

PCOS AI: Intelligent risk assessment diagnosis and assistance

Group Members: Nandhika Rajmanikandan, Kavya Pachchava

Group Number: 31

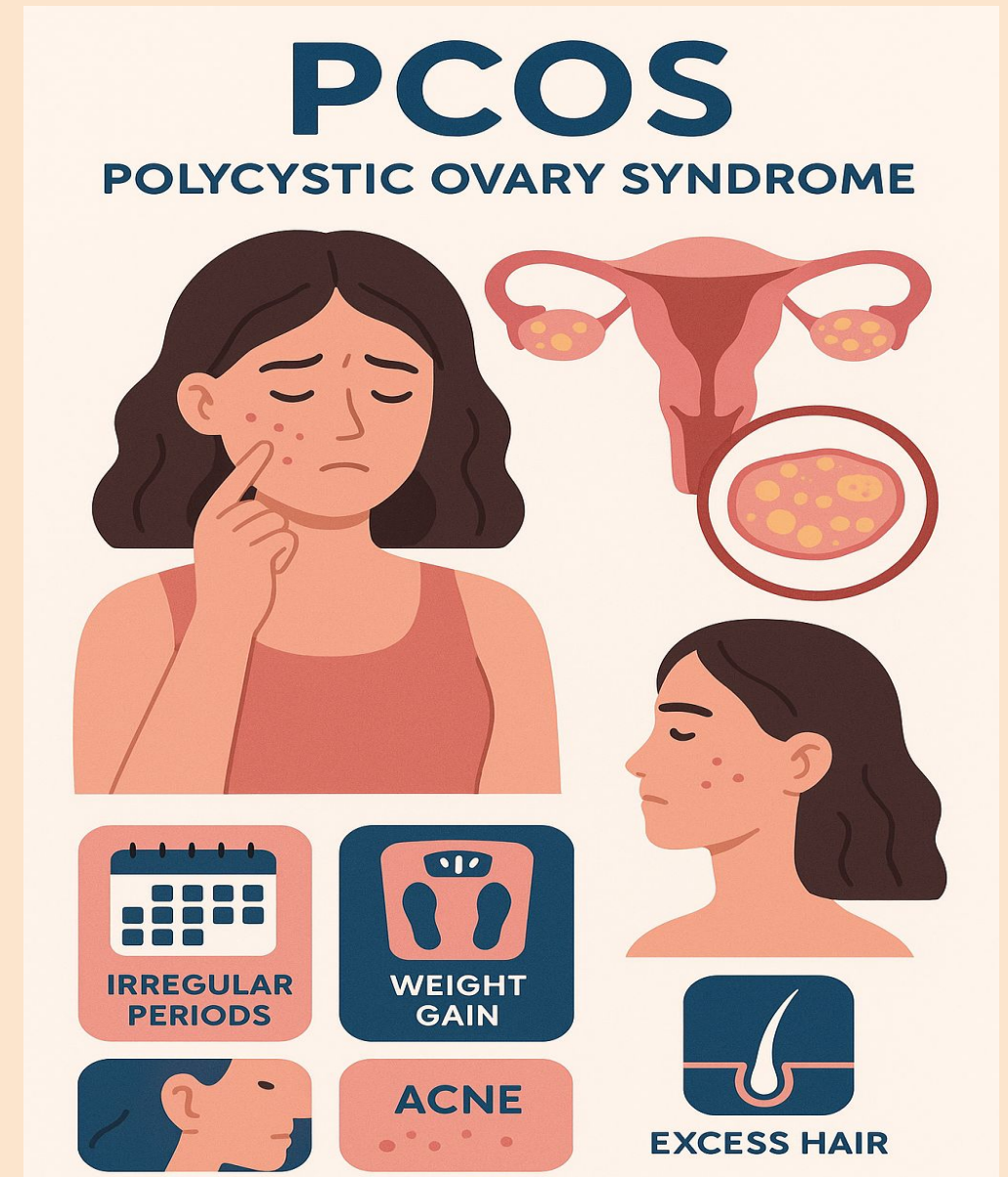
GitHub links: [OvaryTales- Kavya Pachchava](#)

[OvaryTales- Nandhika raj](#)

Youtube link: [Final Project Presentation](#)

Content

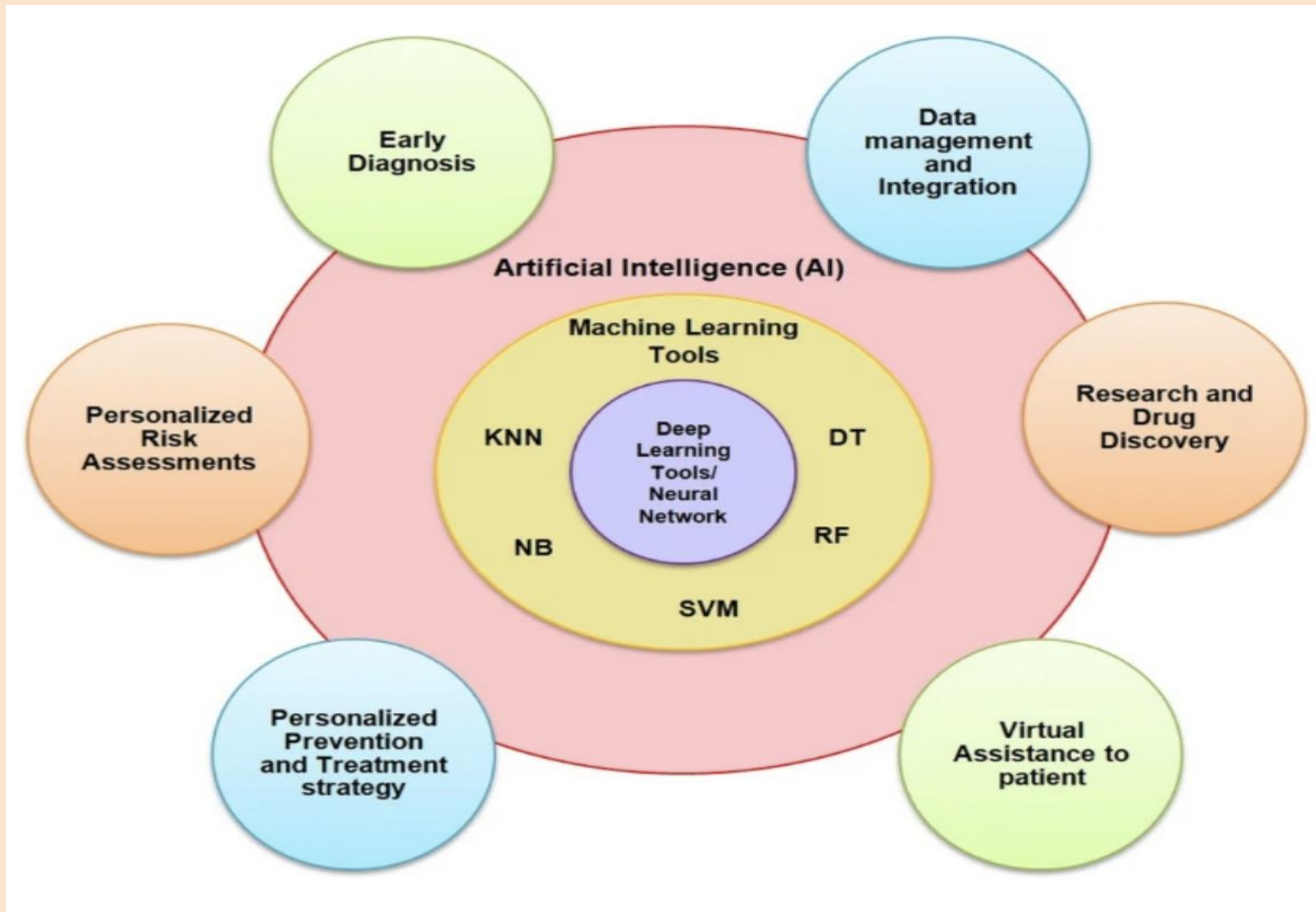
- Introduction
- Data & preprocessing
- Correlation & its interpretation
- Exploratory data analysis
- Model building
 1. Random forest classifier
 2. Neural network
 3. SVM with ADASYN
 4. EBM
- Conclusion



Introduction

- Polycystic Ovary Syndrome (PCOS) is the most common endocrinopathy in reproductive aged women, with an estimated prevalence ranging from 4% to 20%.
- PCOS is associated with increased incidence of cardiovascular disease, infertility, and of endometrial cancer.
- Diagnosing PCOS can be complicated due to its diverse symptoms and similarities with other endocrine disorders.
- However, advancements in machine learning are proving beneficial in analyzing complex biomedical data, which can enhance the identification of diagnostic biomarkers.
- Objective: Build a reliable machine learning model that supports early diagnosis and reduces misdiagnosis using patient data.





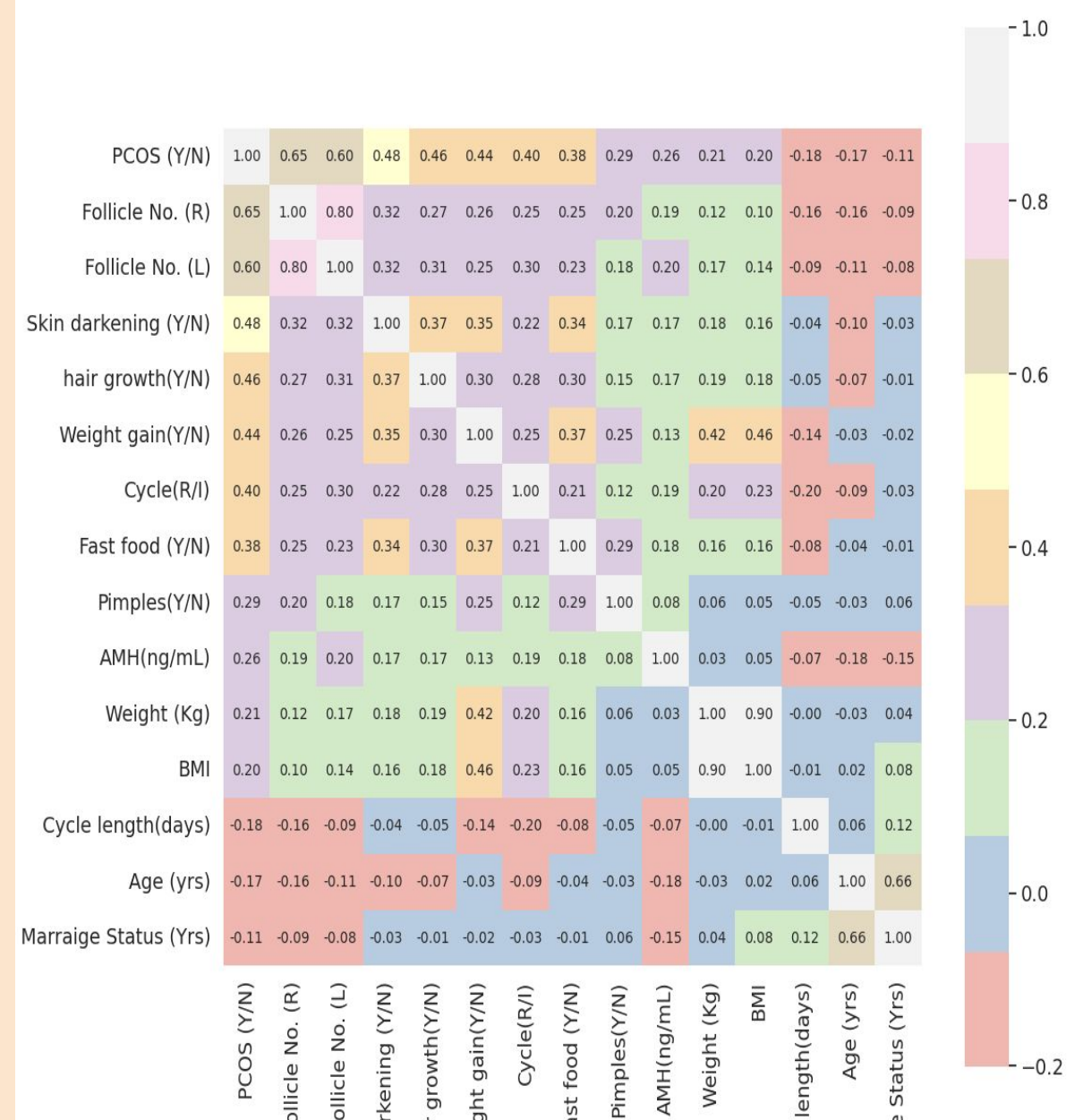
Data & Preprocessing

- The dataset combines clinical, biochemical, and lifestyle features such as hormone levels, follicle counts, BMI, diet, and menstrual data.
- Data cleaning involved converting incorrectly typed numerical values stored as strings, removing extra spaces in column headers, and renaming for consistency.
- Missing values were addressed using median imputation to maintain robustness against outliers.



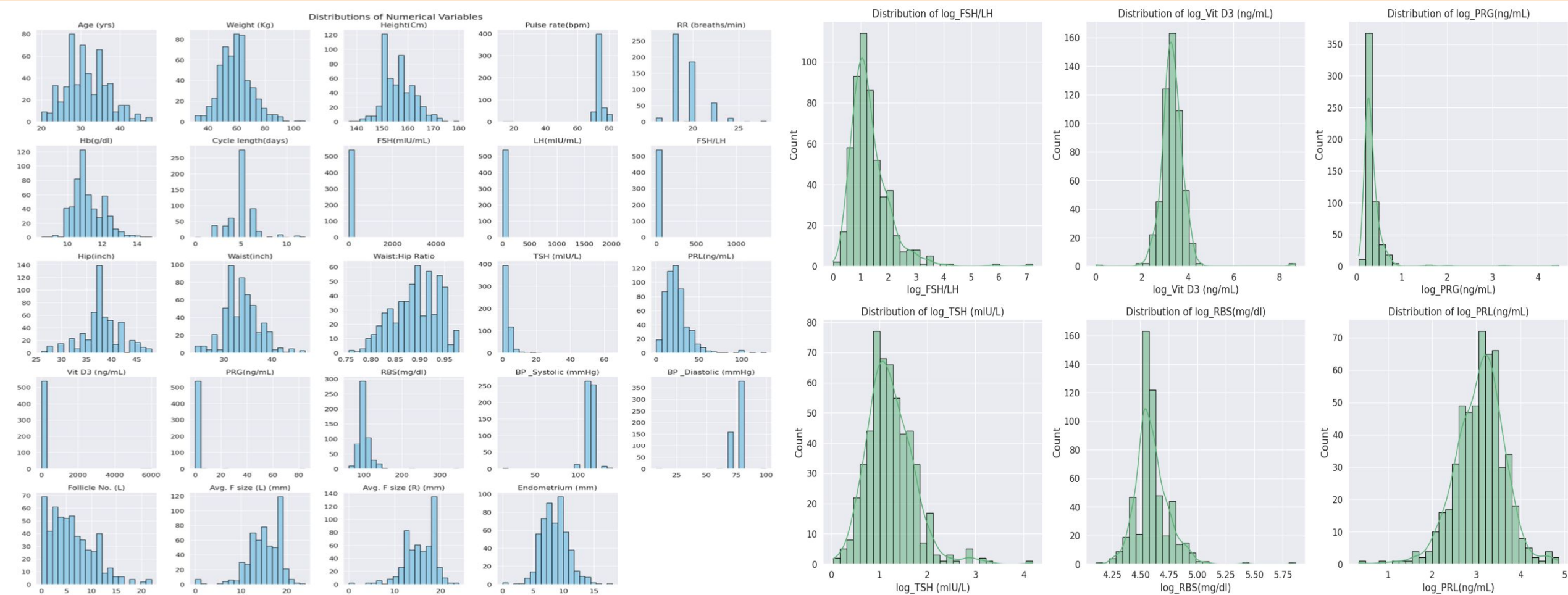
Correlation and its interpretation

- Correlation matrix helped identify strong predictors of PCOS, with Follicle Count (Left and Right) showing highest correlation (> 0.6).
- Heatmap visualization revealed multicollinearity, especially between Right and Left Follicle Count, and between Weight and BMI.
- As a result, redundant features like Right Follicle Number and BMI were removed to enhance model clarity and performance.



Exploratory data analysis

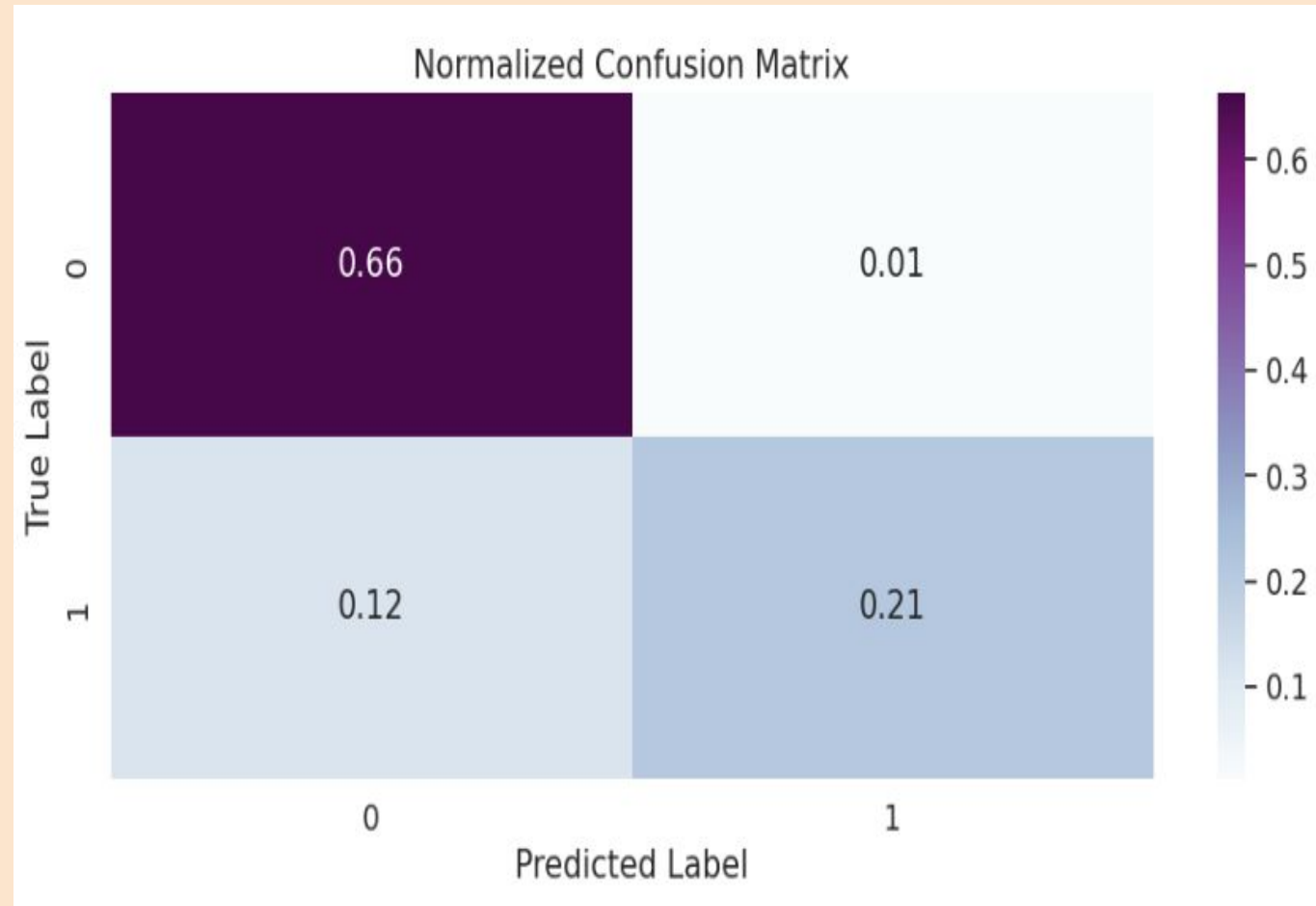
EDA revealed skewed clinical features that were log-transformed, and highlighted patterns such as higher weight and longer cycle lengths in PCOS-positive women, while lifestyle habits showed minimal distinction between groups.



Model building:

1. Random Forest classifier:

- Random Forest was selected for its ability to handle complex, non-linear data and provide feature importance insights.
- It gave 87% accuracy but missed many PCOS cases (64% recall).
- Good start, but not enough for medical use.



Random Forest - Hyperparameter tuning

- We used GridSearchCV to fine-tune the Random Forest model. Key parameters included max_depth, n_estimators, and the splitting criterion.
- The best results were achieved with a max_depth of 8 and 150 estimators. A lower depth helped reduce overfitting while maintaining accuracy.
- This tuning improved the model's ability to generalize, but recall was still limited, suggesting the need for further balancing or alternative models.
- GridSearchCV tuned parameters: max_depth=8, n_estimators=150.

```
# Define base model and hyperparameter grid for GridSearch
rfc = RandomForestClassifier(random_state=42)
param_grid = {
    'n_estimators': [100, 150, 200, 500, 700],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth': [4, 5, 6, 7, 8, 9, 10, 12],
    'criterion': ['gini', 'entropy']
}

# Grid search with cross-validation
grid_search = GridSearchCV(estimator=rfc, param_grid=param_grid, cv=7, n_jobs=-1, verbose=1)
grid_search.fit(X_train, y_train)

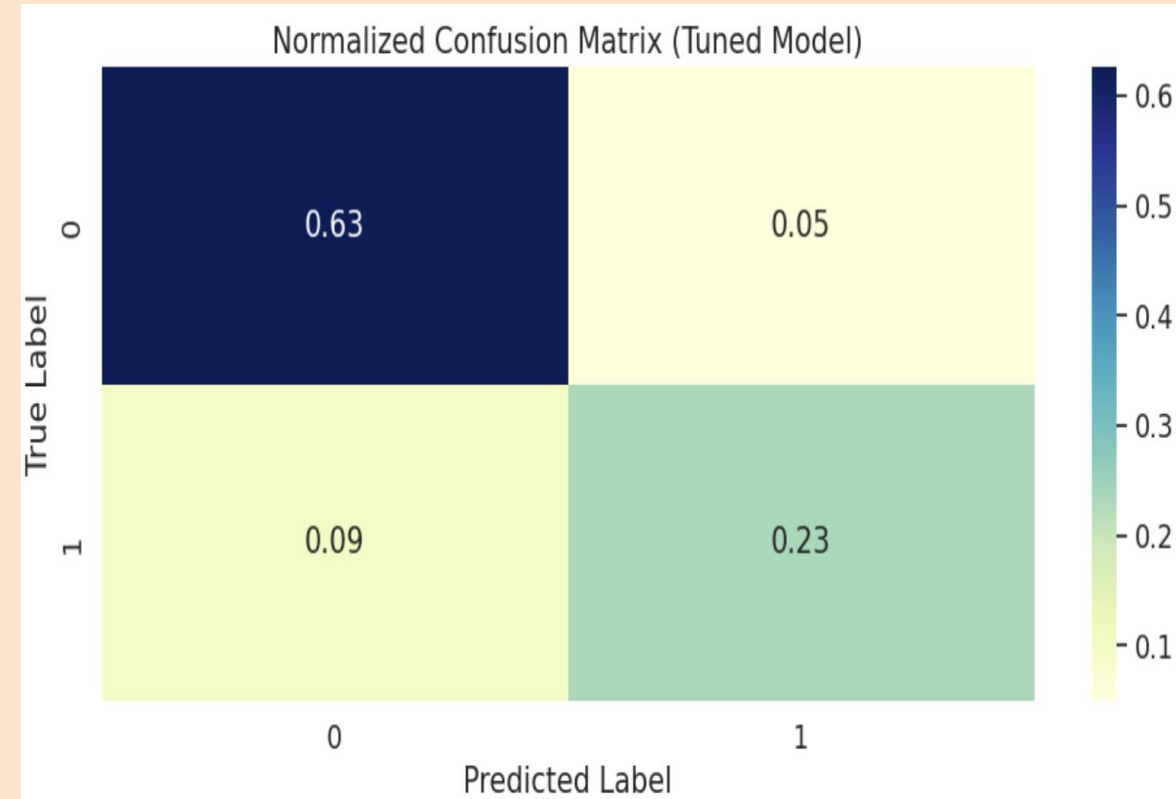
# Train final model with best params
rfc_best = RandomForestClassifier(**grid_search.best_params_, random_state=42)
rfc_best.fit(X_train, y_train)
```



Model Building

2. Neural network:

- A feedforward neural network was trained using KerasTuner with hyperparameter optimization and a standard scaling and one-hot encoding were applied.
- Final model achieved 86% accuracy and 72% recall which is better than random forest, reducing false negatives.



Neural Network - Hyperparameter Tuning

- We used KerasTuner's RandomSearch to explore combinations of layers, units, activation functions, and optimizers.
- The tuning process helped find an architecture that balanced depth and complexity, resulting in improved recall while maintaining good overall accuracy.
- Parameters like dropout and learning rate were also adjusted to prevent overfitting and ensure model stability.
- Standard scaling and one-hot encoding were applied.

```
# Build model function for tuner
def build_model(hp):
    model = Sequential()
    model.add(Dense(hp.Int('units_1', min_value=32, max_value=128, step=16),
                    activation=hp.Choice('act_1', ['relu', 'tanh']),
                    input_shape=(X_train_scaled.shape[1],)))

    if hp.Boolean('second_layer'):
        model.add(Dense(hp.Int('units_2', min_value=32, max_value=128, step=16),
                        activation=hp.Choice('act_2', ['relu', 'tanh'])))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(
        optimizer=hp.Choice('optimizer', ['adam', 'rmsprop']),
        loss='binary_crossentropy',
        metrics=['accuracy']
    )
    return model

# Hyperparameter tuner
tuner = kt.RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=10,
    executions_per_trial=1,
    directory='my_dir',
    project_name='pcos_nn_tuning'
)

# Run search
tuner.search(X_train_scaled, y_train, epochs=20, validation_split=0.2, verbose=1)

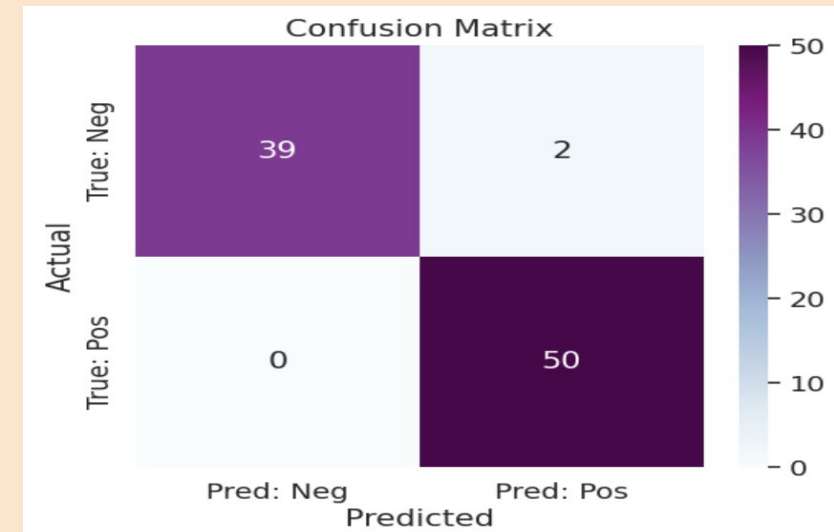
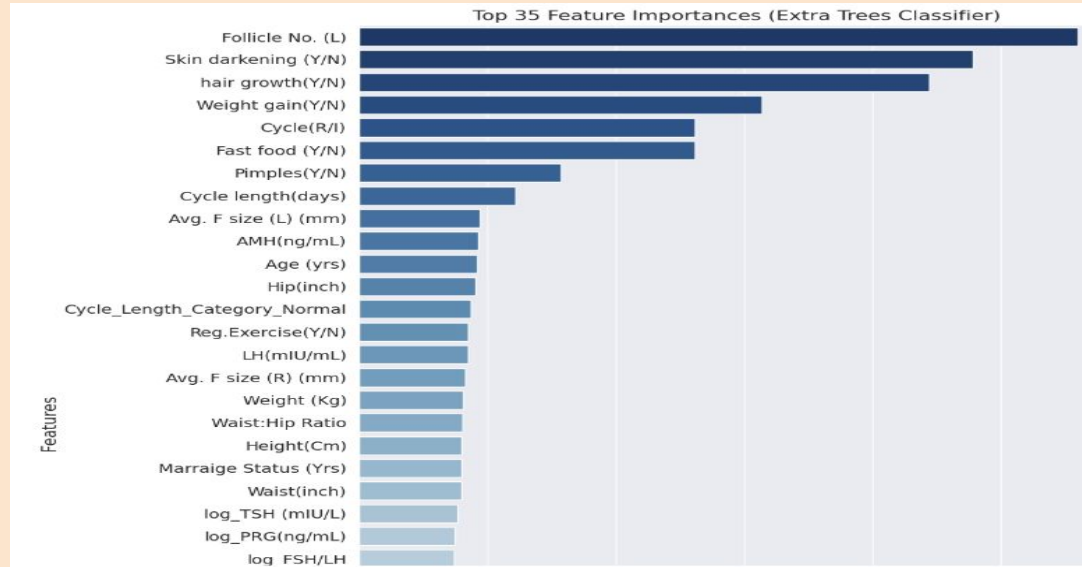
# Get best model
best_model = tuner.get_best_models(num_models=1)[0]
best_params = tuner.get_best_hyperparameters(1)[0].values
print("Best Hyperparameters:", best_params)
```



Model building

3. SVM with ADASYN

- We used the Extra Trees Classifier to select the top 9 most relevant features for the model.
- To address class imbalance, ADASYN (adaptive-synthetic sampling) was used to generate synthetic PCOS-positive samples, while ENN helped remove noisy examples from the majority class.
- A SVM with an RBF kernel was trained and fine-tuned, achieving 97.8% accuracy, 100% recall making it the best-performing model with no missed PCOS cases.



SVM - Hyperparameter Tuning

- We used GridSearchCV to fine-tune the SVM's kernel, C (regularization strength), and gamma (kernel coefficient).
- The best configuration was found with an RBF kernel, moderate C, and tuned gamma, which allowed the model to create flexible decision boundaries.
- These adjustments, combined with ADASYN for class balancing, significantly enhanced the model's recall and precision.
- To address class imbalance, ADASYN was used to generate synthetic PCOS-positive samples, while ENN helped remove noisy examples from the majority class.

```
# ✅ Step 8: SVM Hyperparameter tuning using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10, 100, 1000],
    'gamma': ['scale', 'auto'],
    'kernel': ['rbf', 'linear']
}
grid = GridSearchCV(SVC(probability=True), param_grid, refit=True, cv=5, verbose=1, scoring='recall')
grid.fit(X_train, np.ravel(y_train))

# ✅ Step 9: Best model
best_svm = grid.best_estimator_
print("Best SVM Parameters:", grid.best_params_)
```



Explainable boosting machine (EBM)

- The Explainable Boosting Machine (EBM) is a model designed to be interpretable, using additive models with interaction terms, making it suitable for clinical applications.
- This model achieved strong results with 95.6% accuracy, and both precision and recall at 96%, producing a balanced and reliable performance.
- This model is Good for real-world clinics where doctors need to understand the model.



Conclusion

- Among the models tested, the Support Vector Machine (SVM) with ADASYN achieved the highest accuracy and recall, while the Explainable Boosting Machine (EBM) also performed well with excellent interpretability; in comparison, Random Forest and Neural Network produced decent results but were weaker in sensitivity and transparency.
- In summary, we recommend using SVM for automated detection and EBM for applications requiring clinician trust. Future work could involve testing on larger datasets, deployment in real-world settings, and gathering feedback from healthcare professionals.

Model Comparison Table (Performance Metrics)					
Model	Random Forest	87.1	94.0	64.0	76.0
	Neural Network (Tuned)	86.0	83.0	72.0	77.0
	SVM (with ADASYN+ENN)	97.8	96.1	100.0	98.0
	Explainable Boosting Machine (EBM)	95.6	96.0	96.0	96.0
		Accuracy	Precision	Recall	F1-Score



References

DATASET:

1. [PCOS-Dataset](#)
2. [PCOS- Detection using ultrasound images](#)
3. Chen, W., Miao, J., Chen, J., & Chen, J. (2025). Development of machine learning models for diagnostic biomarker identification and immune cell infiltration analysis in PCOS. *Journal of Ovarian Research*, 18(1), 1.
4. Palomba, S., Seminara, G., Costanzi, F., Caserta, D., & Aversa, A. (2024). Chemerin and Polycystic Ovary Syndrome: A Comprehensive Review of Its Role as a Biomarker and Therapeutic Target. *Biomedicines*, 12(12), 2859. ([DOI](#))
5. Wang, L., Zhang, Y., Ji, F., Si, Z., Liu, C., Wu, X., ... & Chang, H. (2025). Identification of crucial genes for polycystic ovary syndrome and atherosclerosis through comprehensive bioinformatics analysis and machine learning. *International Journal of Gynecology & Obstetrics*.

