



**TITLE: AI-BASED DISEASE PREDICTION USING BLOOD  
SAMPLES.**

**GROUP MEMBER:**

AIZA SAQIB	70147247
MAILA SALEEM	70146709
AYESHA TARIQ	70151037
TANZEEL AHMAD	70147283

**SUBMITTED TO: SIR SYED HAMEDON**

## DESCRIPTION:

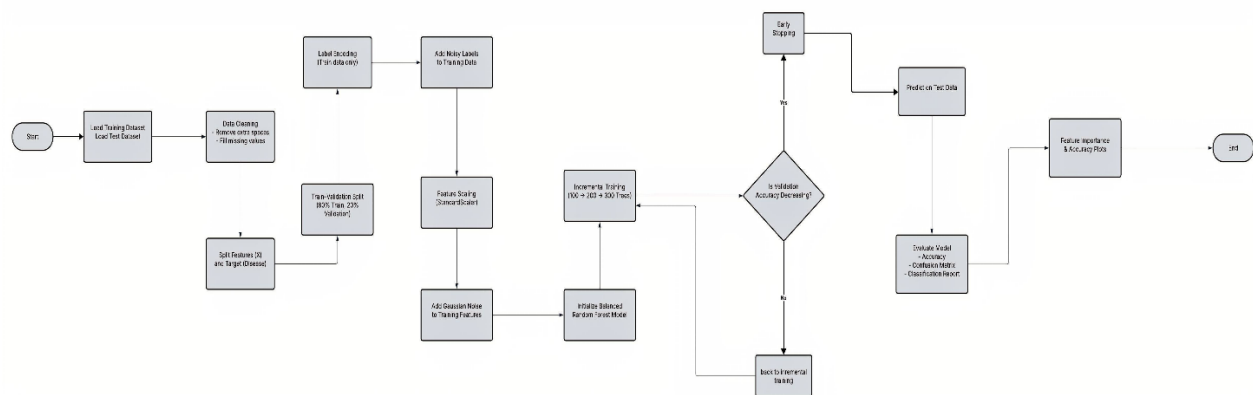
This project focuses on building an AI-based disease prediction system using patient's blood test data. The goal is to predict possible diseases accurately and fairly, even when some diseases appear less frequently in the dataset. To achieve this, a Balanced Random Forest model is used. Unlike traditional machine-learning models that tend to favor common diseases, this model treats all diseases equally, improving predictions for rare and critical conditions. The system is carefully designed to avoid data leakage, control overfitting, and evaluate performance on completely unseen test data, ensuring that it behaves reliably in real-world medical scenarios.

Beyond just the AI model, the project includes a fully functional backend and frontend, making the system interactive and user-friendly. Patients can input blood test values through the frontend, which communicates with the backend to process the data, run predictions, and return results in real time.

The final system not only predicts diseases but also:

- Shows model accuracy
- Displays a confusion matrix

## FLOW DIAGRAM:



## CODE:

```
""" -----
train_df = pd.read_csv("Blood_sample_dataset_balanced.csv").copy()
test_df  = pd.read_csv("blood_samples_dataset_test.csv").copy()

train_df["Disease"] = train_df["Disease"].str.strip()
test_df["Disease"]  = test_df["Disease"].str.strip()

for df in (train_df, test_df):
    for col in df.columns:
        if df[col].dtype == "object":
            df[col] = df[col].fillna(df[col].mode()[0])
        else:
            df[col] = df[col].fillna(df[col].mean())

|

label_encoder = LabelEncoder()
y_train_enc = label_encoder.fit_transform(y_train)
y_val_enc   = label_encoder.transform(y_val)
y_test_enc  = label_encoder.transform(y_test)


rf_model = BalancedRandomForestClassifier(
    n_estimators=300,          # more trees for better test accuracy
    max_depth=10,             # deeper trees for test generalization
    min_samples_leaf=8,
    min_samples_split=12,
    max_features="sqrt",
    warm_start=True,
    random_state=42,
    n_jobs=-1
)

train_scores, val_scores = [], []
```

```

cm = confusion_matrix(y_test_enc, test_pred)
plt.figure(figsize=(10, 7))
sns.heatmap(
    cm, annot=True, fmt="d",
    xticklabels=label_encoder.classes_,
    yticklabels=label_encoder.classes_,
    cmap="YlGnBu"
)
plt.title("Confusion Matrix - Balanced Random Forest")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()

feat_df = pd.DataFrame({
    "Feature": X.columns,
    "Importance": rf_model.feature_importances_
}).sort_values(by="Importance", ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(
    data=feat_df.head(10),
    x="Importance",
    y="Feature"
)
plt.title("Top 10 Most Important Clinical Features")
plt.tight_layout()
plt.show()

print("\n 🌲 Total Trees Used:", len(rf_model.estimators_))

print("\n 📊 FINAL MODEL ACCURACY SUMMARY")
print("=" * 50)
print(f"Training Accuracy    : {train_scores[-1] * 100:.2f}%")
print(f"Validation Accuracy   : {val_scores[-1] * 100:.2f}%")
print(f"Test Accuracy         : {test_acc * 100:.2f}%")

```

## OUTPUT:

Trees: 100 | Train Acc: 0.8359 | Val Acc: 1.0000  
Trees: 200 | Train Acc: 0.8359 | Val Acc: 1.0000  
Trees: 300 | Train Acc: 0.8359 | Val Acc: 1.0000

✅ FINAL TEST ACCURACY: 0.4671

🔍 FIRST 10 TEST SAMPLE RESULTS

-----  
Sample 1

Actual : Thalassemia  
Predicted : Anemia  
Result : Wrong  
-----

Sample 2

Actual : Diabetes  
Predicted : Diabetes  
Result : Correct  
-----

Sample 3

Actual : Healthy  
Predicted : Healthy  
Result : Correct  
-----

Sample 4

Actual : Diabetes  
Predicted : Anemia  
Result : Wrong  
-----

Sample 5

Actual : Healthy  
Predicted : Healthy  
Result : Correct  
-----

Sample 6

Actual : Healthy  
Predicted : Diabetes  
Result : Wrong  
-----

-----  
Sample 7

Actual : Diabetes  
Predicted : Diabetes  
Result : Correct  
-----

Sample 8

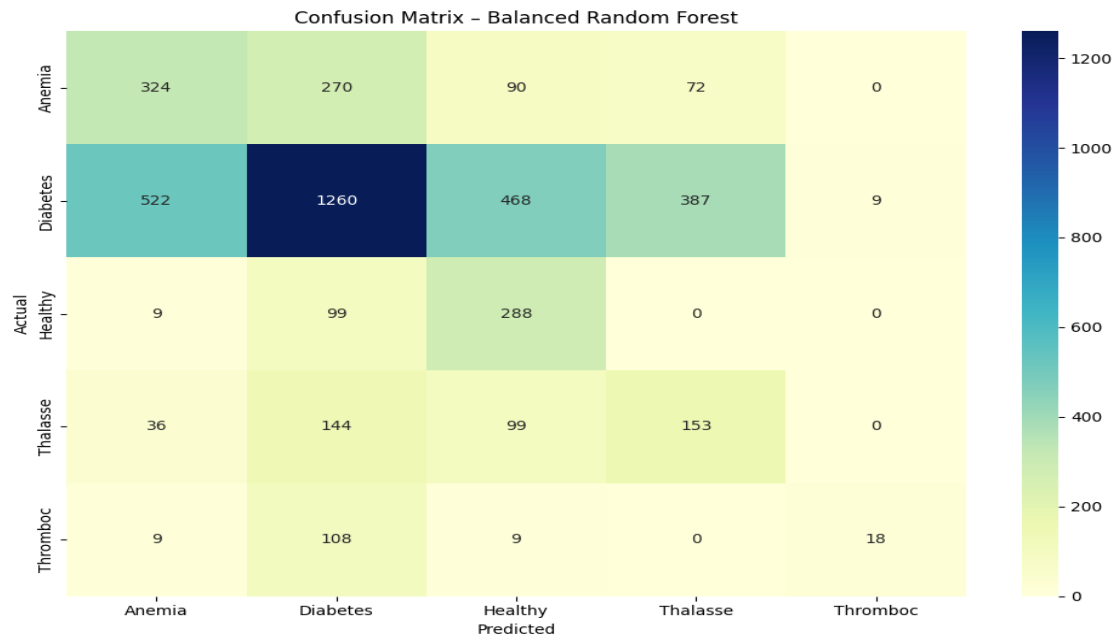
Actual : Diabetes  
Predicted : Healthy  
Result : Wrong  
-----

Sample 9

Actual : Healthy  
Predicted : Diabetes  
Result : Wrong  
-----

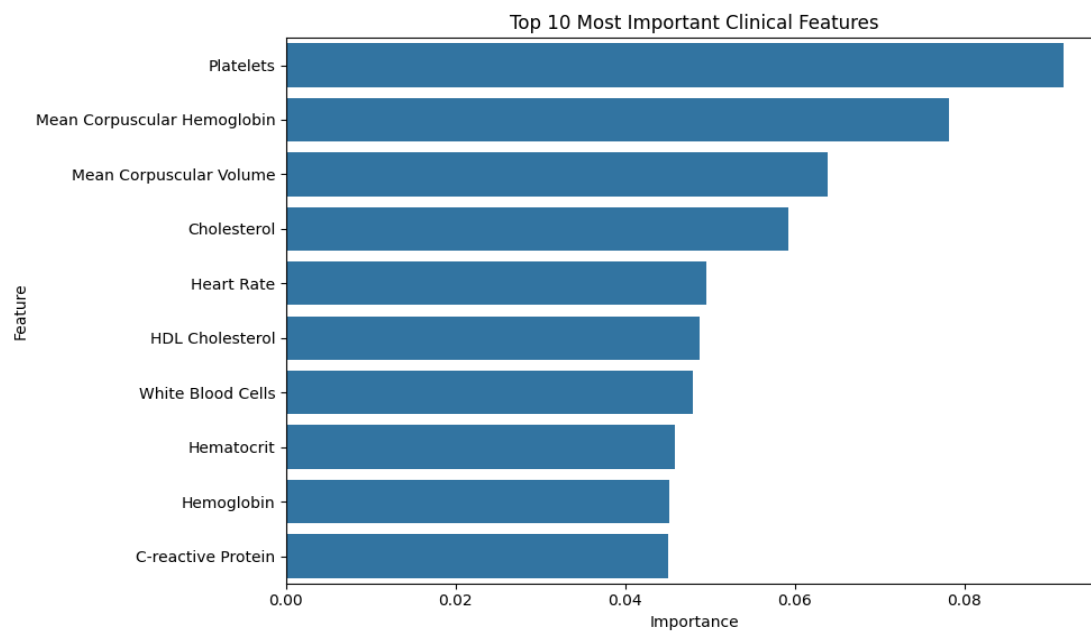
Sample 10

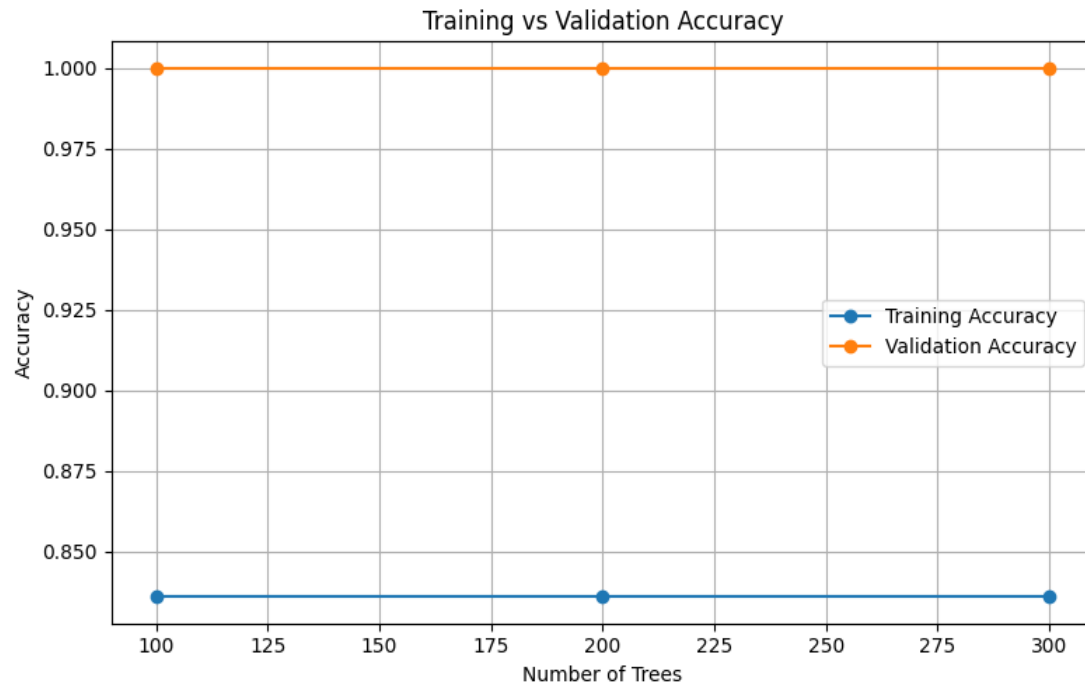
Actual : Diabetes  
Predicted : Diabetes  
Result : Correct



#### CLASSIFICATION REPORT

	precision	recall	f1-score	support
Anemia	0.36	0.43	0.39	756
Diabetes	0.67	0.48	0.56	2646
Healthy	0.30	0.73	0.43	396
Thalassemia	0.25	0.35	0.29	432
Thrombocytopenia	0.67	0.12	0.21	144
accuracy			0.47	4374
macro avg	0.45	0.42	0.38	4374
weighted avg	0.54	0.47	0.48	4374





#### FINAL MODEL ACCURACY SUMMARY

=====  
Training Accuracy : 83.59%  
Validation Accuracy : 100.00%  
Test Accuracy : 46.71%

## USER INTERFACE:

Disease Predictor

DASHBOARD DOCTOR VIEW LOGOUT

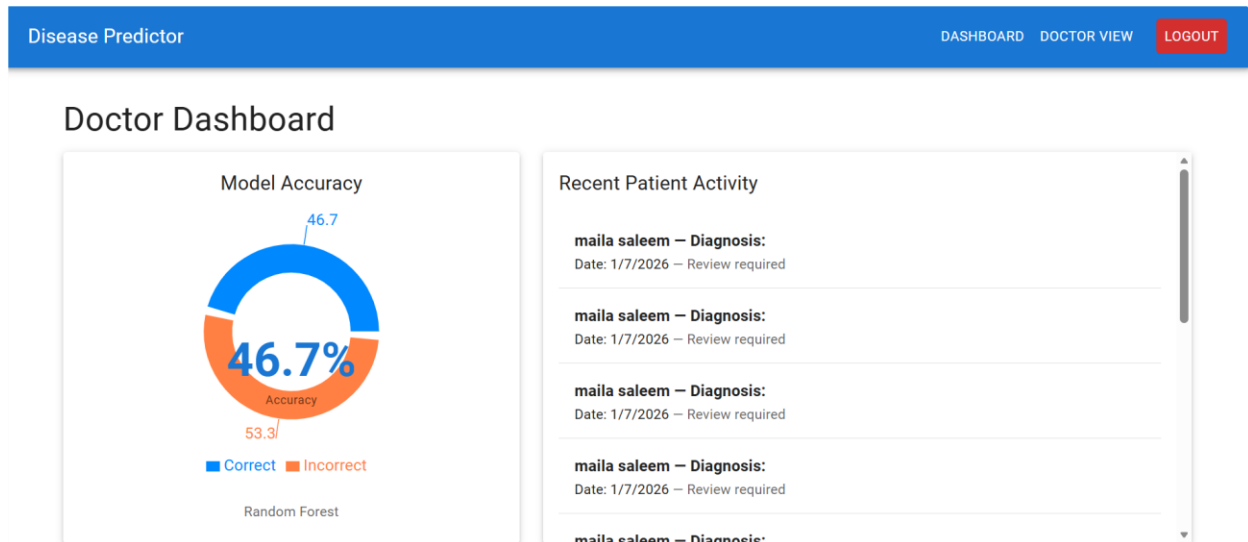
Welcome, Maila Saleem

New Prediction

START CHECKUP

Your History

VIEW REPORTS



## SCOPE:

- Dataset cleaning and preprocessing
- Handling missing values properly
- Implementing leakage-safe scaling
- Label encoding without test contamination
- Applying noise injection to avoid overfitting
- Training Balanced Random Forest model
- Incremental tree training with early stopping
- Model evaluation on validation and test data
- Visualization of results
- Feature importance analysis

**Git link:** <https://github.com/mailasaleem09/AI-PROJECT.git>