. Hyperband tuning algorithm was utilized for efficient search of the hyperparameter space. The best model was determined and trained using the preprocessed and reshaped data. The model's performance was evaluated using metrics such as accuracy, ROC-AUC score, confusion matrix, and classification report.

Next, An FNN model was created using the Keras library for comparison purposes. The FNN model begins with an input layer consisting of 30 neurons. It uses the ReLU activation function, GlorotUniform kernel initializer, and L2 regularization for weight decay. Then, there are three hidden layers. The first hidden layer has 192 neurons with ReLU activation, L2 regularization, batch normalization, and a dropout rate of 0.5. The second hidden layer contains 112 neurons with ReLU activation, L2 regularization, batch normalization, and a dropout rate of 0.4. The third hidden layer has 16 neurons with ReLU activation, L2 regularization, batch normalization, and a dropout rate of 0.3. Finally, the model concludes with an output layer containing a single neuron with a sigmoid activation function. This model was trained on our preprocessed data for 50 epochs with a batch size of 16. Then the model was evaluated using both Stratified 5-Fold Cross-Validation and Split Validation. Keras Tuner was used for hyperparameter tuning, focusing on finding the optimal number of neurons and dropout rates for each hidden layer. A random search strategy was employed, exploring a predefined range of hyperparameter values for maximum validation accuracy. Key metrics, including accuracy, ROC-AUC score, confusion matrix, and classification report, were calculated to assess the model.

4.3 ASSUMPTIONS AND CONSTRAINTS

Assumptions

1. Data Quality: The dataset is accurate, reliable, and can be generalized for the whole population in different regions of the world. values are. It is also assumed that Median Imputation wields the best result for the missing data.
2. Model Performance: Hyperparameter tuning using GridSearchCV and Randomized Search will yield the optimal model configurations.

Constraints and Limitations

1. Dataset Limitations: The dataset is geographically limited to patients from Kerala, India, which may limit the generalizability of the models to other populations or regions. Also more datapoints maybe required for effective model training.
2. Class Imbalance: Despite using SMOTE, the synthetic samples generated may not fully capture the complexity of the minority class, potentially leading to overfitting or biased predictions.

5. RESULTS AND ANALYSIS

5.1 PERFORMANCE METRICS

To evaluate the performance of the models, several widely recognized metrics were carefully selected. These metrics provide comprehensive insights into the models' effectiveness across different aspects of classification tasks. The chosen performance metrics are described below:

1. Accuracy:

Accuracy measures the proportion of correctly classified instances (both true positives and true negatives) out of the total number of instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is a straightforward metric that provides an overall measure of model performance. However, it can be misleading in imbalanced datasets, where the majority class dominates.

2. Precision:

Precision quantifies the proportion of correctly predicted positive instances (true positives) out of all instances predicted as positive (true positives and false positives).

$$Precision = \frac{TP}{TP + FP}$$

Precision is particularly important in scenarios where false positives are costly.

3. Recall:

Recall measures the proportion of actual positive instances that are correctly identified by the model (true positives out of true positives and false negatives).

$$Recall = \frac{TP}{TP + FN}$$

Recall is critical in applications where missing positive instances (false negatives) is undesirable.

4. F1-Score:

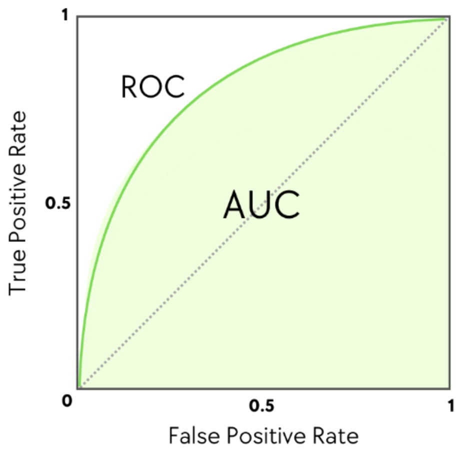
The F1-score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The F1-score is particularly useful in imbalanced datasets, as it balances the trade-off between precision and recall.

5. ROC-AUC (Receiver Operating Characteristic - Area Under Curve):

The ROC-AUC score evaluates the model's ability to distinguish between classes by plotting the true positive rate (recall) against the false positive rate at various threshold settings.



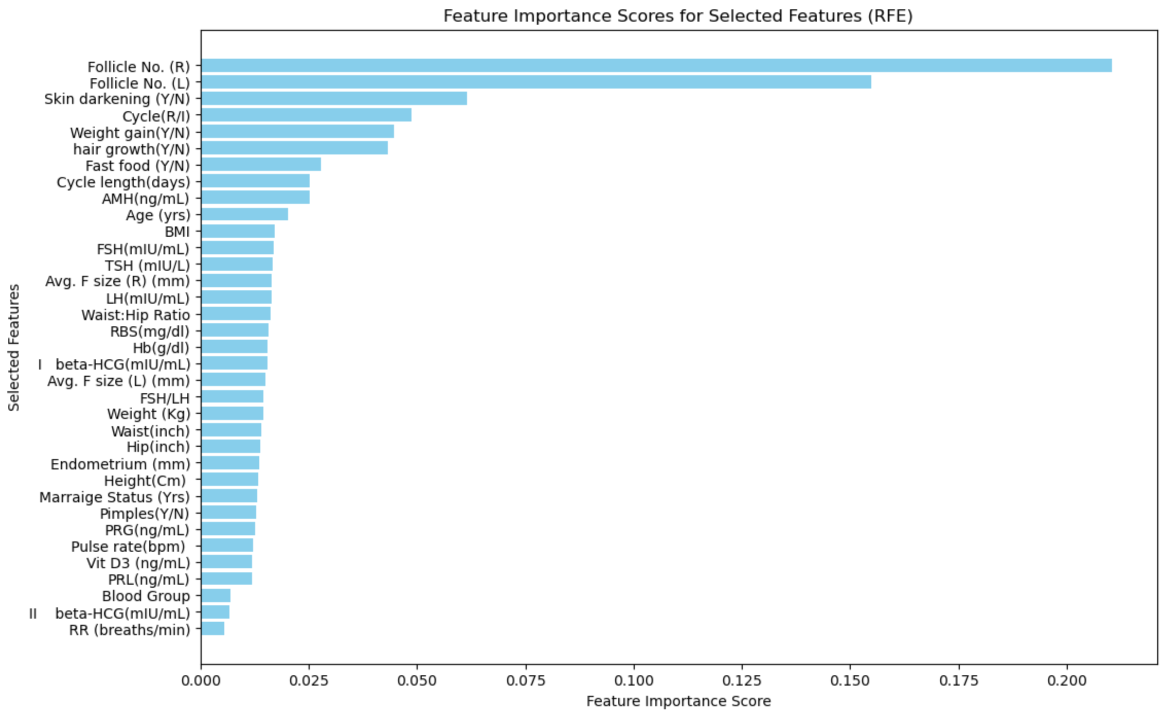
The area under this curve (AUC) provides a single scalar value representing the model's performance. ROC-AUC is a robust metric for evaluating classification models, especially when the class distribution is imbalanced. It provides insights into the model's performance across all classification thresholds.

5.2 EXPERIMENTAL RESULTS

In this section we will analyze our results and findings from our experiment.

5.2.1 Feature selection:

The dataset used in this study comprises a total of 41 features. To enhance computational efficiency and improve model performance, we performed feature selection to identify the most relevant features. After evaluation, we selected the top 35 features for model training. Two distinct methods were employed for feature selection: Recursive Feature Elimination (RFE) and SelectKBest using Mutual Information (MI). Mutual Information (MI) is a filter-based feature selection method that evaluates the statistical dependency between each feature and the target variable. It does not require training a model and is computationally efficient, making it suitable for identifying the most informative features. On the other hand, Recursive Feature Elimination (RFE) is a wrapper-based feature selection method that requires training a model—in this case, a Random Forest classifier—and iteratively removes the least important features based on their contribution to model performance. This approach ensures that the selected features are optimal for the specific model being used. The scores of the features obtained from these two methods are presented below, highlighting the importance of each feature in predicting PCOS.



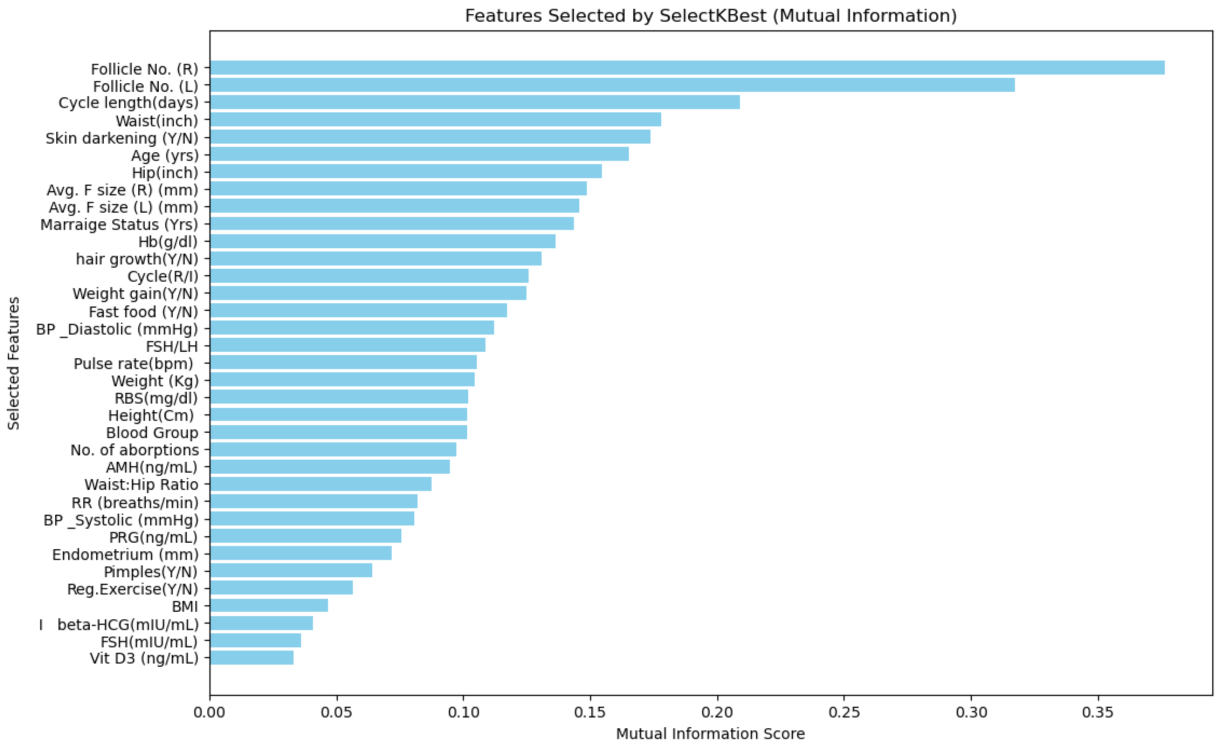


Fig. Scores of Selected Features using RFE and Select K best with MI

In both methods, selected features are almost same. For our final model we used features obtained from RFE method.

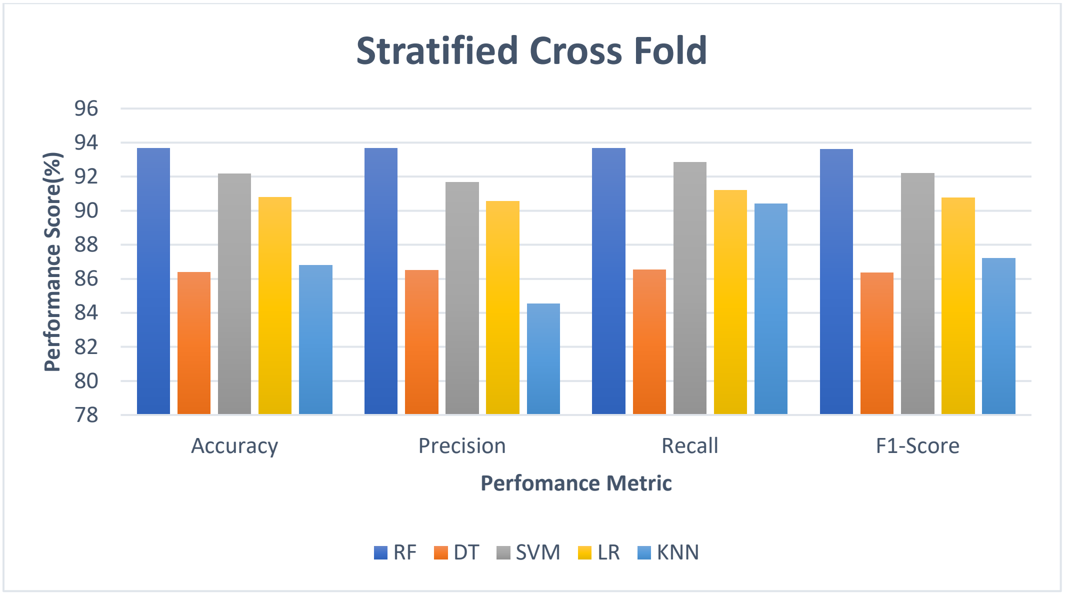
5.2.2 Performance of ML classifiers:

In this subsection, the performance results from ML classifiers have been presented.

a. Stratified K-fold cross-validation:

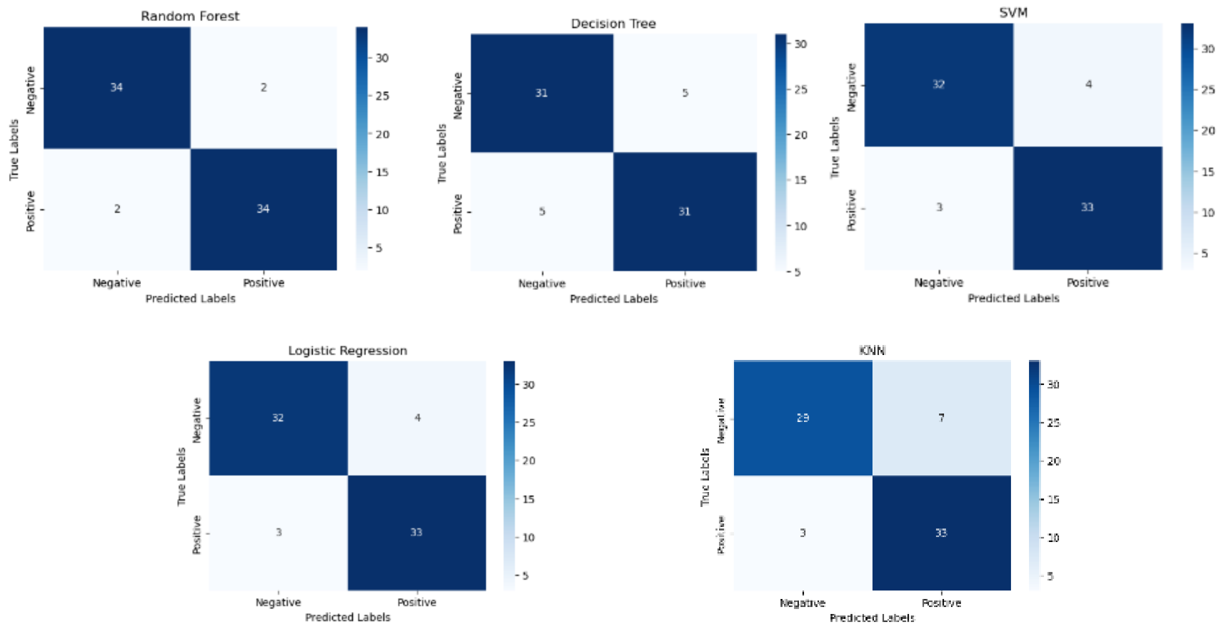
Model	Performance Metric	Mean		Standard Deviation	
		RFE	MI	RFE	MI
RF	Accuracy	93.68%	93.54%	3.11%	2.68%
	Precision	93.67%	93.47%	2.33%	2.75%
	Recall	93.67%	93.67%	5.37%	4.87%
	F1-Score	93.61%	93.49%	3.32%	2.85%
DT	Accuracy	86.4%	87.65%	2.17%	3.26%
	Precision	86.49%	87.99%	3.59%	5.17%
	Recall	86.53%	87.64%	4.87%	5.31%
	F1-Score	86.37%	87.63%	2.39%	3.28%
SVM	Accuracy	92.17%	92.30%	4.04%	3.44%
	Precision	91.67%	91.49%	4.27%	3.75%
	Recall	92.84%	93.39%	5.25%	5.21%
	F1-Score	92.19%	92.35%	4.09%	3.47%
LR	Accuracy	90.79%	91.07%	2.76%	2.91%
	Precision	90.57%	91.54%	2.9%	2.67%
	Recall	91.20%	90.65%	5.9%	6.75%
	F1-Score	90.76%	90.93%	3%	3.29%
KNN	Accuracy	86.81%	87.91%	2.9%	4.15%
	Precision	84.54%	85.55%	3.88%	4.73%
	Recall	90.41%	91.49%	4.45%	5.21%
	F1-Score	87.21%	88.32%	2.78%	3.95%

Overall, the performance scores were comparable between the Recursive Feature Elimination (RFE) and Mutual Information (MI) feature selection methods. The mean scores were calculated over the 10 folds of k-fold cross-validation. However, relying solely on mean scores is insufficient to evaluate a model's performance comprehensively. To gain deeper insights, the standard deviation of the scores was also computed, providing an understanding of the variability in performance across the folds. The Random Forest (RF) model achieved the highest mean accuracy of 93.68% using the RFE method, while the Decision Tree (DT) model yielded the lowest mean accuracy of 86.4% with the same feature selection method. In terms of standard deviation, the Logistic Regression (LR) model exhibited the highest variability (6.75%) when using the MI method, whereas the Decision Tree (DT) model demonstrated the lowest variability (2.17%) with the RFE method. Although the DT model had the lowest mean scores, its relatively low standard deviation indicates consistent performance across all folds. This suggests that while the DT model may not achieve the highest accuracy, it provides stable and reliable predictions. In contrast, the higher standard deviation observed in the LR model suggests greater variability in its performance, which may indicate sensitivity to the dataset's composition in different folds.



RF consistently performed better in each metrics compared to other four models.

Confusion Matrix:



From the confusion matrix, it is evident that False Positive(FP) is highest in Decision Tree(DT) model and lowest in Random Forest(RF) model.

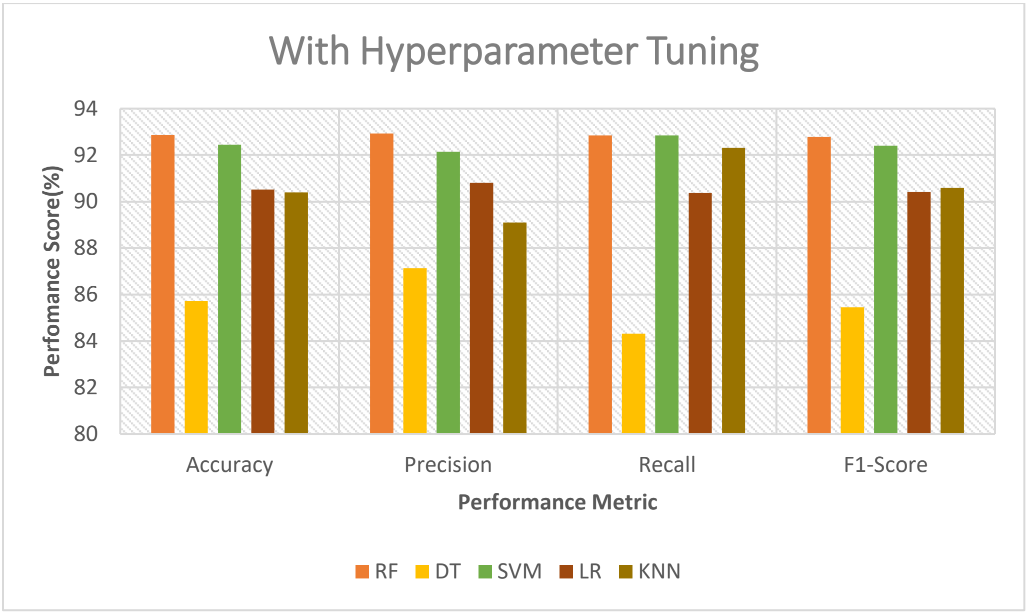
b. HyperParameter Tuning with GridSearchCV:

Model	Performance Metric	Mean	Standard Deviation
RF	Accuracy	92.86%	2.49%
	Precision	92.93%	2.42%
	Recall	92.85%	5.38%
	F1-Score	92.78%	2.27%
DT	Accuracy	85.72%	3.65%
	Precision	87.13%	5.79%
	Recall	84.31%	6.84%
	F1-Score	85.45%	3.94%
SVM	Accuracy	92.45%	4.14%
	Precision	92.14%	4.02%
	Recall	92.85%	6.41%
	F1-Score	92.40%	4.43%
LR	Accuracy	90.52%	3.32%
	Precision	90.80%	3.60%
	Recall	90.37%	6.88%
	F1-Score	90.41%	3.62%
KNN	Accuracy	90.39%	2.87%
	Precision	89.10%	4.48%
	Recall	92.31%	3.81%
	F1-Score	90.58%	2.73%

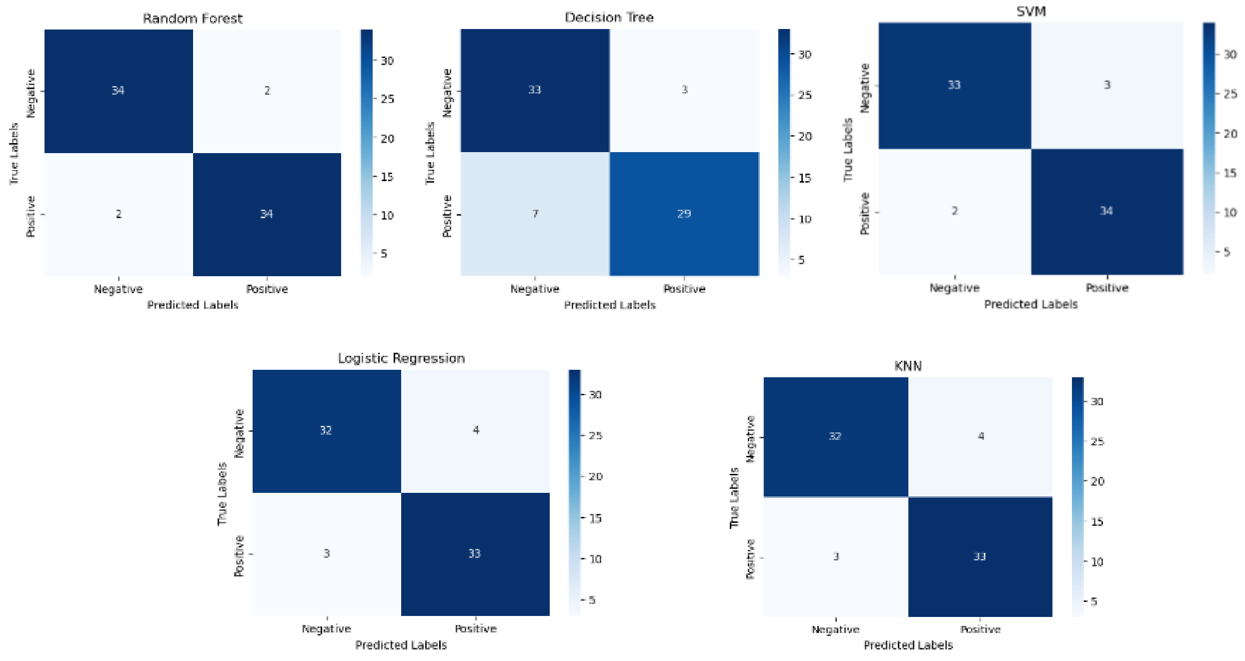
Best parameters found after HyperParameter Tuning:

RF	DT	SVM	LR	KNN
Max Depth = 10	Max Depth = None	C = 1	C = 1	Metric = Manhattan
Min samples leaf = 1	Min samples leaf = 4	Kernel = Poly	Solver = Liblinear	n neighbours = 7
Min samples split = 5	Min samples split = 10			Weights = Distance
n estimators = 300				

After hyperparameter tuning, the results remained largely consistent with those obtained prior to tuning. The Random Forest (RF) model continued to outperform the other models across all performance metrics. It achieved the highest scores in accuracy (92.86%), precision (92.93%), recall (92.855%), and F1-score (92.78%). Additionally, the standard deviations for these metrics were relatively low (2.49%, 2.42%, 5.38%, and 2.27%, respectively), indicating consistent performance across all folds of the cross-validation process. The Support Vector Machine (SVM) model emerged as the closest competitor to RF, with performance metrics of 92.45% accuracy, 92.14% precision, 92.85% recall, and 92.40% F1-score. However, the standard deviations for SVM were notably higher compared to those of RF, suggesting greater variability in its performance across folds. Without considering these standard deviations, the mean scores alone could be misleading, as they do not fully capture the model's stability and reliability.



Confusion Matrix:

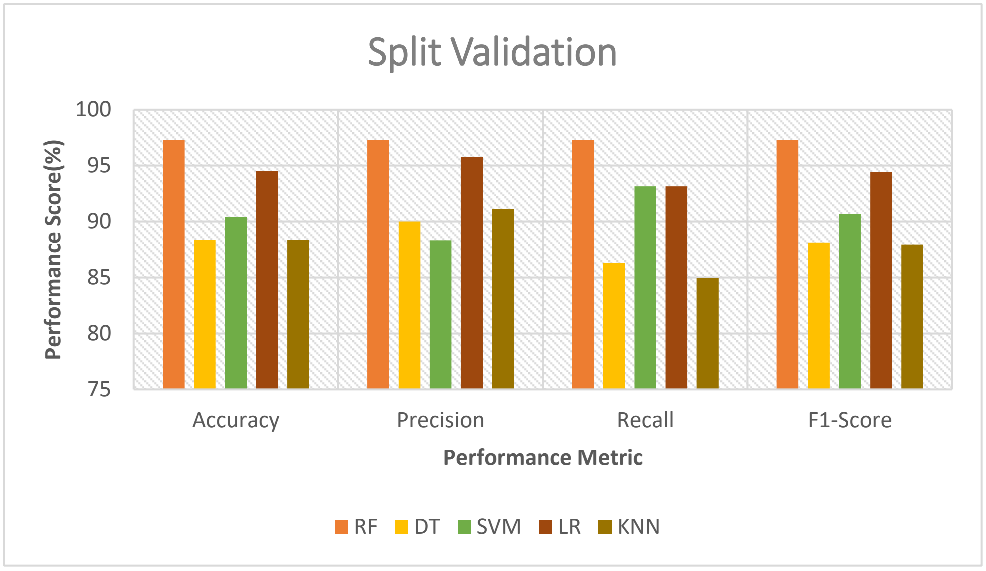


c. Split Validation with HyperParameter Tuning:

Model	Performance Metric	Mean
RF	Accuracy	97.26%
	Precision	97.26%
	Recall	97.26%
	F1-Score	97.26%
DT	Accuracy	88.36%
	Precision	90.00%
	Recall	86.30%
	F1-Score	88.11%
SVM	Accuracy	90.41%
	Precision	88.31%
	Recall	93.15%
	F1-Score	90.67%
LR	Accuracy	94.52%
	Precision	95.77%
	Recall	93.15%
	F1-Score	94.44%
KNN	Accuracy	88.36%
	Precision	91.12%
	Recall	84.93%
	F1-Score	87.94%

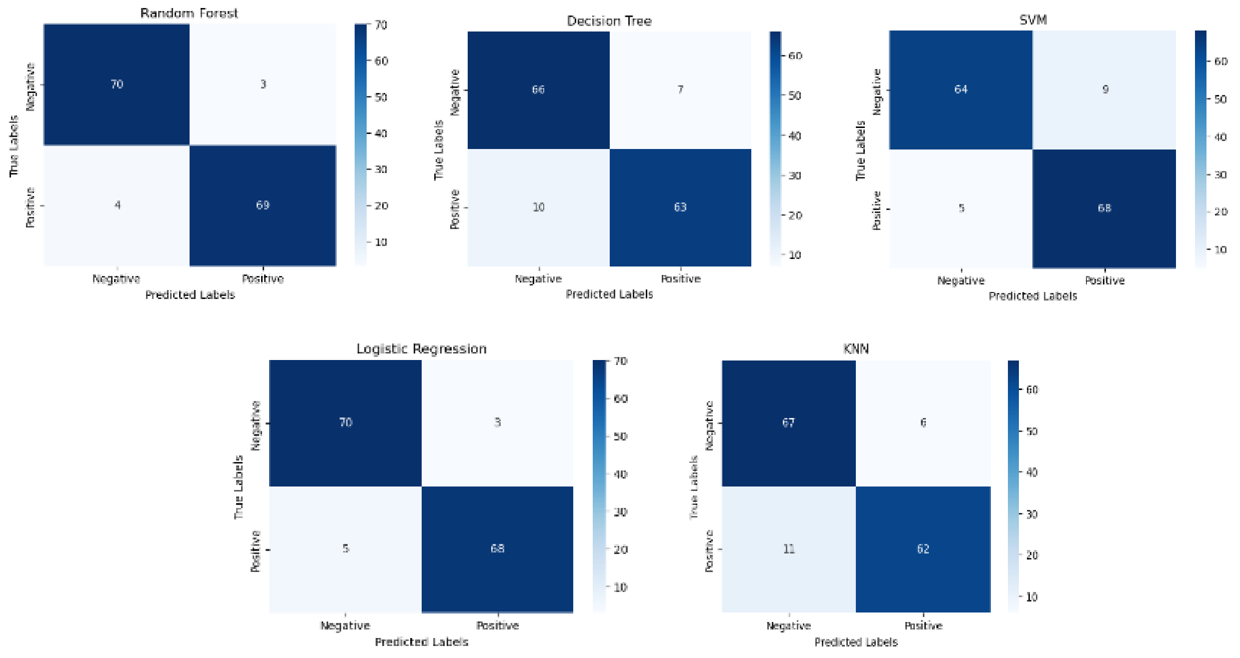
Best parameters found after HyperParameter Tuning:

RF	DT	SVM	LR	KNN
Max Depth = None	Max Depth = None	C = 10	C = 1	Metric = Manhattan
Min samples leaf = 2	Min samples leaf = 4	Kernel = Poly	Solver = Liblinear	n neighbours = 3
Min samples split = 5	Min samples split = 10			Weights = Distance
n estimators = 100				



When using split validation (with an 80:20 train-test ratio) instead of k-fold stratified cross-validation, the performance of all models improved significantly. As expected, the Random Forest (RF) model outperformed the others across all metrics, achieving an accuracy of 97.26%, precision of 95.83%, recall of 94.52%, and an F1-score of 95.17%. The Logistic Regression (LR) model delivered the second-best performance, demonstrating competitive results across the same metrics.

Confusion Matrix:



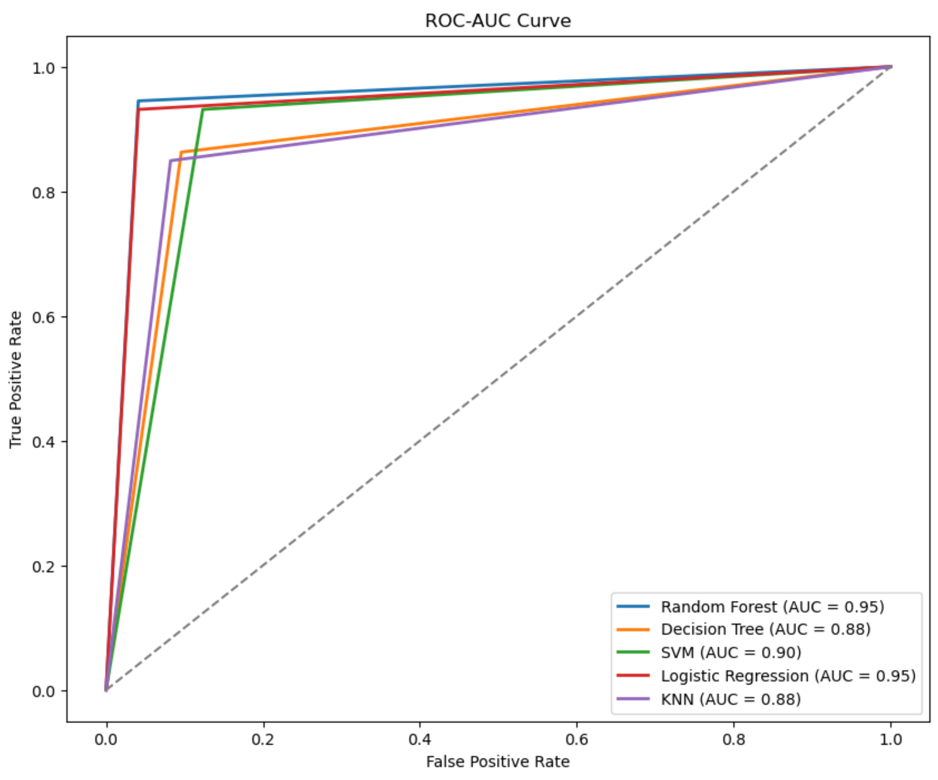


Fig. ROC-AUC curve obtained from ML classifiers

From the ROC-AUC curve, it is evident that the curve for the Random Forest (RF) model is positioned higher than those of the other models, indicating its superior performance with the highest ROC-AUC value (0.95). In contrast, the curve for the K-Nearest Neighbors (KNN) model is the lowest, reflecting its comparatively weaker performance with the lowest ROC-AUC value (0.88). Notably, the Logistic Regression (LR) model demonstrates competitive performance, with its ROC-AUC value being close to that of the RF model.

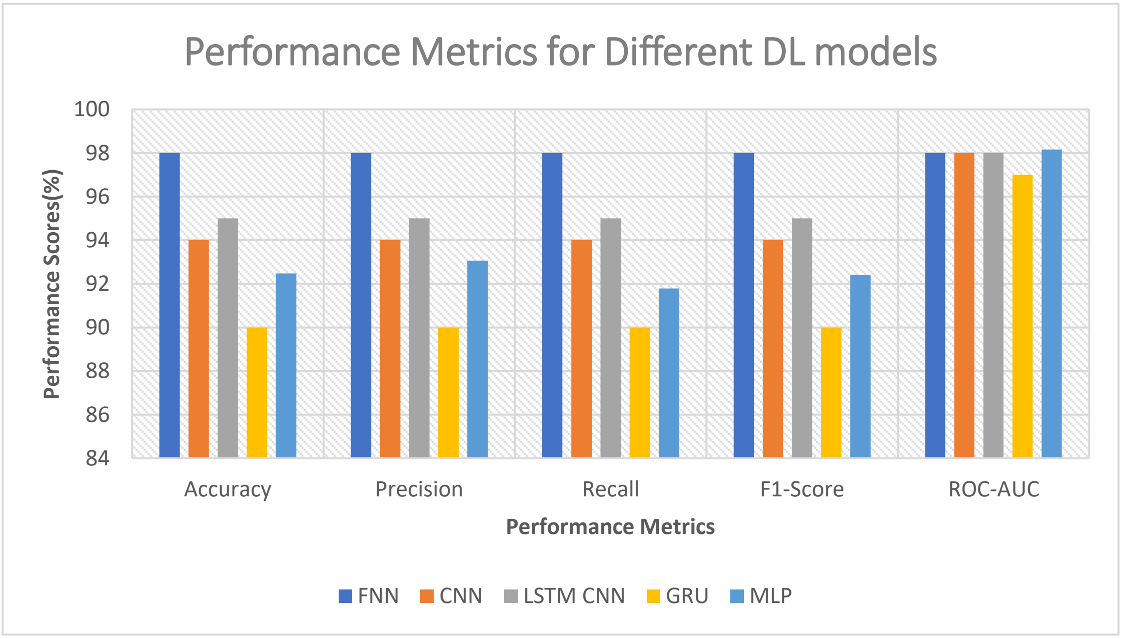
5.2.3 Performance of DL models:

In this subsection, we analyze the findings from the Deep Learning (DL) models. For the DL models, we employed the split validation method (with an 80:20 train-test ratio), as it yielded superior results compared to k-fold cross-validation for this dataset in our earlier experiments with traditional machine learning classifiers. To further optimize the performance of the DL models, we conducted hyperparameter tuning using GridSearchCV and Keras Tuner Random Search. These techniques allowed us to systematically explore the hyperparameter space and identify the optimal configurations for our models.

a. Split validation:

Model Name	Accuracy	Precision	Recall	F1-Score	ROC-AUC
FNN	98%	98%	98%	98%	98%
CNN	94%	94%	94%	94%	98%
LSTM CNN	95%	95%	95%	95%	98%
GRU	90%	90%	90%	90%	97%
MLP	92.47%	93.06%	91.78%	92.41%	98.16%

Although Deep Learning (DL) models are less commonly used for tabular datasets compared to image datasets, they can still be effectively applied to tabular data. In our experiments, the DL models outperformed traditional machine learning (ML) models across all performance metrics. Among the DL models, the Feedforward Neural Network (FNN) achieved the highest scores, surpassing not only other DL models but also all ML models used in our study. The FNN model achieved scores of 98% in accuracy, precision, recall, F1-score, and ROC-AUC for our dataset. The Convolutional Neural Network (CNN) model achieved an accuracy of 94%, and its performance improved further when combined with Long Short-Term Memory (LSTM), resulting in an accuracy of 95%. These results demonstrate the potential of DL models, particularly FNN, in handling tabular datasets effectively.



The ROC-AUC values for the models are very close to each other, indicating that all models exhibit strong and comparable performance in distinguishing between the classes. This suggests that, while some models may have slight advantages, the overall ability to classify the data effectively is consistent across the board.

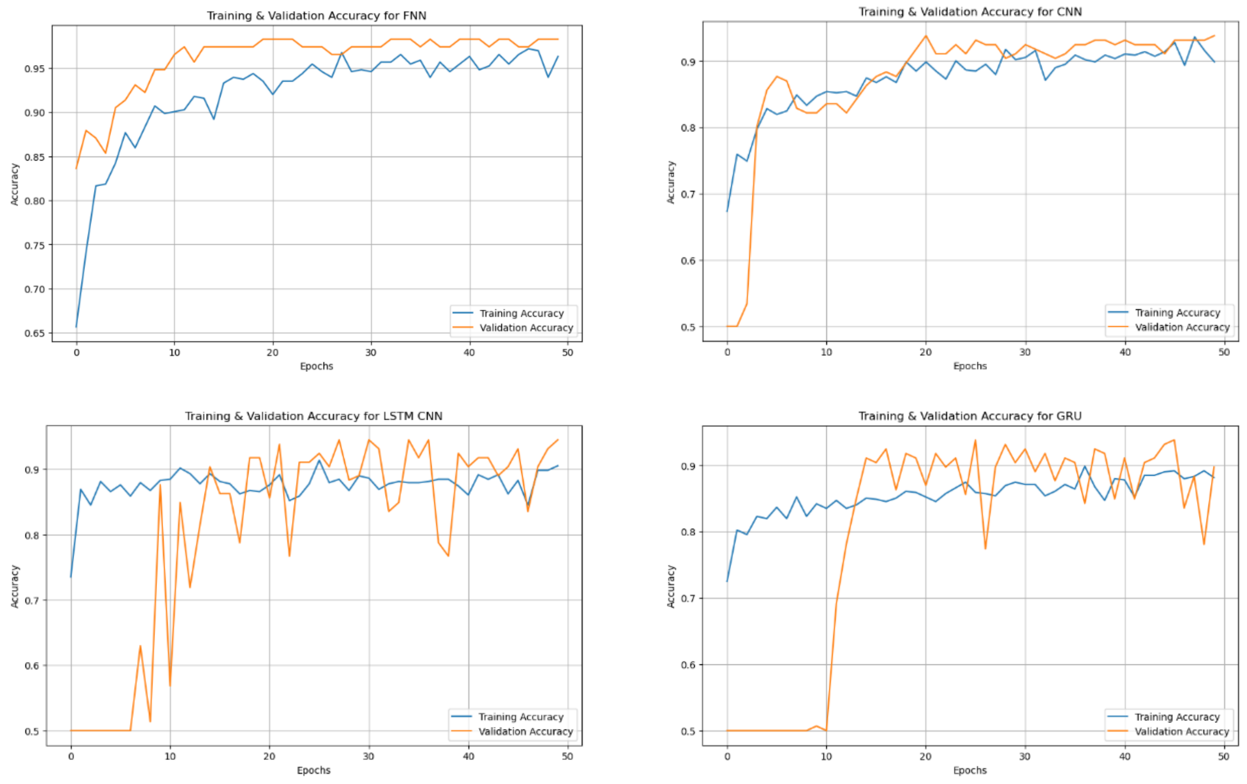
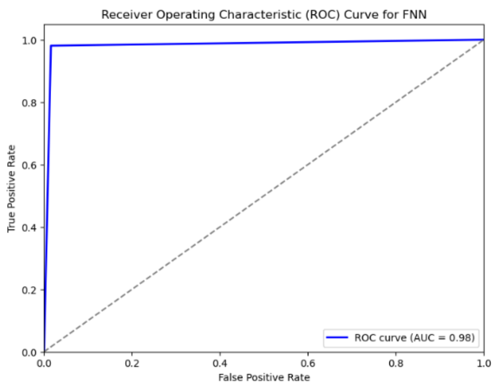
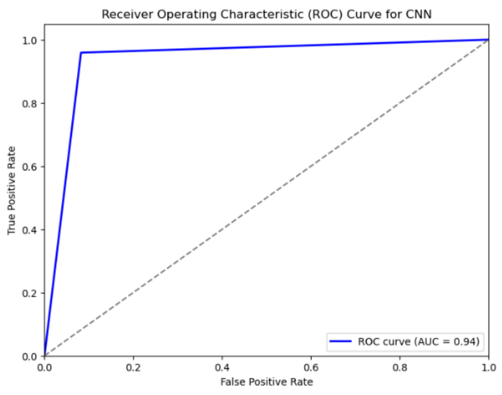
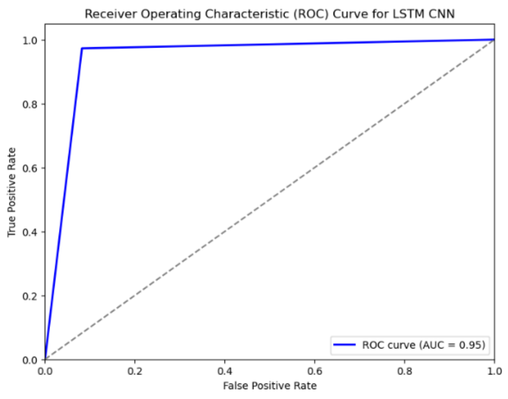


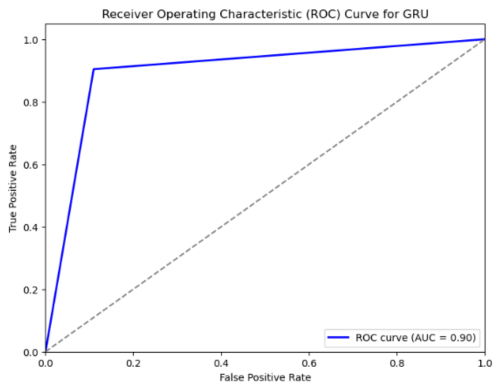
Fig. Training and Validation Accuracy Curve obtained from DL model

For each model, training was conducted over 50 epochs, and the plots of training and validation accuracy reveal distinct trends across the different architectures. The Feedforward Neural Network(FNN) demonstrated the most stable and consistent performance, with both training and validation accuracy remaining high and steady throughout all epochs, indicating excellent generalization without overfitting. The Convolutional Neural Network(CNN) also performed well, with training accuracy steadily increasing to close to 90% by the final epochs, and validation accuracy following a similar trend, maintaining a balanced performance with minimal fluctuations, suggesting robust generalization. In contrast, the Long Short-Term Memory Convolutional Neural Network (LSTM-CNN) showed steady improvement in training accuracy but exhibited significant fluctuations in validation accuracy, with noticeable ups and downs, indicating potential challenges in generalization or overfitting, likely due to the complexity of the combined architecture. Similarly, the Gated Recurrent Unit(GRU) achieved high training accuracy by the final epochs, but its validation accuracy displayed considerable variability, with frequent fluctuations, suggesting that the GRU model may struggle with consistent generalization, potentially due to its recurrent nature and sensitivity to hyperparameters. Overall, FNN and CNN demonstrated the most stable and reliable performance, while LSTM-CNN and GRU showed more variability, highlighting the need for further tuning or regularization to improve their generalization capabilities.









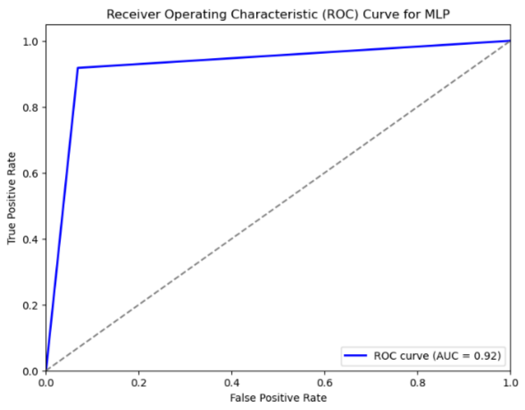
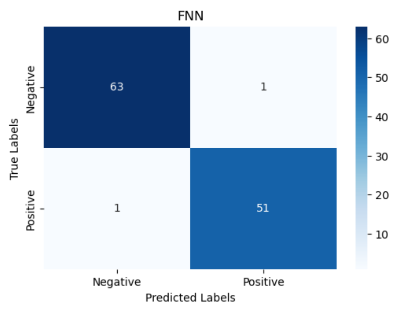
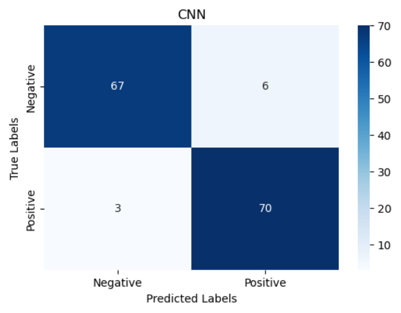
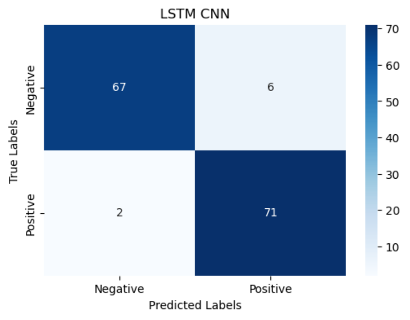


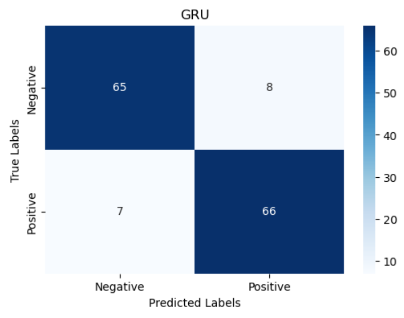
Fig. ROC-AUC Curve obtained from DL Models

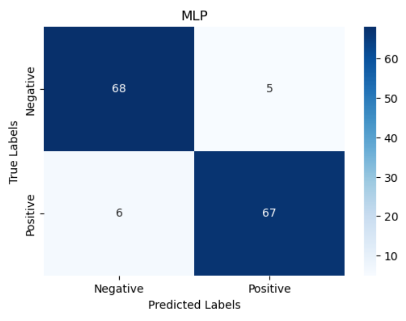
Confusion Matrix:











Best parameters after Hyperparameters tuning:

Model Name	HyperParameter	Accuracy
FNN	model__dropout_1: 0.2, model__dropout_2: 0.2, model__dropout_3: 0.2, model__dropout_4: 0.2	98.48%
CNN	model__neurons: 64, model__filters: 64, model__dropout_rate: 0.4	93.84%
GRU	units_1: 96, dropout_1: 0.2, units_2: 128, dropout_2: 0.4, units_3: 48, dropout_3: 0.5, units_4: 16, dropout_4 : 0.2	92%
MLP	activation: tanh, hidden_layer_sizes: (64, 32), solver: adam	92.48%

5.3 CHALLENGES AND ERROR ANALYSIS

Despite the strong performance of Random Forest (RF), Logistic Regression (LR), Support Vector Machine (SVM), and Deep Learning (DL) models, the Decision Tree (DT) and K-Nearest Neighbors (KNN) models underperformed significantly. Their scores varied considerably across each fold in the cross-validation method, and even after hyperparameter tuning and split validation, their performance did not improve substantially, suggesting that DT and KNN may not be well-suited for this particular dataset. While DL models generally outperformed traditional ML models, the LSTM-CNN and GRU architectures exhibited significant fluctuations in validation accuracy, raising concerns about their ability to generalize effectively. Additionally, these models are computationally expensive and complex, requiring substantial resources for training. The dataset used in this study may not have been large enough to allow these models to identify meaningful patterns, as they typically perform better with larger datasets.

Several factors may have contributed to the observed inaccuracies, including dataset imbalance, which was partially addressed using SMOTE (Synthetic Minority Over-sampling Technique), but a naturally balanced dataset is always preferable for achieving robust model performance. Another potential reason is the limited exploration of feature selection methods, as we only used Recursive Feature Elimination (RFE) and Mutual Information (MI). Other techniques, such as Principal Component Analysis (PCA), Chi-Square Test, LASSO Regression, or Embedded Methods (e.g., feature importance from tree-based models), might yield more informative features and improve model performance. Furthermore, overly complex models like LSTM-CNN and GRU often struggle with small tabular datasets, as they require large amounts of data to capture intricate patterns effectively. Simpler models like RF and FNN, which are better suited for tabular data, consistently outperformed these architectures.

6. DISCUSSION AND INSIGHTS

6.1 CRITICAL EVALUATION

Among the models evaluated, Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Feedforward Neural Network (FNN), Convolutional Neural Network (CNN), and Multilayer Perceptron (MLP) demonstrated superior performance, exhibiting consistency and robustness in handling tabular data effectively and efficiently. These models achieved high accuracy and reliability, making them strong candidates for practical applications. On the other hand, the underperformance of Decision Tree (DT), K-Nearest Neighbors (KNN), LSTM-CNN, and Gated Recurrent Unit (GRU) models was unexpected. While these models did not perform as well in this study, they present opportunities for further investigation and improvement in future research.

6.2 PRACTICAL APPLICATIONS

The high accuracy and robustness of Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Feedforward Neural Network (FNN), Convolutional Neural Network (CNN), and Multilayer Perceptron (MLP) make them particularly suitable for the automated diagnosis of Polycystic Ovary Syndrome (PCOS). These models can be deployed in healthcare settings, such as hospitals, to streamline and expedite the diagnosis process, replacing traditional and time-consuming methods. Additionally, the effectiveness of these models can be tested on other datasets to evaluate their generalizability and competency in classifying other diseases using tabular data. This would extend the scope of our research and provide valuable insights into the applicability of these models across different medical domains.

The findings of this study contribute to academic research by offering a detailed comparison of the effectiveness of various machine learning and deep learning models for tabular datasets. They also highlight the potential of robust models like Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Feedforward Neural Network (FNN), Convolutional Neural Network (CNN), and Multilayer Perceptron (MLP) in advancing automated diagnostic systems, paving the way for future innovations in healthcare and disease detection.

7. FUTURE WORK AND IMPROVEMENTS

7.1 POSSIBLE ENHANCEMENTS

A high-quality dataset is essential for building a robust and generalizable model. In this study, the dataset consisted of 541 entries and contained some missing values, which limited the model's performance and generalization capabilities. To enhance the project's outcomes, a larger and more balanced dataset is required. Such a dataset would improve model consistency and accuracy. Additionally, further tuning of hyperparameters and advanced feature engineering techniques could significantly boost model performance. Incorporating regularization methods like Dropout and L2 regularization in deep learning models can help reduce overfitting and improve generalization. Furthermore, exploring ensemble techniques by combining multiple models, such as Random Forest (RF), Feedforward Neural Network (FNN), and Convolutional Neural Network (CNN), could lead to better performance and open new avenues for future research.

7.2 SCALABILITY AND DEPLOYMENT

In our experiments, Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Feedforward Neural Network (FNN), Convolutional Neural Network (CNN), and Multilayer Perceptron (MLP) demonstrated consistent and reliable performance. Among these, RF, LR, and FNN stand out due to their low computational cost, making them suitable for real-world deployment. These models can be scaled for large-scale applications using cloud-based solutions or edge computing. Integrating these models into user-friendly applications or APIs would facilitate seamless adoption in healthcare settings, enabling automated and efficient diagnosis of Polycystic Ovary Syndrome (PCOS).

7.3 POTENTIAL RESEARCH DIRECTIONS

The pursuit of knowledge and exploration is endless, and there is always room for improvement. Advanced architectures, such as Transformers or Graph Neural Networks (GNNs), could be explored for tabular data to achieve better performance and insights. Explainability techniques like SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), along with statistical approaches such as paired T-tests and ANOVA (Analysis of Variance), can be employed to interpret model predictions and enhance transparency, especially in critical applications like healthcare. Transfer learning techniques could also be applied to leverage pre-trained models or datasets from similar domains, potentially improving the performance of deep learning models. Finally, developing real-time prediction systems using lightweight or optimized architectures would enable deployment in resource-constrained environments, further expanding the practical applicability of these models.

8. ETHICAL CONSIDERATIONS AND SUSTAINABILITY

8.1 ETHICAL ISSUES

In the development and deployment of machine learning (ML) and deep learning (DL) models for healthcare applications, particularly for diagnosing Polycystic Ovary Syndrome (PCOS), several ethical considerations must be addressed:

1. **Data Privacy and Security**: The dataset used in this study contains sensitive health information from patients. Proper anonymization techniques should be applied to protect patient identities, and data should be stored and processed in compliance with regulations.
2. **Informed Consent**: Patients whose data is used for training and testing these models should be informed about how their data will be used and should provide explicit consent.
3. **Accountability**: In the event of a misdiagnosis using the models, it is important to establish clear accountability. Healthcare providers and developers of these models should have protocols in place to address errors and ensure that patients are not harmed by incorrect predictions.

8.2 SUSTAINABILITY

1. **Environmental Impact**: Training deep learning models can be computationally expensive and energy-intensive. To mitigate the environmental impact, it is important to optimize model architectures and use energy-efficient hardware. The deployment of these models should be cost-effective to ensure that they can be widely adopted, especially in resource-constrained settings
2. **Long-term Maintenance**: For these models to remain effective, they must be regularly updated with new data and retrained to account for changes in medical knowledge and patient demographics.
3. **Continuous Monitoring and Evaluation**: After deployment, the models should be continuously monitored to ensure that they are performing as expected.

9. CONCLUSION

This study conducted a comprehensive comparative analysis of various machine learning (ML) and deep learning (DL) models for the detection of Polycystic Ovary Syndrome (PCOS). The primary objective of our study was to identify the most effective models capable of achieving high performance metrics while addressing challenges such as class imbalance, feature selection and model generalization.

The preprocessing steps, including median imputation , MinMax normalization , and Synthetic Minority Oversampling Technique (SMOTE) , enhanced model performance by addressing missing values, scaling features, and mitigating class imbalance. Feature selection techniques like Recursive Feature Elimination (RFE) and SelectKBest were used to refine the dataset by identifying the most relevant features.

The results demonstrated that Random Forest (RF) emerged as the top-performing ML model, achieving an accuracy of 97.26% using split validation outperforming other traditional ML models, including Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), and K-Nearest Neighbors (KNN), across all performance metrics. DT and KNN exhibited significant limitations, showing inconsistent performance even after hyperparameter tuning and split validation. This suggests their unsuitability for this specific dataset and the proneness to overfit.

Feedforward Neural Networks (FNN) achieved the highest accuracy of 98.48% , surpassing other DL architectures such as 1D-Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU), and Multilayer Perceptrons (MLP). CNNs and LSTM-CNN hybrids also performed well, with accuracies of 93.84% and 95%, respectively. However, GRU and LSTM-CNN architectures exhibited erratic validation accuracy, highlighting challenges in achieving consistent generalization for complex recurrent models when applied to tabular data.

Despite the strong overall performance of RF, LR, SVM, FNN, CNN and MLP some challenges were identified. Models like DT, KNN, LSTM-CNN, and GRU underperformed due to factors such as dataset size limitations, sensitivity to hyperparameters, and difficulties in capturing intricate patterns in tabular data. Additionally, the computationally expensive nature of certain DL models underscores the need for larger datasets and advanced regularization techniques to improve their applicability.

This research highlights the transformative potential of AI-driven diagnostic systems in revolutionizing PCOS detection. By leveraging robust models like RF and FNN, healthcare providers can achieve faster, more accurate, and reliable diagnoses, reducing the burden of traditional diagnostic methods. Future work will focus on expanding the dataset, refining feature engineering strategies, and exploring ensemble techniques and advanced DL architectures to further enhance model efficacy and adaptability.

10. REFERENCES

[1] R. Azziz *et al.*, "Polycystic ovary syndrome," Aug. 11, 2016, *Nature Publishing Group*. doi: 10.1038/nrdp.2016.57.

[2] F. Giallauria, F. Orio, S. Palomba, G. Lombardi, A. Colao, and C. Vigorito, "Cardiovascular risk in women with polycystic ovary syndrome," Oct. 2008. doi: 10.2459/JCM.0b013e32830b58d4.

[3] E. Diamanti-Kandarakis and A. Dunaif, "Insulin resistance and the polycystic ovary syndrome revisited: An update on mechanisms and implications," *Endocr Rev*, vol. 33, no. 6, pp. 981–1030, Dec. 2012, doi: 10.1210/er.2011-1034.

[4] G. Bozdag, S. Mumusoglu, D. Zengin, E. Karabulut, and B. O. Yildiz, "The prevalence and phenotypic features of polycystic ovary syndrome: A systematic review and meta-Analysis," Dec. 01, 2016, *Oxford University Press*. doi: 10.1093/humrep/dew218.

[5] A. E. Joham *et al.*, "Polycystic ovary syndrome," *Lancet Diabetes Endocrinol*, vol. 10, no. 9, pp. 668–680, Sep. 2022, doi: 10.1016/S2213-8587(22)00163-2.

[6] R. S. Legro *et al.*, "Diagnosis and treatment of polycystic ovary syndrome: An endocrine society clinical practice guideline," *Journal of Clinical Endocrinology and Metabolism*, vol. 98, no. 12, pp. 4565–4592, 2013, doi: 10.1210/jc.2013-2350.

[7] S. Frøssing, M. C. Nylander, E. Chabanova, C. Kistorp, S. O. Skouby, and J. Faber, "Quantification of visceral adipose tissue in polycystic ovary syndrome: dual-energy X-ray absorptiometry versus magnetic resonance imaging," *Acta radiol*, vol. 59, no. 1, pp. 13–17, Jan. 2018, doi: 10.1177/0284185117711475.

[8] V. Thakre, “PCOcare: PCOS Detection and Prediction using Machine Learning Algorithms,” *Biosci Biotechnol Res Commun*, vol. 13, no. 14, pp. 240–244, Dec. 2020, doi: 10.21786/bbrc/13.14/56.

[9] B. Purnama, U. N. Wisesti, Adiwijaya, F. Nhita, A. Gayatri, and T. Mutiah, “A classification of polycystic Ovary Syndrome based on follicle detection of ultrasound images,” in *2015 3rd International Conference on Information and Communication Technology (ICoICT)*, IEEE, May 2015, pp. 396–401. doi: 10.1109/ICoICT.2015.7231458.

[10] P. Dutta, S. Paul, and M. Majumder, “An Efficient SMOTE Based Machine Learning classification for Prediction &amp;amp; Detection of PCOS,” Nov. 08, 2021. doi: 10.21203/rs.3.rs-1043852/v1.

[11] M. Sumathi, P. Chitra, R. Sakthi Prabha, and K. Srilatha, “Study and detection of PCOS related diseases using CNN,” *IOP Conf Ser Mater Sci Eng*, vol. 1070, p. 012062, Feb. 2021, doi: 10.1088/1757-899X/1070/1/012062.

[12] B. Cahyono, Adiwijaya, M. S. Mubarok, and U. N. Wisesty, “An implementation of convolutional neural network on PCO classification based on ultrasound image,” in *2017 5th International Conference on Information and Communication Technology (ICoIC7)*, IEEE, May 2017, pp. 1–4. doi: 10.1109/ICoICT.2017.8074702.

[13] S. Bharati, P. Podder, and M. R. Hossain Mondal, “Diagnosis of Polycystic Ovary Syndrome Using Machine Learning Algorithms,” in *2020 IEEE Region 10 Symposium (TENSYMP)*, IEEE, 2020, pp. 1486–1489. doi: 10.1109/TENSYMP50017.2020.9230932.

[14] 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 *IICETA 2022 - 5th International Conference on Engineering Technology and its Applications*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 532–536. doi: 10.1109/IICETA54559.2022.9888677.

[15] R. Ahmad, L. A. Maghrabi, I. A. Khaja, L. A. Maghrabi, and M. Ahmad, “SMOTE-Based Automated PCOS Prediction Using Lightweight Deep Learning Models,” *Diagnostics*, vol. 14, no. 19, Oct. 2024, doi: 10.3390/diagnostics14192225.

[16] Prasoon Kottarathil, “Polycystic ovary syndrome (PCOS),” Kaggle. Available online: https://www.kaggle.com/prasoonkottarathil/polycystic-ovary-syndrome-pcos%7D%7D